Data-Driven Constrained Control for Power Systems

Daniel Tabas

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2024

Reading Committee:

Baosen Zhang, Chair

Daniel Kirschen

Lillian Ratliff

Program Authorized to Offer Degree:
Electrical and Computer Engineering

University of Washington

Abstract

Data-Driven Constrained Control for Power Systems

Daniel Tabas

Chair of the Supervisory Committee:
Baosen Zhang
Electrical and Computer Engineering

Power systems are evolving rapidly along several fronts. Renewable energy installations at the distribution and transmission levels are introducing new levels of variability and volatility, which are only exacerbated by increasing occurrences of extreme whether. Meanwhile, inverter-based controls and battery energy storage systems are providing new levels of flexibility while phasor measurement units and smart meters are improving the observability of power systems. Taken together, these changes both necessitate and enable new strategies for power system operation and control. These new strategies must utilize the fast responsiveness of inverter-based devices to compensate for increased volatility of power systems. They must do so in a way that respects the engineering constraints of power systems while handling uncertainty effectively.

This dissertation addresses the design of policy functions for frequency and voltage regulation in modern power systems considering the joint challenges of computational complexity, uncertainty, and safety. First, we consider the problem of safe exploration for frequency regulation from the perspective of centralized control. We then move to a decentralized setting motivated by building energy management and develop new algorithms that yield probabilistic safety guarantees at execution time. Theoretical results are backed up by simulations demonstrating the advantages of the

proposed methods.

TABLE OF CONTENTS

Page

List of Figures iii

List of Tables vi

Chapter 1: Introduction 1
1.1 Motivation 1
1.2 Context within sequential decision making analytics 4
1.3 Context within power systems literature 6
1.4 Outline 7

Chapter 2: Computationally Efficient Safe Reinforcement Learning for Power Systems 10
2.1 Problem Formulation 12
2.2 Controller Design 15
2.3 Simulations 20

Chapter 3: Safe and Efficient Model Predictive Control Using Neural Networks: An Interior Point Approach 27
3.1 Problem Formulation 30
3.2 Phase I: Finding a Feasible Point 33
3.3 Phase II: Optimizing Performance 35
3.4 Simulations 38

Chapter 4: Interpreting Primal-Dual Algorithms for Constrained Multiagent Reinforcement Learning 43
4.1 Problem formulation 47
4.2 Occupation measure 49
4.3 Discounted risk metrics 52

4.4 Primal-dual value functions 54
4.5 Simulations 57

Chapter 5: Conclusion 63

Bibliography 68

Appendix A: Additional Results for Chapter 2 79
A.1 Derivation of (2.11) 79
A.2 Additional lemmas 79
A.3 Proof of Lemma 2.1 80
A.4 Proof of Lemma 2.2 81
A.5 Proof of Theorem 2.1 81

Appendix B: Additional Results for Chapter 4 83
B.1 Theoretical results 83
B.2 Simulation details 86

LIST OF FIGURES

Figure Number Page

1.1 Annual U.S. electricity net generation by energy source for all sectors (1990-2022). Data source: U.S. Energy Information Administration, Monthly Energy Review, October 2023, Table 7.2a Electricity Net Generation Total (All Sectors) and Table 10.6 Solar Electricity Net Generation. EIA note: Zero-carbon generation does not include generation from distributed energy sources or small-scale solar PV. . 2

1.2 Large-scale battery storage capacity additions by region (2010–2022). Data source: U.S. Energy Information Administration, 2022 Form EIA-860 Early Release, Annual Electric Generator Report. 2

1.3 Applications served by large-scale battery storage (2022). Data source: U.S. Energy Information Administration, 2022 Form EIA-860 Early Release, Annual Electric Generator Report. 3

2.1 Policy network architecture for safe learning. The components are: Kx_t, a safe linear feedback included for numerical stability; $\Omega(x_t)$, the set of safe actions from observed state x_t; $\psi_\theta(x_t)$, a neural network; and the closed-form *gauge map* which maps neural network outputs to the current set of safe actions $\Omega(x_t)$. 16

2.2 Action of the gauge map from $\mathbb{B}_\infty$ to randomly generated $\mathcal{Q}$, with the $\frac{1}{4}$, $\frac{1}{2}$, and $\frac{3}{4}$ level sets of the respective gauge functions shown in white. For each point $v \in \mathbb{B}_\infty$ and its image $w \in \mathcal{Q}$, v and w have the same direction and $\gamma_{\mathcal{Q}}(w) = \|v\|_\infty$. 18

2.3 In the policy network, the gauge map is used to map virtual actions to safe actions. 20

2.4 Illustration of 9-bus power system model. 22

2.5 Accumulated cost over several test trajectories with a fixed initial condition and randomly generated disturbance sequences, showing that the RCI policy network achieves better performance than the safe linear feedback. 24

2.6 Maximum angle deviation per training episode for the safe policy network (blue) and the baseline policy network with soft penalty (orange). The safe policy network guarantees safety during training, while soft penalties eventually drive the baseline policy towards constraint satisfaction. 25

2.7 Even though soft penalties succeed in driving policies to be safe during training, they do not necessarily provide safety during testing. In this example, a policy network that was safe during training (Fig. 2.6) still exhibits constraint violations during testing. In contrast, starting from the same initial conditions and subject to the same disturbance sequence, the proposed safe policy network guarantees constraint satisfaction. 26

3.1 Illustration of the interior point approach to learning-based MPC. The set $\mathcal{F}(x_0)$ represents the MPC feasible set, while $\mu_0(x_0)$ and $\mu_\theta(x_0)$ are control input sequences representing solutions to the Phase I and Phase II problems, respectively. The neural network μ_θ moves the Phase I solution to a more optimal solution. 30

3.2 The proposed control policy uses a neural network combined with the Phase I solution and a *gauge map* to constrain the decision $\mathbf{u}$ to the MPC feasible set $\mathcal{F}(x_0)$. The first action from the sequence $\mathbf{u}$ is extracted and implemented. On the right, the action of the gauge map is illustrated. 36

3.3 Training trajectories for the three types of neural netwokrs. Our proposed Gauge-based approach achieves lower cost at a much faster rate. 41

3.4 Solve time vs. trajectory cost for the networks under consideration applied to the 3-state system. The Gauge NN is Pareto-efficient in terms of cost and computation time compared to the other techniques with safety guarantees (Online MPC and Projection NN). 42

4.1 Building energy management with a voltage constraint at the point of common coupling. Key: PV = photovoltaic array, BESS = battery energy storage system, SWH = smart water heater, SCC = smart climate control system. 45

4.2 Example of the occupation measure for various levels of γ. 50

4.3 Effective horizon length as a function of γ. 51

4.4 Example of VaR and CVaR at risk level $\beta = 0.9$. 54

4.5 Temporal difference error trajectories in a simple policy evaluation task. 56

4.6 $\Pr\{C(x) \geq 0.1 \mid x \sim \mu_\gamma^\pi\}$ measured throughout training. Key: SC = structured critic, MP = modified penalty (Prop. 4.4). Both modifications speed convergence to a safe policy. The shaded region represents ± 1 standard deviation across 5 training runs. 58

4.7 $\text{CVaR}(\beta = 0.9, C, \mu_\gamma^\pi)$ measured throughout training. Key: SC = structured critic, MP = modified penalty (Prop. 4.5). The dashed lines represent the CVaR upper bound used in Prop. 4.5. The panel on the right shows progress toward the original objective through the total original returns, $\sum_{i=1}^{2} \prod_{t=0}^{T} r_t^i$, without penalty terms. The shaded region represents ± 1 standard deviation across 5 training runs. The rewards increase then decrease because the agents first learn to navigate towards the landmark, which is outside the safe region, then learn to back off to satisfy the constraint. 60

4.8 Learning curves for the 3-building scenario. The horizontal axis is in increments of 1000 training episodes, for all plots. *Top row:* Individual agent costs (left) and total cost (right), corresponding to building temperature deviations. *Second row:* average actor loss (left) and critic loss (right). *Third row:* voltage excursions from the interval $[0.95, 1.05]$ per unit (cumulative per episode). *Last row:* Dual variables and CVaR upper bound from Proposition 4.5. 62

B.1 Evaluation of the discounted sum constraint throughout training, showing that the structured critic helps the actor to find safer policies faster. Each line and shaded region represents the mean and standard deviation over 5 training runs. Key: SC = structured critic. 90

LIST OF TABLES

Table Number Page

3.1 Hyperparameters for the three neural networks. 40
3.2 Open-loop test results. 41

B.1 Simulation hyperparameters. 88
B.2 Accuracy of CVaR upper bound. 91

DEDICATION

To my grandparents

Joseph Horowitz

Eileen Horowitz

David Tabas

Harriet Tabas Brecher

Eugene Brecher

May their memory be a blessing

Chapter 1
INTRODUCTION

1.1 Motivation

The renewable energy revolution is well underway (Figure 1.1). Large-scale solar and wind installations have grown to encompass 14% of all electricity generation in the U.S., up from 4% a decade ago [3]. With the growth of variable renewable energy sources comes the need for energy storage, and battery installations have also ballooned in recent years (Figure 1.2). As shown in Figure 1.3, batteries do much more than just shift energy demand in time. They also participate in ancillary (backup) services such as frequency regulation and spinning reserve, thereby providing a buffer against the intermittency of renewable energy sources at multiple timescales. These trends are mirrored at the distribution level and in microgrids where rooftop solar and home energy storage devices are seeing rapid growth.

Along with these trends, new challenges and opportunities are arising. Inverter-connected resources contribute to the overall power supply without increasing the mechanical inertia of the system (provided by synchronous generators), thereby reducing stability margins [74]. At the same time, the variability of wind and solar energy resources threatens the reliability of the system [97]. Finally, the sheer quantity of new devices in the system makes coordination difficult from a computational perspective. In terms of opportunities, inverter-connected energy resources can be controlled on fast timescales and programmed with nearly arbitrary control laws. Thus, sophisticated control techniques can be implemented to deal with some of the challenges described above.

These challenges and opportunities necessitate new algorithms for sequential de-

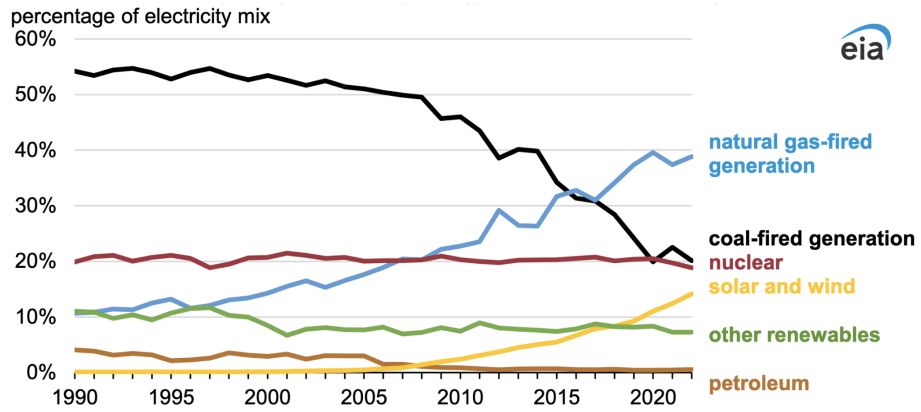

Figure 1.1: Annual U.S. electricity net generation by energy source for all sectors (1990-2022). Data source: U.S. Energy Information Administration, Monthly Energy Review, October 2023, Table 7.2a Electricity Net Generation Total (All Sectors) and Table 10.6 Solar Electricity Net Generation. EIA note: Zero-carbon generation does not include generation from distributed energy sources or small-scale solar PV.

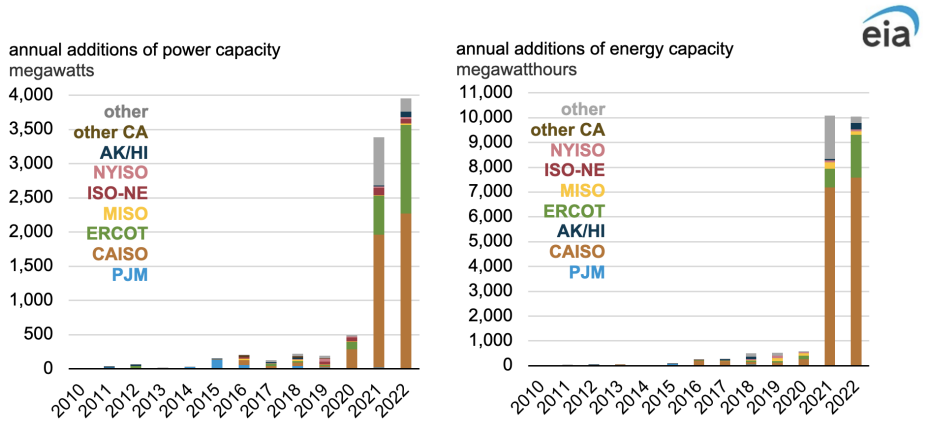

Figure 1.2: Large-scale battery storage capacity additions by region (2010–2022). Data source: U.S. Energy Information Administration, 2022 Form EIA-860 Early Release, Annual Electric Generator Report.

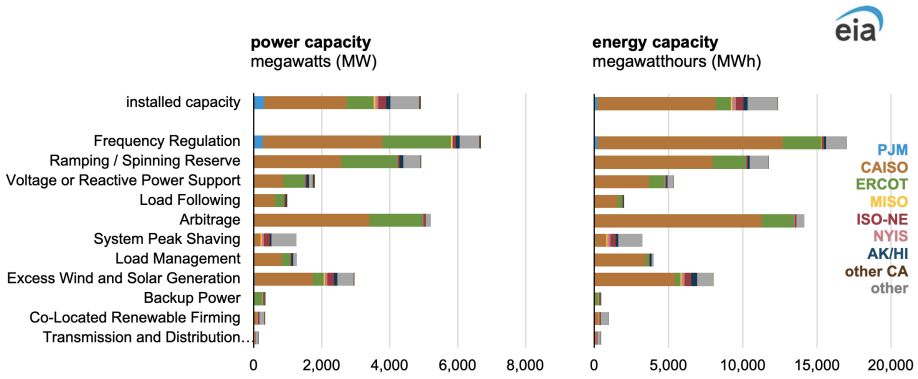

Figure 1.3: Applications served by large-scale battery storage (2022). Data source: U.S. Energy Information Administration, 2022 Form EIA-860 Early Release, Annual Electric Generator Report.

cision making in power systems. New techniques must address three fundamental issues in power systems:

1. *Constraints.* Power systems are subject to many types of constraints. In particular, we consider operational constraints on the frequencies, voltages, and line flows throughout a power system. These values must not exceed certain bounds in order to protect generation and transmission equipment.

2. *Uncertainty.* The power injections from wind and solar plants are difficult to predict and quick to vary due to the fluctuating nature of the weather. In addition, power demand can fluctuate and equipment can fail. The system operator must be able to maintain the system even in unexpected scenarios.

3. *Computational limits.* Energy resources may be able to receive real and reactive power setpoints from centralized controllers on longer timescales, such as seconds or minutes, but on shorter timescales, they must be programmed with control laws to decide how to act in order to preserve the system in case of an

emergency. Due to the short timescales and limited computational resources on individual devices, the control laws must be quick to evaluate and cannot involve solving large optimization problems.

The purpose of this dissertation is to propose new sequential decision making algorithms that address these three complexities.

1.2 Context within sequential decision making analytics

Policies for sequential decision making can be grouped into four "meta-classes" [81]. Policy function approximations (PFAs) are closed-form decision rules comprising everything from linear feedback to neural networks. Cost function approximations (CFAs) are parameterized optimization problems that are usually used to solve problems without state dynamics such as bandit problems. Value function approximations (VFAs) such as value iteration and Q learning trade off between immediate reward and future value. Direct lookahead approximations (DLAs), most notably model predictive control (MPC), directly optimize a sequence of decisions using some approximate model of the future.

The last three meta-classes (CFAs, VFAs, and DLAs) all involve solving a optimization problem at each time step. Compared to PFAs, these strategies handle constraints with ease but at the cost of computational complexity. In this dissertation we focus on designing PFAs with safety guarantees in order to improve the state of the art in the inherent tradeoff between safety and computational complexity.

However, the best approach to policy design is application-specific and often involves combining techniques from each of the meta-classes. In Chapter 2, VFAs are used to guide a gradient-based policy search using actor-critic reinforcement learning. In Chapter 3, DLAs appear when we use a known, differentiable loss function to optimize a PFA. And in Chapter 4, CFAs appear when we include a tunable parameter in the objective function to incentivize safe behavior.

Within the realm of designing safe policy functions, there exist many strategies as well as many different notions of safety [41]. Safe *exploration* involves the design of correct-by-construction policy functions that uphold state and action constraints from the first episode of training onward. By contrast, the paradigm of safe *execution* permits constraint violations during training while providing guarantees on the behavior of the trained policy. Notions of safety can either be absolute (precluding any occurrence of a constraint violation) or probabilistic (limiting the risk of constraint violations in some way). More stringent notions of safety require more model information in order to enforce.

In Chapters 2 and 3, we consider the problem of safe exploration with absolute guarantees of safety. There are three main approaches to this class of problems [51]. Action *replacement* uses a classifier to identify unsafe actions and replace them with safe ones provided by a backup controller before they are executed. Action *projection* involves projecting the action recommended by the policy function onto a time-varying set of safe actions. Action *masking* involves directly modifying the policy function so that it can only output safe actions. In Chapters 2 and 3, we focus on the masking approach. When using the replacement approach, it is difficult to verify the optimality of the backup controller or the replacement operation. When using the projection approach, it is difficult to avoid the computational burden of the projection operation. By contrast, we propose masking procedures that are closed-form and differentiable. The closed-form property ensures computational efficiency in contrast to projection-based approaches. The differentiable property enables backpropagation so that the masking procedure does not interfere with the optimality of a gradient-based policy search, in contrast to replacement-based approaches. Masking for problems with continuous state and action spaces is not well-studied [51], and the work in Chapters 2 and 3 addresses this gap.

However, the masking approach requires a degree of model information that is not available in multi-agent settings. In Chapter 4, we study penalty-based approaches

that modify the reward function of each agent in order to incentivize safe behavior. Although popular in the literature, penalty-based methods suffer from a lack of interpretable safety guarantees. In Chapter 4, we provide a new perspective showing that penalty methods can yield meaningful probabilistic safety guarantees in the safe execution paradigm.

1.3 Context within power systems literature

Recent review articles [21, 42, 107] have highlighted the potential role of reinforcement learning in leveraging new data sources such as smart meters and phasor measurement units to figure out how to best operate power systems in the face of increased volatility, uncertainty, and complexity. To this end, several important applications have been identified: frequency regulation, voltage regulation, and economic dispatch to name a few. In this context, there has been a pronounced need for new reinforcement learning algorithms that operate safely, meaning with guarantees on stability or constraint satisfaction.

Chapter 2 focuses on reinforcement learning for primary frequency control. Recent work on this topic has considered safety in the context of frequency stability [27] or has used soft penalties to enforce safe behavior without any guarantees [48]. Other works disregard safety criteria entirely [99]. In Chapter 2, we provide a method to guarantee that reinforcement-learning based frequency controllers will satisfy hard constraints on frequency and angle deviations.

Chapter 4 studies building energy management under distribution system-level voltage constraints. A closely related work is [80] which trains RL agents to minimize voltage violations, rather than training them to optimize some building-relevant objective subject to voltage constraints. In [87], a method is proposed to use RL for decentralized voltage control with stability guarantees. The objective function contains a "transient" term reflecting the discounted accumulation of voltage violations over time, in contrast to more traditional pointwise-in-time or steady-state voltage

criteria. The authors in [87] identify an outstanding need to unify these disparate perspectives, and Chapter 4 makes a substantial contribution towards this goal.

1.4 Outline

In Chapter 2, we propose a computationally efficient approach to safe reinforcement learning (RL) for frequency regulation in power systems with high levels of variable renewable energy resources. The approach draws on set-theoretic control techniques to craft a neural network-based control policy that is guaranteed to satisfy safety-critical state constraints, without needing to solve a model predictive control or projection problem in real time. By exploiting the properties of robust controlled-invariant polytopes, we construct a novel, closed-form "safety-filter" that enables end-to-end safe learning using any policy gradient-based RL algorithm. We then apply the safety filter in conjunction with the deep deterministic policy gradient (DDPG) algorithm to regulate frequency in a modified 9-bus power system, and show that the learned policy is more cost-effective than robust linear feedback control techniques while maintaining the same safety guarantee. We also show that the proposed paradigm outperforms DDPG augmented with constraint violation penalties[1].

In Chapter 3, we extend the techniques developed in Chapter 2 to situations in which the reward function is known, leading to a model predictive control (MPC)-inspired policy search. MPC provides a useful means for controlling systems with constraints, but suffers from the computational burden of repeatedly solving an optimization problem in real time. Offline (explicit) solutions for MPC attempt to alleviate real time computational challenges using either multiparametric programming or machine learning. The multiparametric approaches are typically applied to linear or quadratic MPC problems, while learning-based approaches can be more flexible

[1]Paragraph adapted from D. Tabas and B. Zhang, "Computationally Efficient Safe Reinforcement Learning for Power Systems," 2022 American Control Conference (ACC), Atlanta, GA, USA, 2022, pp. 3303-3310.

and are less memory-intensive. Existing learning-based approaches offer significant speedups, but the challenge becomes ensuring constraint satisfaction while maintaining good performance. In this chapter, we provide a neural network parameterization of MPC policies that explicitly encodes the constraints of the problem. By exploring the interior of the MPC feasible set in an unsupervised learning paradigm, the neural network finds better policies faster than projection-based methods and exhibits substantially shorter solve times. We use the proposed policy to solve a robust MPC problem, and demonstrate the performance and computational gains on a standard test system from the literature[2].

In Chapter 4, we turn our attention towards building energy management which, due to privacy considerations, is inherently a multi-agent problem [80]. This requires investigating a relatively relaxed notion of safety in order to come to a meaningful solution. Most constrained multiagent reinforcement learning (C-MARL) algorithms use a primal-dual approach to enforce constraints through a penalty function added to the reward [64]. In this chapter, we study the structural effects of this penalty term on the MARL problem. First, we show that the standard practice of using the constraint function as the penalty leads to a weak notion of safety. However, by making simple modifications to the penalty term, we can enforce meaningful probabilistic (chance and conditional value at risk) constraints. Second, we quantify the effect of the penalty term on the value function, uncovering an improved value estimation procedure. We use these insights to propose a constrained multiagent advantage actor critic (C-MAA2C) algorithm. Simulations in a simple constrained multiagent environment affirm that our reinterpretation of the primal-dual method in terms of probabilistic constraints is effective, and that our proposed value estimate accelerates convergence to a safe joint policy. We also support our findings using a 3-building simulation

[2]Paragraph adapted from D. Tabas and B. Zhang, "Safe and Efficient Model Predictive Control Using Neural Networks: An Interior Point Approach," 2022 IEEE 61st Conference on Decision and Control (CDC), Cancun, Mexico, 2022, pp. 1142-1147.

environment including various distributed energy resources and a distribution system solver[3].

The dissertation concludes with some closing remarks and directions for future research.

[3]Paragraph adapted from Tabas, D., Zamzam, A.S. and Zhang, B. (2023). Interpreting Primal-Dual Algorithms for Constrained Multiagent Reinforcement Learning. *Proceedings of The 5th Annual Learning for Dynamics and Control Conference*, in *Proceedings of Machine Learning Research* 211:1205-1217.

Chapter 2

COMPUTATIONALLY EFFICIENT SAFE REINFORCEMENT LEARNING FOR POWER SYSTEMS[1]

Power systems are a quintessential example of safety-critical infrastructure, in which the violation of operational constraints can lead to large blackouts with high economic and human cost. As variable renewable energy resources are integrated into the grid, it becomes increasingly important to ensure that the system states, such as generator frequencies and bus voltages, remain within a "safe" region defined by the operators [24].

The design of safe controllers concerns the ability to ensure that an uncertain dynamical system will satisfy hard state and action constraints during execution of a control policy [65, 71]. Recently, set-theoretic control [13] has been applied to a wide range of safety-critical problems in power system operation [36, 106]. This approach involves computing a *robust controlled-invariant set* (RCI) along with an associated control policy which is guaranteed to keep the system state inside the RCI [13,36,37]. If the RCI is contained in the feasible region of the (safety-critical) state constraints, then the associated control policy is considered to be safe.

However, the set-theoretic approach requires several simplifying assumptions for tractability, leading to controllers with suboptimal performance. First, the disturbances to the system are assumed to be bounded in magnitude but otherwise arbitrary [13, 106]. Second, the RCIs must be restricted to simple geometric objects such as polytopes or ellipsoids [69]. Third, many approaches select an RCI and control

[1]Adapted from D. Tabas and B. Zhang, "Computationally Efficient Safe Reinforcement Learning for Power Systems," 2022 American Control Conference (ACC), Atlanta, GA, USA, 2022, pp. 3303-3310.

policy in tandem, which usually requires the control policy to be linear and forces a tradeoff between performance and robustness [12, 62, 73]. Fourth, nonlinear systems must be treated as linear systems plus an unknown-but-bounded linearization error [36].

Once an RCI is generated using the conservative assumptions listed above, data-driven approaches can use learning to improve performance with respect to the true behavior of the disturbances and nonlinearities without risk of taking unsafe actions [7, 25, 44, 95]. However, these techniques require solving a model predictive control (MPC) or projection problem each time an action is executed, which may be too computationally expensive. Several approaches that avoid repeatedly solving an optimization problem have also been proposed. One such approach involves tracking the vertices of the set of safe actions, and using a neural network to specify an action by choosing convex weights on these vertices. However, this is only possible when the RCI has exceedingly simple geometry [109]. Other strategies only guarantee safety in expectation, and do not rule out constraint violations in every situation [1, 102]. Controllers with Lyapunov stability or robust control guarantees have also been proposed [27, 29, 33], but stability does not always translate to constraint satisfaction.

In this chapter, we present a method to design safe, data-driven, and closed-form control policies for frequency regulation in power systems. Our approach combines the advantages of set-theoretic control and learning. In particular, we use simple linear controllers to find a maximal RCI, and then use reinforcement learning (RL) to train a neural network-based controller that improves performance while maintaining safety. The safety of this control policy is accomplished by constraining the output of the neural network to the present set of safe actions. By leveraging the structure of polytopic RCIs, we construct a closed-form *safety filter* to map the neural network's output into the safe action set without solving an MPC or projection problem. The safety filter is differentiable, allowing end-to-end training of the neural network using any policy gradient-based RL algorithm. We demonstrate our proposed control de-

sign on a frequency regulation problem in a 9-bus power system model consisting of several generators, loads, and inverter-based resources (IBRs). The simulation results demonstrate that our proposed policy maintains safety and outperforms safe linear controllers without repeatedly solving an optimization problem in real time.

We focus on applying our algorithm to the problem of primary frequency control in power systems. Frequency is a signal in the grid that indicates the balance of supply and demand. Generators typically respond to the change in frequency by adjusting their power output to bring the frequency back to nominal (e.g., 60 Hz in the North American system) [53, 108]. For conventional generators, these responses are limited to be linear (possibly with a dead-band). In contrast, IBRs such as solar, wind and battery storage can provide almost any desired response to frequency changes, subject to some actuation constraints [2]. Currently, however, these resources still use linear responses, largely because of the difficulty in designing nonlinear control laws. Recently, RL based methods have been introduced in the literature (see, e.g. [21] and the references within). However, most approaches treat safety and constraint satisfaction as soft penalties, and cannot provide any guarantees [21, 55, 99].

The rest of the chapter is organized as follows. Section 2.1 introduces the power system model and formulates the problem of safety-critical control from a set-theoretic perspective. Section 2.2 describes the proposed controller design. Section 2.3 presents simulation results for the modified 9-bus power system.

2.1 Problem Formulation

2.1.1 Model assumptions

In this chapter we are interested in a linear system with control inputs and disturbances. We write the system evolution as

$$x_{t+1} = Ax_t + Bu_t + Ed_t, \tag{2.1}$$

where $x_t \in \mathbb{R}^n$, $u_t \in \mathbb{R}^m$ and $d_t \in \mathbb{R}^p$ are vectors of the state variables, control inputs, and disturbances at time t. We assume the disturbance d_t is bounded but otherwise can take arbitrary values. More precisely, we assume that d_t lies in a compact set. This boundedness assumption on d_t is fairly general, since it allows the disturbances to capture uncontrolled input into the system, model uncertainties in A, B, and E, and linearization error. For more compact notation, we will sometimes summarize (2.1) as $x^+ = f(x_t, u_t, d_t)$.

The constraints on inputs are $u_t \in \mathcal{U} \subset \mathbb{R}^m$ and $d_t \in \mathcal{D} \subset \mathbb{R}^p$ for all t. The sets $\mathcal{U}$ and $\mathcal{D}$ are assumed to be polytopes, defined as the bounded intersection of a finite number of halfspaces or linear inequalities [18]. Specifically, $\mathcal{U}$ and $\mathcal{D}$ are defined as

$$\mathcal{U} = \{u \in \mathbb{R}^m \mid -\bar{u} \leq V_u u \leq \bar{u}\} \text{ and} \tag{2.2}$$

$$\mathcal{D} = \{d \in \mathbb{R}^p \mid -\bar{d} \leq V_d d \leq \bar{d}\}. \tag{2.3}$$

In safety-critical control problems such as frequency regulation, operators want to keep the system states within hard constraints. For example, frequencies are generally kept within a tenth of a hertz of the nominal frequency and rotor angle deviations are limited for stability considerations [53]. We use the set

$$\mathcal{X} = \{x \in \mathbb{R}^n \mid -\bar{x} \leq V_x x \leq \bar{x}\} \tag{2.4}$$

to denote the constraints that the state x must satisfy in real-time.

2.1.2 Safety-critical control

Because of the presence of disturbances, it may not be possible for the system state to always remain in $\mathcal{X}$. Some states close to the boundary of $\mathcal{X}$ could be pushed out by a disturbance no matter the control action, while for other states in $\mathcal{X}$, there may exist a control action such that no disturbance would push the state outside of the prescribed region. This motivates the definition of a *robust controlled-invariant set*.

Definition 2.1 (Robust controlled-invariant set (RCI) [13])**.** An RCI is a set $\mathcal{S}$ for which there exists a feedback control policy $u_t = \pi_0(x_t) \in \mathcal{U}$ ensuring that all system trajectories originating in $\mathcal{S}$ will remain in $\mathcal{S}$ for all time, under any disturbance sequence $d_t \in \mathcal{D}$.

If $\mathcal{S}$ is contained in $\mathcal{X}$, then π_0 is a safe policy. Often, the goal is to find the policy that maximizes the size of $\mathcal{S}$ while being contained in $\mathcal{X}$, since it corresponds to making most of the acceptable states safe [69]. In general, this is a difficult problem. Fortunately, if we restrict the policy to be linear, there are many well-studied techniques that have been shown to be successful at producing large safety sets [14, 61, 62, 93].

In this chapter, we assume that $\mathcal{S}$ is a polytope described by $2r$ linear inequalities, and that π_0 is a linear feedback control policy. Specifically, we assume

$$\mathcal{S} = \{x \in \mathbb{R}^n \mid -\bar{s} \leq V_s x \leq \bar{s}\} \subseteq \mathcal{X} \text{ and} \tag{2.5}$$

$$\pi_0(x) = Kx \tag{2.6}$$

where $V_s \in \mathbb{R}^{r \times n}$, $\bar{s} \in \mathbb{R}^r$, and $Kx \in \mathcal{U}$ for all $x \in \mathcal{S}$. For robustness, we choose the largest RCI satisfying (2.5). The algorithm used for choosing $(\mathcal{S}, \pi_0)$ is described in [61]. The algorithm uses a convex relaxation to find an *approximately* maximal RCI $\mathcal{S}$ and an associated K as the solution to an SDP. The objective of the SDP is to maximize the volume of the largest inscribed ellipsoid inside $\mathcal{S}$.

Of course, a linear policy that maximizes the size of $\mathcal{S}$ may not lead to satisfactory control performance. The set $\mathcal{S}$ is chosen jointly with the policy π_0, but there could be many policies (not necessarily linear) that keep $\mathcal{S}$ robustly invariant. We want to optimize over this class of nonlinear policies to improve the performance of the system. To explore the full range of safe policies, we define the *safe action set at time* t as

$$\Omega(x_t) := \{u_t \in \mathcal{U} \mid x_{t+1} \in \mathcal{S},\ \forall\ d_t \in \mathcal{D}\} \tag{2.7}$$

where it is assumed that $x_t \in \mathcal{S}$. By induction, any policy that chooses actions from $\Omega(x_t)$ is a safe policy [13].

We define the set of safe policies with respect to the RCI $\mathcal{S}$ as

$$\Pi := \{\pi : \mathbb{R}^n \to \mathbb{R}^m \mid \pi(x_t) \in \Omega(x_t),\ \forall\ x_t \in \mathcal{S}\}. \tag{2.8}$$

Given $\mathcal{S}$, we search for a policy by optimizing over Π:

$$\min_{\pi \in \Pi} \mathbb{E}_{x_0 \in \mathcal{S}, d_t \in \mathcal{D}} \left[\frac{1}{T} \sum_{t=1}^{T} J(x_t, u_t) \right] \tag{2.9a}$$

$$\text{subject to: } x_{t+1} = Ax_t + Bu_t + Ed_t \tag{2.9b}$$

$$u_t = \pi(x_t) \tag{2.9c}$$

where $\mathbb{E}_{x_0 \in \mathcal{S}, d_t \in \mathcal{D}}$ is the expectation with respect to randomness in initial conditions and in the sequence of disturbances, and $J(x_t, u_t)$ is the cost associated with occupying state x_t and taking action u_t. To estimate (2.9a), d_t is sampled from $\mathcal{D}$ but treated as stochastic, so that standard RL algorithms can be used to solve (2.9) [95]. This relaxation does not require thorough sampling of $\mathcal{D}$ to preserve safety, since the constraint $\pi \in \Pi$ imposes state and input constraint satisfaction for *all* possible disturbances $d_t \in \mathcal{D}$. The solution of (2.9) depends on the distribution of d_t over $\mathcal{D}$ but safety is guaranteed for any $\pi \in \Pi$.

One example of a cost function that can be used in (2.9) is the classical LQR cost on state and control [45], but other non-quadratic cost functions can also be used. For example, for sparsity-promoting controllers, we may set $J(x_t, u_t) = x_t^T Q x_t + c\|u_t\|_1$, where $Q \succeq 0$ and $c > 0$ [34].

2.2 Controller Design

In this section, we describe how set-theoretic control techniques can be used to create a safety guarantee for data-driven controllers without solving an MPC or projection problem in real time. Since d_t is unknown, data-driven approaches for choosing π are

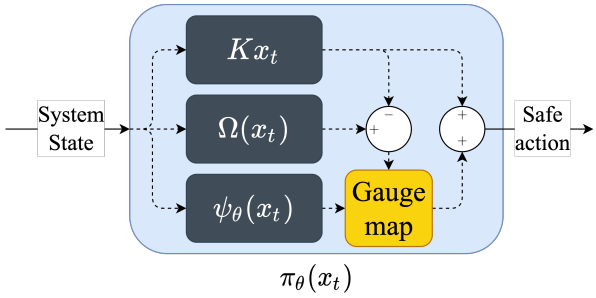

Figure 2.1: Policy network architecture for safe learning. The components are: Kx_t, a safe linear feedback included for numerical stability; $\Omega(x_t)$, the set of safe actions from observed state x_t; $\psi_\theta(x_t)$, a neural network; and the closed-form *gauge map* which maps neural network outputs to the current set of safe actions $\Omega(x_t)$.

appropriate if safety guarantees can be maintained. For control problems with continuous state and action spaces, one class of RL algorithms involves parameterizing π as a neural network or other function approximator and using stochastic optimization to search over the parameters of that function class for a (locally) optimal policy.

A common approach to safety-critical control with RL is to combine a model-predictive controller with a neural network providing an action recommendation or warm start [7, 95]. However, this makes it difficult to search over Π efficiently and leads to control policies with higher computational overhead. One optimization-free approach involves tracking the vertices of $\Omega(x_t)$ and using a neural network to choose convex weights on the vertices of $\Omega(x_t)$. However, this is only possible when $\mathcal{S}$ has exceedingly simple geometry [109]. While it is difficult to constrain the output of a neural network to arbitrary polytopes such as $\Omega(x_t)$, it is easy to constrain the output to $\mathbb{B}_\infty$, the ∞-norm unit ball in $\mathbb{R}^m$, using activation functions like sigmoid or hyperbolic tangent in the output layer. By establishing a correspondence between points in $\mathbb{B}_\infty$ and points in $\Omega(x_t)$, we will use neural network-based controllers to parameterize Π.

In particular, we construct a class of safe, differentiable, and closed-form policies

π_θ, parameterized by θ, that can approximate any policy in Π. The policy first chooses a "virtual" action in $\mathbb{B}_\infty$ using a neural network ψ_θ. The policy then uses a novel, closed-form, differentiable "safety filter" to equate $\psi_\theta(x_t)$ with an action in $\Omega(x_t)$. Figure 2.1 illustrates the way ψ_θ, Ω, and π_0 are interconnected using a novel *gauge map* in order to form the policy π_θ. In order to efficiently map between $\mathbb{B}_\infty$ and $\Omega(x_t)$, we now introduce the concepts of *C-sets* and *gauge functions*.

Definition 2.2 (C-set [13])**.** A *C-set* is a set that is convex and compact and that contains the origin as an interior point.

Any C-set can be used as a "measuring stick" in a way that generalizes the notion of a vector norm [13]. In particular, the gauge function (or Minkowski function) of a vector $v \in \mathbb{R}^m$ with respect to a C-set $\mathcal{Q} \subset \mathbb{R}^m$ is given by

$$\gamma_{\mathcal{Q}}(v) = \inf\{\lambda \geq 0 \mid v \in \lambda\mathcal{Q}\}. \tag{2.10}$$

If $\mathcal{Q}$ is a polytopic C-set defined by $\{w \in \mathbb{R}^m \mid F_i^T w \leq g_i, i = 1, \ldots, r\}$, then the gauge function is given by

$$\gamma_{\mathcal{Q}}(v) = \max_i \left\{ \frac{F_i^T v}{g_i} \right\}, \tag{2.11}$$

which is easy to compute since it is simply the maximum over r elements. Equation (2.11) is derived in Appendix A.1. We will use (2.11) to construct a closed-form, differentiable bijection between $\mathbb{B}_\infty$ and $\Omega(x_t)$.

2.2.1 Gauge map

We will first show how to use the gauge function to construct a bijection from $\mathbb{B}_\infty$ to any C-set $\mathcal{Q}$, and will then generalize to the case when $\mathcal{Q}$ does not contain the origin as an interior point. For any $v \in \mathbb{B}_\infty$, we define the *gauge map from* $\mathbb{B}_\infty$ *to* $\mathcal{Q}$ as

$$G(v|\mathcal{Q}) = \frac{\|v\|_\infty}{\gamma_{\mathcal{Q}}(v)} \cdot v. \tag{2.12}$$

We assign $G(0|\mathcal{Q}) := 0$ and justify this choice in Appendix A.5.

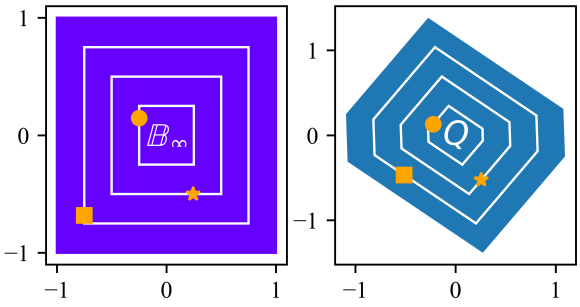

Figure 2.2: Action of the gauge map from $\mathbb{B}_\infty$ to randomly generated $\mathcal{Q}$, with the $\frac{1}{4}$, $\frac{1}{2}$, and $\frac{3}{4}$ level sets of the respective gauge functions shown in white. For each point $v \in \mathbb{B}_\infty$ and its image $w \in \mathcal{Q}$, v and w have the same direction and $\gamma_{\mathcal{Q}}(w) = \|v\|_\infty$.

Lemma 2.1. *For any C-set $\mathcal{Q}$, the gauge map $G : \mathbb{B}_\infty \to \mathcal{Q}$ is a bijection. Specifically, $w = G(v|\mathcal{Q})$ if and only if w and v have the same direction and $\gamma_{\mathcal{Q}}(w) = \|v\|_\infty$.*

The proof of Lemma 2.1 is provided in Appendix A.3. By Lemma 2.1, choosing a point in $\mathbb{B}_\infty$ is equivalent to choosing a point in $\mathcal{Q}$. The action of the gauge map is illustrated in Figure 2.2.

We cannot directly use the gauge map to convert between points in $\mathbb{B}_\infty$ and points in $\Omega(x_t)$, since $\Omega(x_t)$ may not contain the origin as an interior point. Instead, we must temporarily "shift" $\Omega(x_t)$ by one of its interior points, making it a C-set. Lemma 2.2 provides an efficient way to achieve this.

Lemma 2.2. *If $\pi_0(x) = Kx$ is a policy in Π, then for any x_t in the interior of $\mathcal{S}$, the set $\hat{\Omega}_t := [\Omega(x_t) - Kx_t]$ is a C-set.*

The proof of Lemma 2.2 is provided in Appendix A.4. Figure 2.3 illustrates the way the gauge map and Lemma 2.2 are used in the policy network as a safety filter, by transforming the output of the policy network from $\mathbb{B}_\infty$ to $\Omega(x_t)$.

2.2.2 Policy architecture

Theorem 2.1. *Assume the system dynamics and constraints are given by* (2.1), (2.2) *and* (2.3), *and there exists a choice of* $(\mathcal{S}, \pi_0)$ *conforming to* (2.5) *and* (2.6). *Let* $\psi_\theta : \mathcal{S} \to \mathbb{B}_\infty$ *be a neural network parameterized by* θ. *Then for any* x_t *in the interior of* $\mathcal{S}$, *the policy*

$$\pi_\theta(x_t) := G(\psi_\theta(x_t)|\hat{\Omega}_t) + Kx_t \tag{2.13}$$

has the following properties.

1. π_θ *is a safe policy.*

2. π_θ *can be computed in closed form.*

3. π_θ *is differentiable at* x_t.

4. π_θ *can approximate any policy in* Π.

We will comment briefly on the last property and leave the proof of Theorem 2.1 to Appendix A.5. The ability of π_θ to approximate any policy in Π given proper choice of θ is based on the function approximation properties of ψ_θ [46] and the ability of the gauge map to establish a one-to-one correspondence between points in $\mathbb{B}_\infty$ and actions in $\Omega(x_t)$.

2.2.3 Policy optimization through reinforcement learning

We parametrize the search over Π using (2.13) with parameter θ, and we choose θ to optimize (2.9) using policy gradient RL algorithms. The policy gradient theorem from reinforcement learning allows one to use past experience to estimate the gradient of the cost function (2.9a) with respect to θ [10]. This is a standard approach for RL in continuous control tasks [60]. Policy gradient methods require that it be

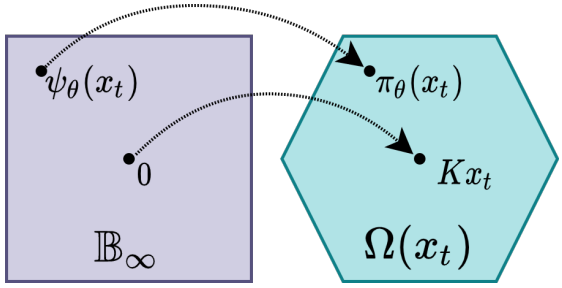

Figure 2.3: In the policy network, the gauge map is used to map virtual actions to safe actions.

possible to compute the gradient of π_θ with respect to θ. More specifically, G must be differentiable (Thm. 2.1, part 3) or else the safety filter would have to be treated as an uncertain influence whose behavior must be estimated from data. The parameter θ is randomly initialized at the beginning of the policy gradient algorithm.

In addition to being differentiable, π_θ has two other noteworthy attributes. First, under the optimal choice of θ, the controller π_θ performs no worse than π_0. This is because π_0 is a feasible solution to (2.9), so the optimal solution to (2.9) can do no worse. Second, unlike projection-based methods [44], the structure of π_θ facilitates exploration of the interior of the safe action set. This is because smooth functions such as the sigmoid or hyperbolic tangent can be used as activation functions in the output layer of ψ_θ to constrain its output to $\mathbb{B}_\infty$. By tuning the steepness of the activation function, it is possible to bias the output of ψ_θ towards or away from the boundary of $\mathbb{B}_\infty$.

2.3 Simulations

2.3.1 Power system model

The main application considered in this chapter is frequency control in power systems. We consider a system with N synchronous electric generators. The standard linearized

swing equation at generator i is:

$$\dot{\delta}_i = \omega_i \tag{2.14a}$$

$$M_i\dot{\omega}_i = -D_i\omega_i - \sum_{j=1}^{N} K_{ij}(\delta_i - \delta_j) + \sum_{k=1}^{m} b_{ik}u_k - \sum_{l=1}^{p} e_{il}d_l, \tag{2.14b}$$

where δ_i is the rotor angle, ω_i is the frequency deviation, and M_i and D_i are the inertia and damping coefficients of generator i. The coefficients K_{ij}, b_{ik}, and e_{il} are based on generator and transmission line parameters taken from a modified IEEE 9-bus test case, and are computed by solving the DC power flow equations. Thus, the size of the coefficient measures the influence of each element on the dynamics of generator i. The quantity u_k represents controller k, an IBR such as a battery energy storage system or wind turbine [52,98], where the active power injections can be controlled in response to a change in system frequency. The feasible control set $\mathcal{U} \subset \mathbb{R}^m$ represents limits on power output for each of the m IBRs.

The disturbance d_l captures the uncertainties both in load and in uncontrolled renewable resources. It is also possible to use d to account for parameteric uncertainties, linearization error associated with the linearized swing equation dynamics, or error associated with the DC power flow approximation, by adding virtual disturbances at every bus in the system [36, 67]. The disturbance set $\mathcal{D} \subset \mathbb{R}^p$ is conservatively estimated from the capacity of the p uncontrolled elements [22].

Discretizing the continuous-time system in (2.14) and assembling block components gives a system in the form of (2.1). Let δ and $\omega \in \mathbb{R}^N$ be vectors representing the rotor angles and frequency deviations of all generators in the system, and let the system state be represented by $x = \begin{bmatrix}\delta & \omega\end{bmatrix}^T \in \mathbb{R}^n$ where $n = 2N$. Using time step τ,

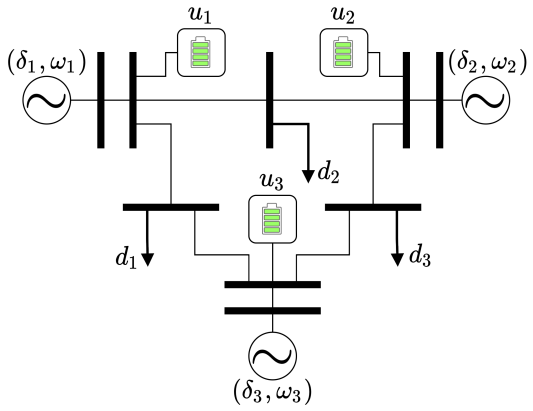

Figure 2.4: Illustration of 9-bus power system model.

the discrete-time system matrices are given by

$$A = \begin{bmatrix} I & \tau I \\ -\tau M^{-1}K & I - \tau M^{-1}D \end{bmatrix},$$
$$B = \begin{bmatrix} 0 \\ M^{-1}\hat{B} \end{bmatrix}, E = \begin{bmatrix} 0 \\ M^{-1}\hat{E} \end{bmatrix}$$

where $[M]_{ii} = M_i$, $[D]_{ii} = D_i$, $[K]_{ij} = K_{ij}$, $[\hat{B}]_{ik} = b_{ik}$, and $[\hat{E}]_{il} = e_{il}$.

We simulate the proposed policy network architecture on a 9-bus power system consisting of three synchronous electric generators, three controllable IBRs, and three uncontrolled loads. The time step for discretization is 0.05 seconds. The load is modeled as autoregressive noise defined by

$$d_{t+1} = \alpha d_t + (1 - \alpha)\hat{d} \tag{2.15}$$

where $\hat{d} \in \mathbb{R}^p$ is randomly generated from a uniform distribution over $\mathcal{D}$, and $\alpha \in (0, 1)$. The system is illustrated in Figure 2.4.

2.3.2 RL algorithm

To train the policy network, we used the Deep Deterministic Policy Gradient (DDPG) algorithm [60], an algorithm well-suited for RL in continuous control tasks. DDPG is

an actor-critic algorithm, in which the "actor" or policy chooses actions based on the state of the system, and the "critic" predicts the value of state-action pairs in order to estimate the gradient of the cost function (2.9a) with respect to θ (the "policy gradient"). In our simulations, the cost was given by

$$J(x_t, u_t) = x^T Q x + u^T R u \tag{2.16}$$

where $Q = \textbf{block diag}\{1000 I_N, 10 I_N\}$, $R = 5 I_m$, and I_N is the identity matrix in $\mathbb{R}^{N \times N}$. The actor was given by (2.13). The function ψ_θ was parameterized by a neural network with two hidden layers of 256 nodes each, with ReLU activation functions in the hidden layers. We use sigmoid functions in the last layer to limit the the outputs to be within $[-1, 1]$. The critic, or value network, had the same hidden layers as ψ_θ but a linear output layer. We trained the system for 200 episodes of 100 time steps each.

2.3.3 Benchmark comparisons

To demonstrate the advantages of the proposed policy architecture, we compare against two benchmarks. The first is the linear controller Kx, chosen to maximize the size of the associated RCI. Using the algorithm in [61], we choose $(\mathcal{S}, K)$ by solving the optimization problem

$$\max_{\mathcal{S} \in \mathcal{S}, K \in \mathbb{R}^{n \times m}} \text{vol}(\mathcal{S}) \tag{2.17a}$$

$$\text{s.t. Invariance: } (A + BK)\mathcal{S} \oplus E\mathcal{D} \subseteq \mathcal{S} \tag{2.17b}$$

$$\text{Safety: } \mathcal{S} \subseteq \mathcal{X} \tag{2.17c}$$

$$\text{Control bounds: } K\mathcal{S} \subseteq \mathcal{U} \tag{2.17d}$$

where $\oplus$ denotes Minkowski set addition and $\mathcal{S}$ is a class of polytopes described by (2.5). Figure 2.5 displays the accumulated cost during a number of test trajectories, showing that π_θ is a more cost-effective controller than Kx when the same $\mathcal{S}$ is used

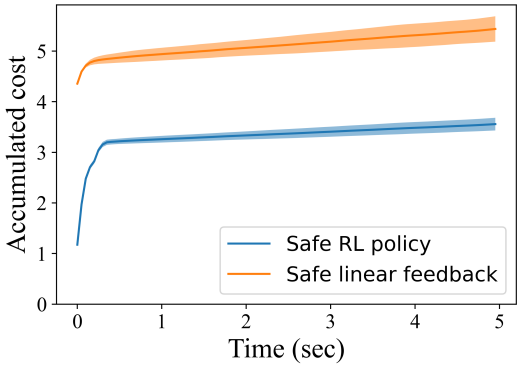

Figure 2.5: Accumulated cost over several test trajectories with a fixed initial condition and randomly generated disturbance sequences, showing that the RCI policy network achieves better performance than the safe linear feedback.

for each policy. This makes sense, since the nonlinear policy is afforded additional flexibility in balancing performance and robustness. Since π_θ and Kx are both policies in Π, the learned policy has the same safety guarantees as the linear policy.

The second benchmark is a policy network that is trained using DDPG augmented with a soft penalty on constraint violations, in order to incentivize remaining in $\mathcal{X}$. The policy network for this benchmark consists of two 256-node hidden layers with ReLU activation, and hyperbolic tangent activation functions in the output layer that clamp the output to the box-shaped set $\mathcal{U}$. The soft penalty is the total constraint violation, calculated as

$$\lambda \| \max\{V_x x_t - \bar{x}, 0\} - \min\{V_x x_t + \bar{x}, 0\} \|_1$$

where the max and min are taken elementwise, and $\lambda > 0$.

In Fig. 2.6, we plot an example of the maximum angle deviation during training. We place a hard constraint of 0.1 radians on this angle deviation. For the policy given by (2.13), the trajectory always stays within this bound, by the design of the the controller. For a policy trained with a soft penalty, trajectories initially exit the safe set. With enough training, the trajectories eventually satisfy the state constraints.

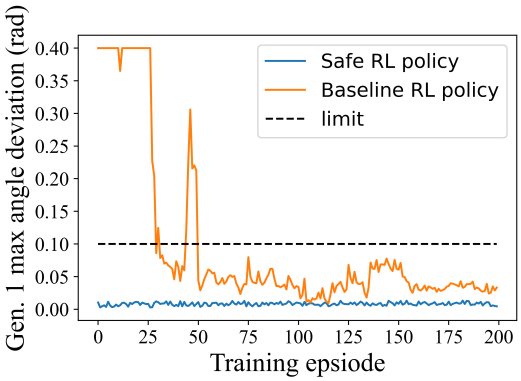

Figure 2.6: Maximum angle deviation per training episode for the safe policy network (blue) and the baseline policy network with soft penalty (orange). The safe policy network guarantees safety during training, while soft penalties eventually drive the baseline policy towards constraint satisfaction.

Figure 2.7 shows that safety in training does not imply safety in testing. The policy network trained using a soft penalty can still result in constraint violations, whereas the safe policy network does not. In some sense, this is not unexpected. Only a limited number of disturbances can be seen during training, and because of the nonlinearity of the neural network-based policy, it is difficult to provide guarantees from the cost alone. In addition, picking the right soft penalty parameter is nontrivial. If the penalty λ is too low, constraint satisfaction will not be incentivized, and if λ is too high, convergence issues may arise [101]. In our experiments, we tuned λ by hand to strike the middle ground, but even automatic, dynamic tuning of λ during training is not guaranteed to prevent constraint violations in all cases [101].

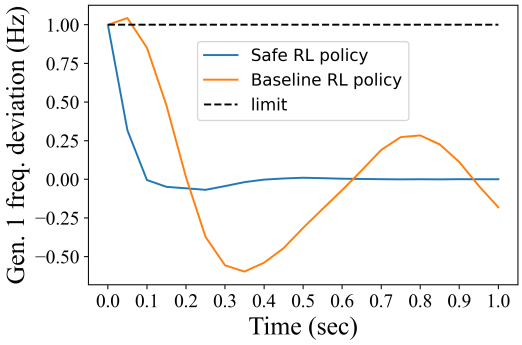

Figure 2.7: Even though soft penalties succeed in driving policies to be safe during training, they do not necessarily provide safety during testing. In this example, a policy network that was safe during training (Fig. 2.6) still exhibits constraint violations during testing. In contrast, starting from the same initial conditions and subject to the same disturbance sequence, the proposed safe policy network guarantees constraint satisfaction.

Chapter 3

SAFE AND EFFICIENT MODEL PREDICTIVE CONTROL USING NEURAL NETWORKS: AN INTERIOR POINT APPROACH[1]

Model predictive control (MPC) [82] is a powerful technique for controlling systems that are subject to state and input constraints, such as agricultural [31], automotive [47], and energy systems [2]. However, many applications require fast decision-making which may preclude the possibility of repeatedly solving an optimization problem online [6].

A popular approach for accelerating MPC is to move as much computation offline as possible [6, 103]. These techniques, known as explicit MPC, involve precomputing the solution to the MPC problem over a range of parameters or initial conditions. Most of the research effort has focused on problems with linear dynamics and constraints, and linear or quadratic cost functions. In this case, the explicit MPC solution is a piecewise affine (PWA) function defined over a polyhedral partition of the state constraints. However, many of the applications of interest have cost functions that are not necessarily linear or quadratic, or even convex. Further, the memory required to store the partition and affine functions can be prohibitive even for modestly-sized problems.

In order to reduce the complexity of explicit MPC, the optimal offline solution can be approximated. Approximations generally fall into two categories: partition-based solutions [43,49,50] that generate piecewise control laws over coarser state space

[1]Adapted from D. Tabas and B. Zhang, "Safe and Efficient Model Predictive Control Using Neural Networks: An Interior Point Approach," 2022 IEEE 61st Conference on Decision and Control (CDC), Cancun, Mexico, 2022, pp. 1142-1147.

partitions, and learning-based solutions [5, 20, 32, 76] that use function approximation to compactly represent the optimal MPC policy. In this chapter, we focus on the latter with the key contribution of ensuring constraint satisfaction while exploring all feasible policies.

Constraint satisfaction is crucial in many engineering applications, and the ability of MPC to enforce constraints is a major factor in its popularity. However, it is not straightforward to guarantee that a learning-based solution will satisfy constraints. The main challenge arises from the fact that while neural networks can limit their outputs to be in simple regions, there is no obvious way of forcing complex constraint satisfaction at the output. In [5, 68], supervised and unsupervised learning were used to approximate the solution of MPCs, but did not provide any feasibility guarantees. By contrast, [20] trains an NN using a policy gradient approach, and guarantees feasibility by projecting the NN output into the feasible action set. However, this extra optimization step slows down the speed of online implementation, making it difficult to use in applications that require high-frequency solutions [109]. Supervised learning approaches that provide safety guarantees [32, 76] rely on a choice of MPC oracle that is not obvious when persistent disturbances are present.

In this chapter, we propose an NN architecture for approximating explicit solutions to finite-horizon MPC problems with linear dynamics, linear constraints, and arbitrary differentiable cost functions. The proposed architecture guarantees constraint satisfaction without relying on projections or MPC oracles. By exploring the *interior* of the feasible set, we demonstrate faster training and evaluation, and comparable closed-loop performance relative to other NN architectures.

The proposed approach has parallels in interior point methods for convex optimization [18]. Interior point methods first solve a *Phase I* problem to find a strictly feasible starting point. This solution is used to initialize the *Phase II* algorithm for optimizing the original problem. Our approach accelerates both phases. The Phase I solution is given by a simple function (e.g., affine map) and the Phase II problem is

solved using an NN architecture that can encode arbitrary polytopic constraints (Fig. 3.1).

The Phase II solution builds on a technique first proposed in [91], which uses a *gauge map* to establish equivalence between compact, convex sets. With respect to [91], the current work has three novel aspects. First, the reinforcement learning (RL) algorithm in [91] only uses information about the constraints, and does not use information about the cost function or dynamics. The resulting policy is safe, but can exhibit suboptimal performance. The MPC formulation in the current chapter gives rise to a training algorithm that can exploit knowledge about the system, improving performance. Second, the MPC formulation permits explicit consideration for future time steps. The RL formulation cannot optimize entire trajectories due to the presence of constraints. This inability to "look ahead" again limits the performance of the RL algorithm. Finally, the previous work required a Phase I that used a linear feedback to find a strictly feasible point. A linear feedback, however, may not exist for some problems. The current work proposes a more general class of Phase I solutions (piecewise affine), while providing a way to manage the complexity of the Phase I solution.

We demonstrate the effectiveness of the proposed technique on a 3-state test system, and compare to standard projection- and penalty-based approaches for learning with constraints. The results show that the proposed technique achieves Pareto efficiency in terms of closed-loop performance and online computation effort. All code is available at `github.com/dtabas/gauge_networks`.

Notation

The p-norm ball for $p \geq 1$ is $\mathbb{B}_p = \{z \mid \|z\|_p \leq 1\}$. A *polytope* $\mathcal{P} \subset \mathbb{R}^n := \{z \in \mathbb{R}^n \mid Fz \leq g\}$ is the (bounded) intersection of a finite number of halfspaces. Scaling of polytopes by a factor $\lambda > 0$ is defined as $\lambda\mathcal{P} = \{\lambda z \in \mathbb{R}^n \mid Fz \leq g\} = \{z \in \mathbb{R}^n \mid Fz \leq \lambda g\}$. Given a matrix F and a vector g, the ith row of F is denoted $F^{(i)T}$ and the ith

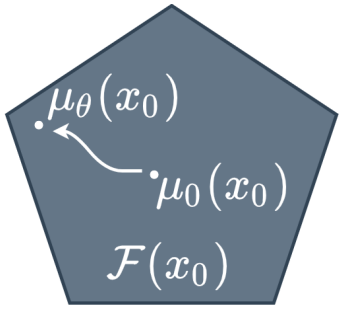

Figure 3.1: Illustration of the interior point approach to learning-based MPC. The set $\mathcal{F}(x_0)$ represents the MPC feasible set, while $\mu_0(x_0)$ and $\mu_\theta(x_0)$ are control input sequences representing solutions to the Phase I and Phase II problems, respectively. The neural network μ_θ moves the Phase I solution to a more optimal solution.

component of g is $g^{(i)}$. The interior of any set $\mathcal{Q}$ is denoted **int** $\mathcal{Q}$. The value of a variable y at a time interval t is denoted y_t. A state or control trajectory of length τ is written as the vector $\mathbf{x} = \left[x_1^T, \ldots, x_\tau^T\right]^T \in \mathbb{R}^{n\tau}$ or $\mathbf{u} = \left[u_0^T, \ldots, u_{\tau-1}^T\right]^T \in \mathbb{R}^{m\tau}$. The column vector of all ones is $\mathbf{1}$. The symbol $\circ$ denotes function composition.

3.1 Problem Formulation

In this chapter, we consider the problem of regulating discrete-time dynamical systems of the form

$$x_{t+1} = Ax_t + Bu_t + d_t \tag{3.1}$$

where $x_t \in \mathbb{R}^n$ is the system state at time t, $u_t \in \mathbb{R}^m$ is the control input, and $d_t \in \mathbb{R}^n$ is an uncertain input that captures exogenous disturbances and/or linearization error (if the true system dynamics are nonlinear) [17]. We assume the pair (A, B) is stabilizable. The input constraints (actuation limits) are $\mathcal{U} = \{u \in \mathbb{R}^m \mid F_u u \leq g_u\}$ while the state constraints arising from safety-critical engineering considerations are $\mathcal{X} = \{x \in \mathbb{R}^n \mid F_x x \leq g_x\}$.

We consider the problem of operating the system (3.1) using finite-horizon model predictive control. The goal is to choose, given initial condition $x_0 \in \mathcal{X}$, a sequence of inputs $\mathbf{u}$ of length τ that minimizes the cost of operating the system while respecting the operational constraints.

However, since the disturbances d_t are unknown ahead of time, the designer must carefully consider how to achieve both optimality and constraint satisfaction. Robust MPC literature contains many ways to handle the presence of disturbances in both the cost and constraints [9]. For example, the *certainty-equivalent* approach [6] considers only the nominal system trajectory, while the *min-max* approach [43] considers the worst-case disturbance. Interpolating between these two extremes, the *tube-based* approach [54] considers the cost of a nominal trajectory while guaranteeing that the true trajectory satisfies constraints. A *stochastic* point of view in [38] considers the disturbance as a random variable and minimizes the expected cost while providing probabilistic guarantees for constraint satisfaction.

In most robust MPC formulations, the set of possible disturbances is modeled as either a finite set, a bounded set, or a probability distribution [86]. In this chapter, we assume the disturbances lie in a closed and bounded set $\mathcal{D} := \{d \in \mathbb{R}^n \mid F_d d \leq g_d\}$. In order to ensure constraint satisfaction, we operate the system within a *robust control invariant set* (RCI) $\mathcal{S} \subseteq \mathcal{X}$, defined as a set of initial conditions for which there exists a feedback policy in $\mathcal{U}$ keeping all system trajectories in $\mathcal{S}$, under any disturbance sequence in $\mathcal{D}$ [13]. In our simulations, we used approximately-maximal RCIs computed with the semidefinite program from [61].

With $\mathcal{S} := \{x \in \mathbb{R}^n \mid F_s x \leq g_s\}$, we define the *target set* $\mathcal{T}$ as $\{x \in \mathbb{R}^n \mid x + d \in \mathcal{S},\ \forall\ d \in \mathcal{D}\} = \{x \in \mathbb{R}^n \mid F_s x \leq \tilde{g}_s\}$ where for each row i, $\tilde{g}_s^{(i)} = g_s^{(i)} - \max_{d \in \mathcal{D}} F_s^{(i)T} d$ [13]. Any policy that maps $\mathcal{S}$ to $\mathcal{T}$ under the nominal dynamics will map $\mathcal{S}$ to itself under the true dynamics, rendering $\mathcal{S}$ robustly invariant. By constraining the nominal state to the target set, robust constraint satisfaction is guaranteed for the first time step. Since $\mathcal{S}$ is RCI, this is sufficient for keeping closed-loop trajectories inside $\mathcal{S}$.

Under this formulation, the MPC problem is posed as follows, given initial state x_0:

$$\min_{\mathbf{u}} \sum_{k=0}^{\tau-1} l(x_k, u_k) + l_F(x_\tau) \tag{3.2a}$$

$$\text{subject to } \forall\ k\text{: } x_{k+1} = Ax_k + Bu_k \tag{3.2b}$$

$$x_{k+1} \in \mathcal{T} \tag{3.2c}$$

$$u_k \in \mathcal{U} \tag{3.2d}$$

where l and l_F are stage and terminal costs that are differentiable but possibly non-linear or even non-convex. Although (3.2) differs from the standard tube-based approach, the techniques introduced in this chapter can be applied to a variety of MPC formulations.

In this chapter, we seek to derive a safe feedback policy $\pi_\theta : \mathbb{R}^n \to \mathbb{R}^m$ that approximates the explicit solution to (3.2) by first approximating the optimal control sequence with a function $\mu_\theta : \mathbb{R}^n \to \mathbb{R}^{m\tau}$ and then implementing the first action of the sequence in the closed loop. In practice, any MPC policy implemented in closed loop must be stabilizing and recursively feasible. Recursive feasibility is the property that closed-loop trajectories generated by the MPC controller will not lead to states in which the MPC problem is infeasible. This property is guaranteed when $\mathcal{S}$ is RCI [13]. If recursive feasibility is not guaranteed, then a backup controller must be developed or a control sequence that is feasible for the most immediate time steps can be used. There is suggestion in the literature that the latter approach performs quite well in practice [96], but the theoretical aspects remain open. In terms of stability, recursive feasibility guarantees that trajectories will remain within a bounded set. Since this work focuses on constraint satisfaction, we do not consider stricter notions of stability.

3.2 Phase I: Finding a Feasible Point

The feasible set of (3.2) is a polytope $\mathcal{F}(x_0) \subseteq \mathbb{R}^{m\tau}$, defined by the following inequalities in $\mathbf{u}$:

$$H_s(M_0 x_0 + M_u \mathbf{u}) \leq \tilde{h}_s, \tag{3.3a}$$

$$H_u \mathbf{u} \leq h_u \tag{3.3b}$$

where $H_s, H_u, M_0, M_u, \tilde{h}_s$, and h_u are block matrices and vectors derived from the system dynamics and constraints. In this chapter, we assume that $\mathcal{F}(x_0)$ has nonempty interior for all $x_0 \in \mathcal{S}$. Since the state constraints $\mathcal{S}$ form an RCI, $\mathcal{F}(x_0)$ is already guaranteed to be nonempty, and the assumption of nonempty interior is only marginally more restrictive.

The gauge map technique introduced in [91] provides a way to constrain the outputs of a neural network $\mu_\theta : \mathbb{R}^n \to \mathbb{R}^{m\tau}$ to $\mathcal{F}(x_0)$ without a projection or penalty function, but $\mathcal{F}(x_0)$ must contain the origin in its interior. If this is not the case, then we must temporarily "shift" $\mathcal{F}(x_0)$ by subtracting any one of its interior points. In this section, we discuss several ways to reduce the complexity of finding an interior point.

We begin by considering the feasibility problem for the one-step safe action set defined as $\mathcal{V}(x_0) = \{u \in \mathbb{R}^m \mid u \in \mathcal{U}, Ax_0 + Bu \in \mathcal{T}\}$, which is guaranteed to have an interior point by the assumption on $\mathcal{F}(x_0)$. One way to find an interior point of $\mathcal{V}(x_0)$ is to minimize the maximum constraint violation:

$$\min_{u,s} s \tag{3.4a}$$

$$\text{subject to: } F_s(Ax_0 + Bu) \leq \tilde{g}_s + s\mathbf{1} \tag{3.4b}$$

$$F_u u \leq g_u + s\mathbf{1} \tag{3.4c}$$

which has an optimal cost $s^* \leq 0$ if $\mathcal{V}(x_0)$ is nonempty, and $s^* < 0$ if $\mathcal{V}(x_0)$ has nonempty interior [18]. To avoid solving a linear program online during closed-loop

implementation, the solution to (3.4) can be stored as a piecewise affine (PWA) function $\pi_0(x_0) : \mathbb{R}^n \to \mathbb{R}^m$ [50]. Although solutions to multiparametric LPs can be demanding on computer memory, we take advantage of the fact that feasibility problems have low accuracy requirements: any suboptimal solution to (3.4) that achieves a cost $s < 0$ for all $x_0 \in \mathcal{S}$ is acceptable.

Definition 3.1. A function $\pi_0 : \mathbb{R}^n \to \mathbb{R}^m$ is said to solve (3.4) if, for all $x_0 \in \mathcal{S}$, the optimal cost of (3.4) is negative when the decision variable u is fixed at $\pi_0(x_0)$.

Existing techniques for approximate multiparametric linear programming [39], especially those that generate continuous solutions [89], can be used to reduce the memory requirements of offline solutions to (3.4).

To show just how far one can go with reducing complexity, we will construct an affine (rather than PWA) function that solves (3.4), for the system studied in Section 3.4. Let $\pi_0(x_0) = Wx_0 + w$. If $W \in \mathbb{R}^{m \times n}$ and $w \in \mathbb{R}^m$ satisfy

$$F_x(Ax_0 + B(Wx_0 + w)) < \tilde{g}_x \tag{3.5a}$$

$$F_u(Wx_0 + w) < g_u \tag{3.5b}$$

for all $x_0 \in \mathcal{S}$, then $\pi_0(x_0) = Wx_0 + w$ solves (3.4). The following optimization problem can be solved to find W and w or certify that none exists. Let $\mathcal{Y}(s) = \{x_0 \in \mathbb{R}^n \mid F_s(Ax_0 + B(Wx_0 + w)) \leq \tilde{g}_s + s\mathbf{1}, F_u(Wx_0 + w) \leq g_u + s\mathbf{1}\}$. If the optimal cost of

$$\min_{W,w,s} s \text{ subject to } \mathcal{S} \subseteq \mathcal{Y}(s) \tag{3.6}$$

is negative, then (3.5) holds for all $x_0 \in \mathcal{S}$, thus π_0 solves (3.4). This happens to be the case for the example in Section 3.4, taken from [103]. The constraint in (3.6) is a polytope containment constraint in halfspace representation, thus (3.6) can be solved as a linear program [85].

Now consider the feasibility problem for $\mathcal{F}(x_0)$, which is obtained by replacing (3.4b) and (3.4c) with (3.3a) and (3.3b), and changing the optimization variable from

$u \in \mathbb{R}^m$ to $\mathbf{u} \in \mathbb{R}^{m\tau}$. One would naturally expect the complexity of the PWA solution to this feasibility problem to increase rapidly with the time horizon τ, as more decision variables and constraints are added. However, the next proposition shows that the cardinality of the stored partition can be made constant in τ.

Proposition 3.1 (Phase I solution)**.** If π_0 solves (3.4), then the vector $\mu_0(x_0) := \left[\pi_0(x_0)^T, \ldots, \pi_0(x_{\tau-1})^T\right]^T$, where $x_{k+1} = Ax_k + B\pi_0(x_k)$, is an interior point of $\mathcal{F}(x_0)$ for any $x_0 \in \mathcal{S}$.

Proof. If π_0 solves (3.4), then $\pi_0(x) \in \textbf{int } \mathcal{V}(x)$ for all $x \in \mathcal{S}$. Applying the definition of $\mathcal{V}$ in an inductive argument, it is straightforward to show that the state trajectory associated with $\mu_0(x_0)$ is entirely contained in $\mathcal{S}$. Fix any such trajectory $\{x_1, \ldots, x_\tau\} \subset \mathcal{S}$ originating from $x_0 \in \mathcal{S}$ under policy π_0. For any $k \in \{1, \ldots, \tau\}$, $x_k \in \mathcal{S}$ implies $\pi_0(x_k) \in \textbf{int } \mathcal{V}(x_k)$, which implies $\pi_0(x_k) \in \textbf{int } \mathcal{U}$ and $Ax_k + B\pi_0(x_k) \in \textbf{int } \mathcal{T}$. Since this holds for all k, the constraints defining $\mathcal{F}(x_0)$ hold strictly at $\mu_0(x_0)$. □

In our simulations on the example from [103], (3.6) was feasible with negative optimal cost, meaning that a polyhedral partition of the state space was not needed (see Section 3.4). This indicates that the minimum number of regions in a polyhedral state space partition associated with a PWA solution to (3.4) is in general very small relative to the number of regions in an explicit solution to (3.2).

3.3 Phase II: Optimizing Performance

In this section, we construct a class of policies from $x_0 \in \mathcal{S}$ to $\mathcal{F}(x_0)$, that can be trained using standard machine learning packages. Although it is difficult to constrain the output of a neural network to an arbitrary polytope such as $\mathcal{F}(x_0)$, it is easy to constrain the output to the hypercube $\mathbb{B}_\infty$ by applying a clamping function elementwise in the output layer. We apply a mapping between polytopes that is closed-form, differentiable, and bijective. This mapping establishes an equivalence

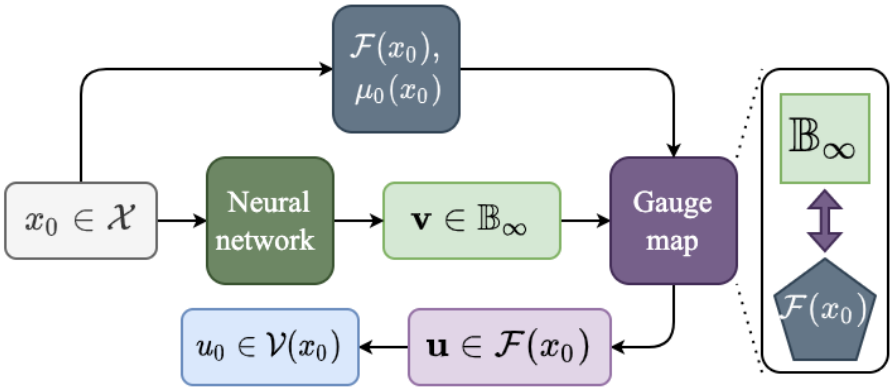

Figure 3.2: The proposed control policy uses a neural network combined with the Phase I solution and a *gauge map* to constrain the decision $\mathbf{u}$ to the MPC feasible set $\mathcal{F}(x_0)$. The first action from the sequence $\mathbf{u}$ is extracted and implemented. On the right, the action of the gauge map is illustrated.

between $\mathbb{B}_\infty$ and $\mathcal{F}(x_0)$, allowing one to constrain the outputs of the policy to $\mathcal{F}(x_0)$. The mapping from $\mathbb{B}_\infty$ to $\mathcal{F}(x_0)$ is called the *gauge map*. The concept is illustrated in Figure 3.2.

We begin constructing the gauge map by introducing some preliminary concepts. A *C-set* is a convex, compact set that contains the origin as an interior point. The *gauge function* with respect to C-set $\mathcal{P} \subset \mathbb{R}^n$, denoted $\gamma_\mathcal{P} : \mathbb{R}^n \to \mathbb{R}_+$, is the function whose sublevel sets are scaled versions of $\mathcal{P}$. Specifically, the gauge of a vector v with respect to $\mathcal{P}$ is given by $\gamma_\mathcal{P}(v) = \inf\{\lambda \geq 0 \mid v \in \lambda\mathcal{P}\}$. If $\mathcal{P}$ is a polytopic C-set given by $\{v \in \mathbb{R}^k \mid Fv \leq g\}$, then $\gamma_\mathcal{P}$ is the pointwise maximum over a finite set of affine functions [91]:

$$\gamma_\mathcal{P}(v) = \max_i \frac{F^{(i)T} v}{g^{(i)}}. \tag{3.7}$$

Given two C-sets $\mathcal{P}$ and $\mathcal{Q}$, the *gauge map* $G : \mathcal{P} \to \mathcal{Q}$ is

$$G(v \mid \mathcal{P}, \mathcal{Q}) = \frac{\gamma_\mathcal{P}(v)}{\gamma_\mathcal{Q}(v)} \cdot v. \tag{3.8}$$

This function maps level sets of $\gamma_{\mathcal{P}}$ to level sets of $\gamma_{\mathcal{Q}}$.

Proposition 3.2. Given two polytopic C-sets $\mathcal{P}$ and $\mathcal{Q}$, the gauge map $G : \mathcal{P} \to \mathcal{Q}$ is subdifferentiable and bijective. Further, given a function π_0 from Proposition 3.1, the set $\tilde{\mathcal{F}}(x) := [\mathcal{F}(x) - \pi_0(x)]$ is a C-set for all $x \in \mathcal{S}$.

Proof. The properties of subdifferentiability and bijectivity are established in Chapter 2. For the C-set property, fix $x \in \mathcal{S}$. Since $\mathcal{S}$, $\mathcal{U}$, and $\mathcal{D}$ are convex and compact, so is $\mathcal{F}(x)$. Since $\mu_0(x)$ is an interior point of $\mathcal{F}(x)$, the set $\tilde{\mathcal{F}}(x)$ contains the origin as an interior point and is therefore a C-set. □

We now use the gauge map in conjunction with the Phase I solution to construct a neural network whose output is confined to $\mathcal{F}(x_0)$. Let $\psi_\theta : \mathcal{S} \to \mathbb{B}_\infty$ be a neural network parameterized by θ. A safe policy is constructed by composing the gauge map $G : \mathbb{B}_\infty \to \tilde{\mathcal{F}}(x_0)$ with ψ_θ, then adding $\mu_0(x_0)$ to map the solution into $\mathcal{F}(x_0)$:

$$\mu_\theta(x_0) = G(\cdot \mid \mathbb{B}_\infty, \tilde{\mathcal{F}}(x_0)) \circ \psi_\theta(x_0) + \mu_0(x_0). \tag{3.9}$$

Computing the gauge map online simply requires evaluating $H_s M_0 x_0$ from (3.3a) as well as the operations in (3.7).

The function μ_θ has several important properties for approximating the optimal solution to (3.2). First, it leverages the universal function approximation properties of neural networks [46] along with the bijectivity of the gauge map (Proposition 3.2) to explore all interior points of $\mathcal{F}(x_0)$. This is an advantage over projection-based methods [20] which may be biased towards the boundary of $\mathcal{F}(x_0)$ when the optimal solution may lie on the interior. Second, μ_θ is evaluated in closed form, and its outputs are constrained to $\mathcal{F}(x_0)$ without the use of an optimization layer [68] that may have high computational overhead. Finally, the subdifferentiability of the gauge map (Proposition 3.2) enables selection of parameter θ using standard automatic differentiation techniques.

Optimizing the parameter θ

Similar to the approach taken in [5], we optimize θ by sampling $x \in \mathcal{S}$ and applying stochastic gradient descent. At each iteration, a new batch of initial conditions $\{x_0^j\}_{j=1}^M$ is sampled from $\mathcal{S}$ and the loss is computed as

$$J(\theta) = \frac{1}{M} \sum_{j=1}^{M} \sum_{k=0}^{\tau-1} l(x_k^j, u_k^j) + l_F(x_\tau^j) \tag{3.10}$$

with the control sequences $\mathbf{u}^j$ given by $\mu_\theta(x_0^j)$ and state trajectories $\mathbf{x}^j$ generated according to the nominal dynamics. The parameters θ are updated in the direction of $\nabla_\theta J$, which is easily computed using automatic differentiation [8].

3.4 Simulations

3.4.1 Test systems

We simulate the proposed policy using a modified example from [103] with $n = 3$, $m = 2$, and $\tau = 5$. The system matrices, constraints, costs, and Phase I solution (found using (3.6)) are given below:

$$A = \begin{bmatrix} -.5 & .3 & -1 \\ .2 & -.5 & .6 \\ 1 & .6 & -.6 \end{bmatrix}, \; B = \begin{bmatrix} -.601 & -.890 \\ .955 & -.715 \\ .246 & -.184 \end{bmatrix}, \tag{3.11}$$

$$\|x\|_\infty \leq 5, \; \|u\|_\infty \leq 1, \; \|d\|_\infty \leq 0.1, \tag{3.12}$$

$$l(x,u) = \|x\|_2^2 + c_1\|u\|_2^2, \; l_F(x) = c_2\|x\|_2^2 \tag{3.13}$$

$$W = \begin{bmatrix} 0.116 & 0.210 & -0.370 \\ -0.320 & -0.104 & -0.122 \end{bmatrix}, w = \begin{bmatrix} -0.157 \\ -0.0533 \end{bmatrix}$$

where c_1 and c_2 are positive constants. Although quadratic costs are used in the simulations, the proposed method can work with any differentiable cost.

We evaluate the performance of a given policy in both open- and closed-loop experiments. In the open-loop experiments, we evaluate the MPC cost (3.2a) and

compare it to the optimal cost. The fraction suboptimality is

$$\delta = \frac{c_{nn} - c_{mpc}}{c_{mpc}} \tag{3.14}$$

where c_{nn} is the average cost (3.2a) incurred by the control sequence μ_θ on a validation set $\{x_0^j\}_{j=1}^{N_{val}} \subset \mathcal{S}$ and c_{mpc} is the optimal cost.

In the closed-loop experiments, we evaluate the performance of a policy $\pi_\theta(x_t) : \mathbb{R}^n \to \mathbb{R}^m, t \geq 0$ which is derived from $\mu_\theta(x_t)$ by taking the first action in the sequence. We simulate (3.1) for $T \gg \tau$ time steps. The trajectory cost in the closed-loop experiments is computed as $\sum_{t=0}^{T-1} l(x_t, u_t) + l_F(x_T)$ and the disturbance is modeled as an autoregressive sequence [90], $d_{t+1} = \alpha d_t + (1-\alpha)\hat{d}$ where $\alpha \in (0, 1)$ and $\hat{d}$ is drawn uniformly over $\mathcal{D}$.

3.4.2 Benchmarks

We compare the proposed method to two of the most common approaches for learning a solution to (3.2). The first benchmark is a penalty-based approach [35] which enforces the constraints (3.2c) and (3.2d) by augmenting the cost (3.10) with a linear penalty term on constraint violations given by $\beta \cdot \max\{0, F_x x_t - \tilde{g}_x\}$ where the max is evaluated elementwise and $\beta > 0$. Since the penalty-based approach does not encode state constraints in the policy, the policy is constrained to the Cartesian product $\mathcal{U}^\tau = \prod_{k=0}^{\tau-1} \mathcal{U}$ using scaled tanh functions elementwise.

The second benchmark is a projection-based approach [20] which constrains the policy to the set $\mathcal{F}(x_0)$ by solving a convex quadratic program in the output layer of a neural network [4]. The optimization layer $\mathbf{v} \to \mathbf{u}$ returns

$$\arg\min_{\mathbf{u}} \|\mathbf{v} - \mathbf{u}\|_2^2 \text{ subject to } \mathbf{u} \in \mathcal{F}(x_0).$$

Another class of approaches to learning-based MPC seeks to learn the optimal solution to (3.2) using regression [32, 68, 76]. Specifically, data-label pairs (x_0, u_0^*) are generated by sampling x_0 from $\mathcal{S}$, solving (3.2) for each sample, and extracting u_0^*

Table 3.1: Hyperparameters for the three neural networks.

Type	Width	LR	M
Gauge	859	4.7×10^{-4}	1655
Penalty	318	8.7×10^{-4}	133
Projection	956	9.0×10^{-5}	813

from the optimal solution $\mathbf{u}^*$. Then, a neural network or other function approximator is trained to learn the relationship between x_0 and u_0^*. Performance and constraint satisfaction are handled e.g. by bounding the approximation error with respect to the MPC oracle. We do not compare against this type of approach since it requires a large number of trained samples, making it difficult to compare with our and the other unsupervised examples.

3.4.3 Neural network design

The neural networks were designed with n inputs, $m\tau$ outputs, and two hidden layers with rectified linear unit (ReLU) activation functions. The width of the networks was chosen during hyperparameter tuning. In particular, we performed 30 iterations of random search over the width of the network (number of neurons per hidden layer) $\in \{64, \dots, 1024\}$, the batch size (number of initial conditions, M) $\in \{100, \dots, 3000\}$ and the learning rate (LR, the step size for gradient descent) $\in [10^{-5}, 10^{-3}]$. For each set of hyperparameters under consideration, we computed the validation score using (3.14) with $N_{val} = 100$. The hyperparameters after tuning are reported in Table 3.1.

3.4.4 Simulation results

Here we compare our proposed approach (Gauge NN), the penalty-based approach (Penalty NN), the projection-based approach (Projection NN) and the "ground truth"

Table 3.2: Open-loop test results.

Type	δ (3.14)	Solve time (sec)
Gauge	0.007	.0015
Projection	0.010	.024

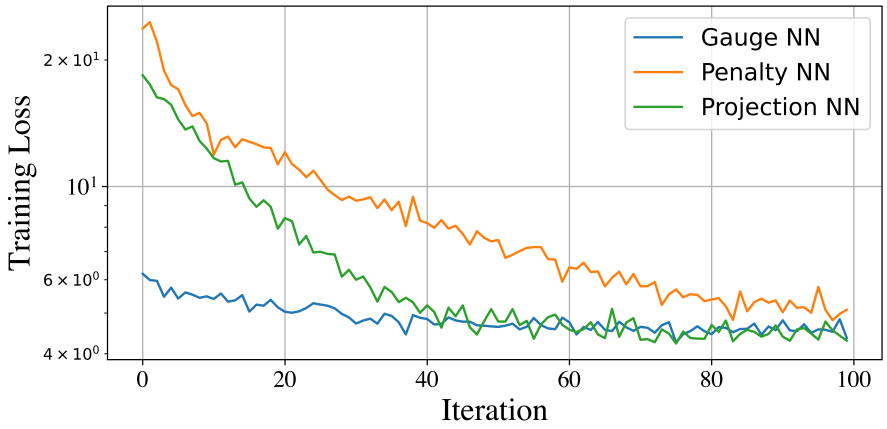

Figure 3.3: Training trajectories for the three types of neural netwokrs. Our proposed Gauge-based approach achieves lower cost at a much faster rate.

obtained by solving (3.2) online in `cvxpy`. The results of the open-loop experiments are shown in Table 3.2, with performance computed relative to the optimal MPC solution using (3.14) with $N_{val} = 100$ trials. The proposed Gauge NN achieves lower cost compared to the projection-based method, and has a much lower computational complexity (solve time is only 6% of projection). Table 3.2 only compares the NNs with safety guarantees because constraint violations are not accounted for in (3.14).

Figure 3.3 shows the training curves for each type of network. The lower training cost achieved by the Gauge NN illustrates that it can be more efficient to explore the interior of the feasible set than the boundary. Since the MPC cost in the simulations is strictly convex, solutions with lower cost are closer to the optimal solution.

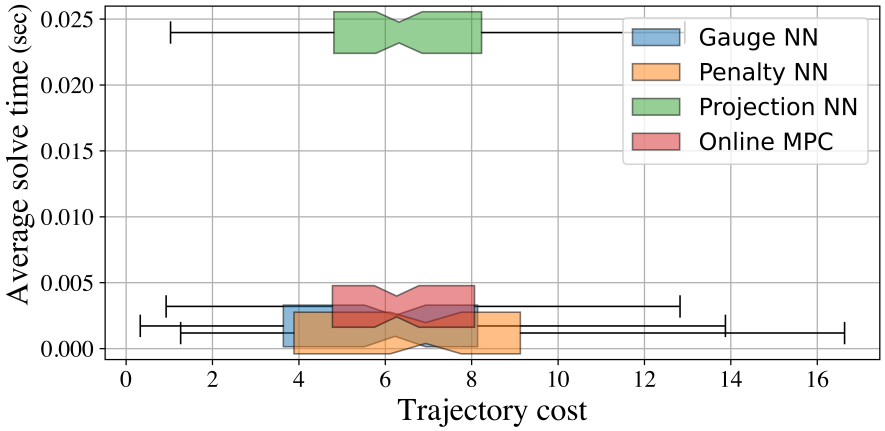

Figure 3.4: Solve time vs. trajectory cost for the networks under consideration applied to the 3-state system. The Gauge NN is Pareto-efficient in terms of cost and computation time compared to the other techniques with safety guarantees (Online MPC and Projection NN).

Figure 3.4 compares the policies in terms of computation time and test performance. The box-and-whisker plots indicate the range of performance over 100 test trajectories of length $T = 50$, while the vertical position of each box indicates the average time to compute a control action. Of the policies with safety guarantees (Gauge NN, Projection NN, and online MPC), the Gauge NN achieves Pareto efficiency in terms of average solve time and median trajectory cost. Our intuition behind the high performance of the neural networks is that (3.2) is a heuristic and the unsupervised learning approach can lead to better closed-loop policies.

Chapter 4

INTERPRETING PRIMAL-DUAL ALGORITHMS FOR CONSTRAINED MULTIAGENT REINFORCEMENT LEARNING[1]

As reinforcement learning (RL) algorithms progress from virtual to cyber-physical applications, it will be necessary to address the challenges of safety, especially when systems are controlled by multiple agents. Examples of multiagent safety-critical systems include power grids [28], building energy management (BEM) systems [11], autonomous vehicle navigation [110], and drone swarms [23]. In each of these applications, agents must learn to operate in a complicated environment while satisfying various local and system-wide constraints. Such constraints, derived from domain-specific knowledge, are designed to prevent damage to equipment, humans, or infrastructure or to preclude failure to complete some task or objective.

Constrained multiagent reinforcement learning (C-MARL) poses challenges beyond the single-agent constrained reinforcement learning (C-RL) problem because the interactions between agents can influence both the satisfaction of constraints and the convergence of policies. The potential scale of C-MARL problems eliminates the possibility of directly using common model-based methods for C-RL, such as in [20,66,92]. The main strategy for tackling C-MARL problems found in the literature is the Lagrangian or primal-dual method (see, e.g. [56, 59, 64, 77] and the references therein). Our aim is to understand some potential drawbacks of this approach and some ways these drawbacks can be mitigated.

---

[1]Adapted from Tabas, D., Zamzam, A.S. and Zhang, B. (2023). Interpreting Primal-Dual Algorithms for Constrained Multiagent Reinforcement Learning. *Proceedings of The 5th Annual Learning for Dynamics and Control Conference*, in *Proceedings of Machine Learning Research* 211:1205-1217.

In the primal-dual approach to C-MARL, each agent receives a reward signal that is augmented with a penalty term designed to incentivize constraint satisfaction. The magnitude of the penalty term is tuned to steer policies away from constraint violations while not unnecessarily overshadowing the original reward. Although this approach has been shown to converge to a safe joint policy under certain assumptions [64], it changes the structure of the problem in a way that is not well understood, leading to two challenges.

The first challenge is that the primal-dual algorithm only enforces *discounted sum constraints* derived from the original safety constraints of the system. As we will show, discounted sum constraints guarantee safety only in expectation, which is difficult to interpret. We propose simple modifications to the penalty term that enable the enforcement of more interpretable constraints, namely, chance constraints [70] and conditional value at risk constraints [83], providing bounds on the probability and the severity of future constraint violations. There have been several C-RL algorithms that work with risk sensitivities [26, 41], but the multiagent context is less studied, and our contributions provide a novel understanding of the safety guarantees provided by C-MARL algorithms.

The second challenge is the fact that the reward is constantly changing as the dual variables are updated, which diminishes the accuracy of value estimates. We quantify this loss of accuracy, and we propose a new value estimation procedure to overcome it. Our proposal builds on results in [94] showing the affine relationship between the value function and the dual variables. We develop a novel class of temporal difference algorithms for value function estimation that directly exploits this observation, giving rise to a value estimate that maintains an accurate derivative with respect to the dual variables. Compared to existing algorithms, our estimates are much more robust to dual variable updates.

The specific C-MARL formulation we study in this chapter is inspired by the BEM problem [11, 72], illustrated in Figure 4.1. The main objective of BEM is

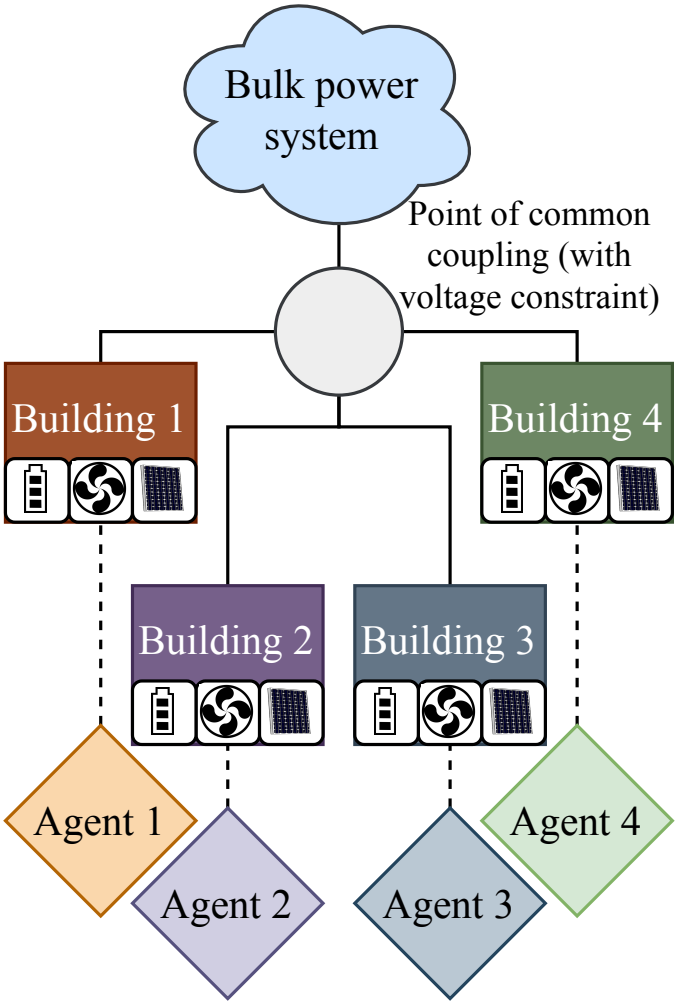

Figure 4.1: Building energy management with a voltage constraint at the point of common coupling. Key: PV = photovoltaic array, BESS = battery energy storage system, SWH = smart water heater, SCC = smart climate control system.

to control a building's resources to minimize the cost of energy consumption while maintaining comfort and convenience for the occupants. However, when BEMs are deployed in multiple buildings, it is critical to ensure that the power network connecting them is safely operated because the uncoordinated control of buildings can cause network-level voltage or power flow violations. This mandates a level of coordination among agents in the learning stage; thus, we adopt the commonly-studied centralized training/decentralized execution (CTDE) framework [40, 63], in which a simulator or coordinator provides global state information, constraint evaluations, and Lagrange multipliers (dual variables) to each agent during training. During the testing (execution) phase, we assume that there is no real-time communication between the agents. This stems from the need for privacy and the lack of communication infrastructure in practical systems[2].

The rest of the chapter is organized as follows. In Section 4.1, we formulate the problem under consideration. In Section 4.2, we provide an overview of our main interpretive tool, the occupation measure [16]. In Section 4.3, we use the occupation measure to reformulate discounted sum constraints as probabilistic constraints. In Section 4.4, we study the value structure of the primal-dual problem and use the results to propose a new value estimation algorithm. In Section 4.5, we provide some simulation results affirming the contribution of the theoretical observations.

Notation

The natural numbers and the real numbers are denoted $\mathbb{N}$ and $\mathbb{R}$, respectively. Given a measurable set $\mathcal{S}$, the set of all possible probability densities over $\mathcal{S}$ is denoted as $\Delta_{\mathcal{S}}$. For any discount factor $\gamma \in (0, 1)$ and any sequence $\{y_t\}_{t=0}^{T}$, the discounted sum operator is $\prod_{t=0}^{T}[y_t \mid \gamma] = (1-\gamma)\sum_{t=0}^{T}\gamma^t y_t$, and $\prod_{t=0}^{\infty}[y_t \mid \gamma] = \lim_{T\to\infty}\prod_{t=0}^{T}[y_t \mid \gamma]$ if the limit exists. We often drop the second argument γ for brevity. The positive

[2]Even in buildings with advanced metering infrastructure or smart meters, they typically only exchange information with the utility a few times a day.

component operator is $[y]_+ = \max\{y, 0\}$, and the logical indicator function $I[\cdot]$ maps $\{\text{True}, \text{False}\}$ to $\{1, 0\}$.

4.1 Problem formulation

4.1.1 Constrained MARL

We consider a noncooperative setting in which n agents pursue individual objectives while subject to global constraints (e.g., a shared resource constraint). We assume there is no real-time communication, and that each agent's action is based only on its local observations. However, policy updates can use global information under the CTDE framework [40, 63]. In this chapter, we consider the case of continuous state and action spaces.

The setting is described by the tuple $(\{\mathcal{X}_i\}_{i\in\mathcal{N}}, \{\mathcal{U}_i\}_{i\in\mathcal{N}}, \{R_i\}_{i\in\mathcal{N}}, f, C, p_0, \gamma)$, where $\mathcal{N}$ is the index set of agents, $\mathcal{X}_i \subset \mathbb{R}^{n_x^i}$ and $\mathcal{U}_i \subset \mathbb{R}^{n_u^i}$ are the state and action spaces of agent i, and $R_i : \mathcal{X}_i \times \mathcal{U}_i \to \mathbb{R}$ is the reward function of agent i. We assume that the sets $\mathcal{X}_i$ and $\mathcal{U}_i$ are compact for all i. Let $\mathcal{X} = \prod_{i\in\mathcal{N}} \mathcal{X}_i$ and $\mathcal{U} = \prod_{i\in\mathcal{N}} \mathcal{U}_i$ be the joint state and action spaces of the system, respectively. Then $f : \mathcal{X} \times \mathcal{U} \to \Delta_{\mathcal{X}}$ describes the state transition probabilities, i.e., $f(\cdot \mid x, u)$ is a probability density function. The function $C : \mathcal{X} \to \mathbb{R}^m$ is used to describe a set of safe states, $\mathcal{S} = \{x \in \mathcal{X} \mid C(x) \leq 0\}$.

Let $p_0 \in \Delta_{\mathcal{X}}$ denote the initial state probability density and $\gamma \in (0, 1)$ be a discount factor. At time t, the state, action, and reward of agent i are x_t^i, u_t^i, and r_t^i, respectively, and constraint j evaluates to $c_t^j = C^j(x_t)$. Using a quantity without a superscript to represent a stacked vector ranging over all $i \in \mathcal{N}$ or all $j \in \{1, \ldots, m\}$, a system trajectory is denoted $\tau = \{(x_t, u_t, r_t, c_t)\}_{t=0}^{\infty}$.

In the noncooperative C-MARL framework, each agent seeks to learn a policy $\pi_i : \mathcal{X}_i \to \Delta_{\mathcal{U}_i}$ that maximizes the expected discounted accumulation of individual rewards. We let $\pi : \mathcal{X} \to \Delta_{\mathcal{U}}$ denote the joint policy, and $f^\pi : \mathcal{X} \to \Delta_{\mathcal{X}}$ is the state transition probability induced by a joint policy π. The tuple (p_0, f, π) induces a state

visitation probability density at each time step, $p_t^\pi(x) = \int_{\mathcal{X}^t} p_0(x_0) \cdot \prod_{k=1}^t f^\pi(x_k \mid x_{k-1})\, dx_0 \ \ldots \ dx_{k-1}$, and we say $p_\infty^\pi(x) = \lim_{t\to\infty} p_t^\pi(x)$ for each $x \in \mathcal{X}$ if the limit exists. The collection of visitation probabilities $\{p_t^\pi\}_{t=0}^\infty$ gives rise to a probability density of trajectories τ, denoted $\mathcal{M} \in \Delta_{\prod_{t=0}^\infty (\mathcal{X}\times\mathcal{A}\times\mathbb{R}^n\times\mathbb{R}^m)}$; thus, the objective of each agent can be stated precisely as maximizing $\mathbb{E}_{\tau\sim\mathcal{M}}[\prod_{t=0}^\infty r_t^i]$.

The agents, however, must settle on a joint policy that keeps the system in the safe set $\mathcal{S}$. Due to the stochastic nature of the system, satisfying this constraint at all times is too difficult and in some cases too conservative. A common relaxation procedure is to formulate an augmented reward $\tilde{r}_t^i = r_t^i - \lambda^T c_t$ where $\lambda \in \mathbb{R}_+^m$, the *Lagrange multiplier* or *dual variable*, is adjusted to incentivize constraint satisfaction. This leads to the primal-dual algorithm for C-MARL, discussed in the next section. The following mild assumption facilitates the analysis.

Assumption 4.1. R^i, C^j, and p_t^π are bounded on $\mathcal{X}$ for all $i \in \mathcal{N}$, all $j \in \{1,\ldots,m\}$, and all $t \in \mathbb{N}$.

The boundedness of R^i and C^j is a common assumption [64,79,94] that we will use to exchange the order of limits, sums, and integrals using the dominated convergence theorem. The assumption of bounded p_t^π is not strictly necessary and does not change the results; however, we use it throughout the chapter to simplify calculations.

4.1.2 Primal-dual algorithms

The augmented reward function leads to the following min-max optimization problem for agent i:

$$\min_{\lambda\geq 0} \max_{\pi_i} \mathbb{E}_{\tau\sim\mathcal{M}}\Big[\prod_{t=0}^\infty [r_t^i - \lambda^T c_t]\Big] \tag{4.1}$$

$$= \min_{\lambda\geq 0} \max_{\pi_i} \left(\mathbb{E}_{\tau\sim\mathcal{M}}\Big[\prod_{t=0}^\infty [r_t^i]\Big] - \lambda^T \mathbb{E}_{\tau\sim\mathcal{M}}\Big[\prod_{t=0}^\infty [c_t]\Big]\right) \tag{4.2}$$

where (4.2) uses absolute convergence (stemming from Assumption 4.1) to rearrange the terms of the infinite sum. Note that the minimization over λ is coupled

across agents. Any fixed point of (4.2) will satisfy $\mathbb{E}_{\tau\sim\mathcal{M}}[\prod_{t=0}^{\infty} c_t] \leq 0$ because if $\mathbb{E}_{\tau\sim\mathcal{M}}[\prod_{t=0}^{\infty} c_t^j] \neq 0$, then the objective value can be reduced by increasing or decreasing λ_j, unless $\mathbb{E}_{\tau\sim\mathcal{M}}[\prod_{t=0}^{\infty} c_t^j] < 0$ and $\lambda_j = 0$. In other words, the primal-dual method enforces a *discounted sum constraint* derived from the safe set $\mathcal{S}$. Although discounted sum constraints are convenient, it is not obvious what they imply about safety guarantees with respect to the original constraints. We begin our investigation of discounted sum constraints by taking a closer look at a state visitation probability density known as the occupation measure.

4.2 Occupation measure

The *occupation measure* describes the average behavior of a Markov process in some sense which will be made precise shortly. As we will show, the occupation measure is instrumental in clarifying the role of discounted sum constraints. In this chapter, we use a definition common for continuous-state, infinite-horizon discounted MDPs [79, 88].

Definition 4.1 (Occupation measure)**.** The occupation measure $\mu_\gamma^\pi \in \Delta_\mathcal{X}$ associated with discount factor γ, induced by a joint policy π, is defined for any $x \in \mathcal{X}$ as $\mu_\gamma^\pi(x) = \prod_{t=0}^{\infty} p_t^\pi(x)$.

In this section, we provide some interpretations for the occupation measure before using it to ascribe meaning to discounted sum constraints. The first question one might ask is whether μ_γ^π is itself a pdf. It is, of course, nonnegative, and the following proposition shows it integrates to unity under mild conditions.

Proposition 4.1. Under Assumption 4.1, $\int_\mathcal{X} \mu_\gamma^\pi(x)dx = 1$.

The proof for Proposition 4.1 is in Appendix B.1. What does μ_γ^π tell us about the behavior of a system under a given policy? It describes the probability of visiting a certain state but with more weight placed on states that are likely to be visited *earlier* in time. In fact, μ_γ^π describes the near-term behavior in the following sense.

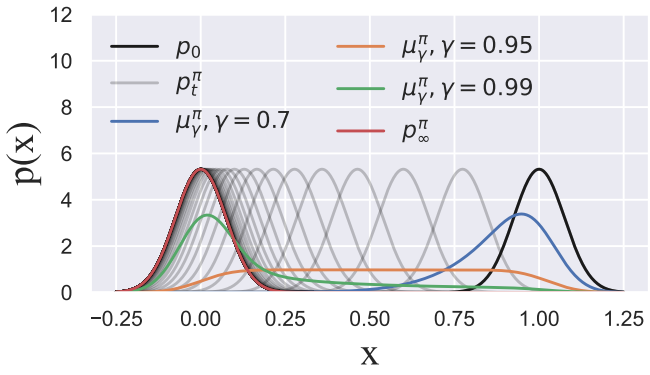

Figure 4.2: Example of the occupation measure for various levels of γ.

Proposition 4.2. Under Assumption 4.1, for any $x \in \mathcal{X}$, the following statements hold:

1. $\lim_{\gamma \to 0^+} \mu_\gamma^\pi(x) = p_0(x)$.

2. $\lim_{\gamma \to 1^-} \mu_\gamma^\pi(x) = \lim_{t \to \infty} p_t^\pi$ if the latter limit exists.

The proof for Proposition 4.2 is in Appendix B.1. Figure 4.2 provides an illustration of the result in Proposition 4.2 when p_t^π evolves as a normal distribution with mean 0.95^t and constant variance. The point at which μ_γ^π equally resembles p_0 and p_∞^π is exactly at $\gamma = 0.95$.

According to Proposition 4.2, the occupation measure describes a state distribution that lies between the initial and long-term behavior of the system. But where exactly does it lie in between these two extremes? The *effective horizon* of a discounted planning problem is often set to $T_1(\gamma) = \frac{1}{1-\gamma}$, which is the expected termination time if the probability of an episode terminating at any given time step is $(1 - \gamma)$ [78]; however, the concept of a random stopping time might not be sensible in all applications. Another way to define the effective horizon is to study the geometric accumulation of weights. In this case, the effective horizon can be measured

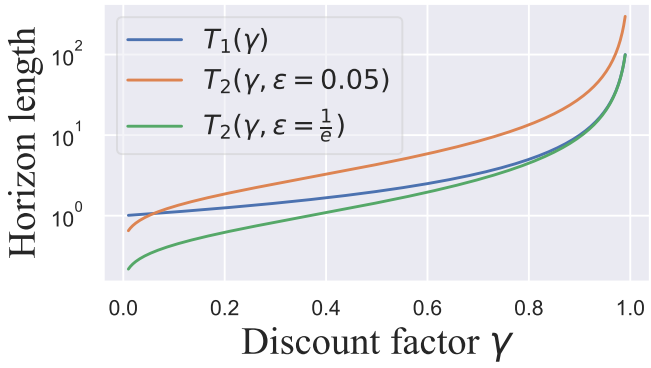

Figure 4.3: Effective horizon length as a function of γ.

as $T_2(\gamma, \varepsilon) = \min\{K \in \mathbb{N} : \prod_{t=0}^{K-1}[1] \geq 1 - \varepsilon\}$, where $\varepsilon \in (0, 1)$ is a tolerance. Using either of these two definitions, the occupation measure can be said to describe the behavior of the system from the start time up to the effective horizon. Specifically, one may truncate the sum in Definition 4.1 at the effective horizon to obtain a conceptual understanding of what the occupation measure describes.

Depending on the application, either T_1 or T_2 can provide a more sensible connection between discounted and finite-horizon problems. But are these two definitions related? The next proposition answers this affirmatively by showing that T_1 is actually a special case of T_2.

Proposition 4.3. $T_1(\gamma) = T_2(\gamma, \varepsilon)$ when ε is set to $\gamma^{\frac{1}{1-\gamma}} \approx \frac{1}{e}$.

The proof for Proposition 4.3 is in Appendix B.1. Proposition 4.3 is illustrated in Figure 4.3, where the effective horizon is plotted as a function of γ for three different values of ε. With an understanding of the occupation measure as a visitation density describing behavior up to the effective horizon, we can begin to derive meaningful risk-related interpretations of discounted sum constraints. These interpretations lead directly to sensible recommendations for the design of C-MARL algorithms.

4.3 Discounted risk metrics

The discounted sum constraint can naturally be reinterpreted as a certain type of average constraint. In particular, Assumption 4.1 ensures the equivalence $\mathbb{E}_{\tau\sim\mathcal{M}}[\prod_{t=0}^{\infty} C(x_t)] = \mathbb{E}_{x\sim\mu_\gamma^\pi}[C(x)]$ [79]. This near-term average does not relate to any well-known risk metrics and hence does not provide a practical safety guarantee. In general, information about the mean of a distribution cannot be used to infer information about its tails; however, simple changes to the penalty function can yield information about either the *probability* of incurring a constraint violation or the expected *severity* of constraint violations.

Proposition 4.4 (Near-term probability of constraint violations)**.** Suppose that for some $\delta_j \in [0, 1]$ and $\alpha_j \in \mathbb{R}$, we have $\mathbb{E}_{\tau\sim\mathcal{M}}[\prod_{t=0}^{\infty} I[C^j(x_t) \geq \alpha_j]] \leq \delta_j$. Then under Assumption 4.1, $\Pr\{C^j(x) \geq \alpha_j \mid x \sim \mu_\gamma^\pi\} \leq \delta_j$.

Proof. $\mathbb{E}_{\tau\sim\mathcal{M}}[\prod_{t=0}^{\infty} I[C^j(x_t) \geq \alpha_j]] = \mathbb{E}_{x\sim\mu_\gamma^\pi}[I[C^j(x) \geq \alpha_j]] = \Pr\{C^j(x) \geq \alpha_j \mid x \sim \mu_\gamma^\pi\}$. The first equality uses Assumption 4.1 to apply an equivalence established in e.g. [79]. The second equality follows from the definition of expectation. □

Proposition 4.4 makes it easy to enforce chance constraints using primal-dual methods. When the penalty term $C^j(x)$ is replaced by the quantity $I[C^j(x) \geq \alpha_j] - \delta_j$, the primal-dual algorithm enforces $\mathbb{E}_{\tau\sim\mathcal{M}}[\prod_{t=0}^{\infty} I[C^j(x_t) \geq \alpha_j]] - \delta_j \leq 0$. By Proposition 4.4, this guarantees that $\Pr\{C^j(x) \geq \alpha_j \mid x \sim \mu_\gamma^\pi\} \leq \delta_j$. Because the probability of constraint violations is defined with x varying over μ_γ^π, we call the resulting guarantee a *near-term* or *discounted chance constraint.* This can be repeated for each $j \in \{1, \ldots, m\}$, providing a set of bounds on the probability of violating *each* constraint by more than its tolerance α_j. On the other hand, we can control the probability of violating *any* constraint as follows. Define the statement $C(x) \geq \alpha$ to be true if $C^j(x) \geq \alpha_j \ \forall \ j \in \{1, \ldots, m\}$, and false otherwise. Then, applying Proposition 4.4 to the test condition $C(x) \geq \alpha$ will result in a bound on

$\Pr\{C(x) \geq \alpha \mid x \sim \mu_\gamma^\pi\}$.

While discounted chance constraints enable one to control the *probability* of extreme events in the near future, conditional value at risk constraints [83] afford control over the *severity* of such events.

Definition 4.2 ([83])**.** Given a risk level $\beta \in (0,1)$, a cost $h : \mathcal{X} \to \mathbb{R}$, and a probability density μ on $\mathcal{X}$, the value at risk (VaR) and conditional value at risk (CVaR) are defined as:

$$\mathrm{VaR}(\beta, h, \mu) = \min\{\alpha \in \mathbb{R} : \Pr\{h(x) \leq \alpha \mid x \sim \mu\} \geq \beta\},$$
$$\mathrm{CVaR}(\beta, h, \mu) = \frac{1}{1-\beta} \int_{h(x) \geq \mathrm{VaR}(\beta,h,\mu)} h(x)\mu(x)dx.$$

In other words, $\mathrm{VaR}(\beta, h, \mu)$ is the least upper bound on h that can be satisfied with probability β, while $\mathrm{CVaR}(\beta, h, \mu)$ describes the expected value in the VaR-tail of the distribution of h. CVaR characterizes the expected severity of extreme events, which can be defined precisely as the $(1-\beta)$ fraction of events x with the worst outcomes as ranked by the cost incurred, $h(x)$. The VaR and CVaR for $h(x) = x$, when x follows a standard normal distribution, are illustrated in Figure 4.4, where the shaded region has an area of $(1-\beta)$. For the rest of the chapter, we assume that the cdf of $h(x)$ is continuous when $x \sim \mu$. For further details and for cases in which this assumption does not hold, we refer the reader to [84].

Proposition 4.5 (Near-term CVaR)**.** For any $\alpha_j \geq 0$, suppose that $\mathbb{E}_{\tau\sim\mathcal{M}}[\prod_{t=0}^{\infty}[[C^j(x_t) - \alpha_j]_+]] \leq \delta_j$. Then, $\mathrm{CVaR}(\beta, C^j, \mu_\gamma^\pi) \leq \alpha_j + (1-\beta)^{-1}\delta_j$.

Proof. Under Assumption 4.1, the identity $\mathbb{E}_{\tau\sim\mathcal{M}}[\prod_{t=0}^{\infty}[C^j(x_t) - \alpha_j]_+] = \mathbb{E}_{x\sim\mu_\gamma^\pi}[[C^j(x) - \alpha_j]_+]$ holds [79]. Next, we use the fact that the CVaR is the minimum value of the convex function in α_j given by $F(\alpha_j \mid \beta, C^j, \mu_\gamma^\pi) := \alpha_j + (1-\beta)^{-1}\mathbb{E}_{x\sim\mu_\gamma^\pi}[[C^j(x) - \alpha_j]_+]$ [83]; thus, F provides an upper bound on CVaR. Some rearranging leads to the result. □

Similar to the chance-constrained case, Proposition 4.5 makes it easy to enforce CVaR constraints in the primal-dual algorithm. Here, the penalty term used is

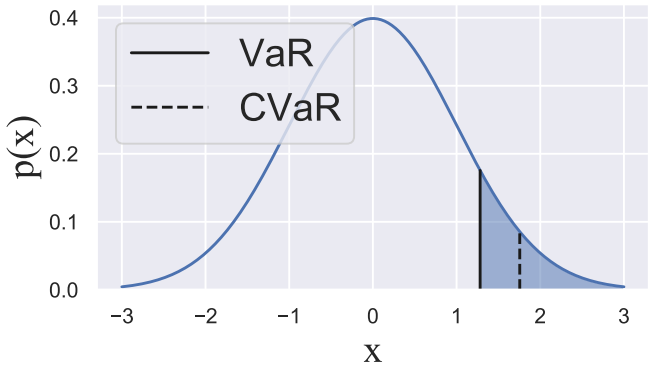

Figure 4.4: Example of VaR and CVaR at risk level $\beta = 0.9$.

$[C^j(x) - \alpha_j]_+ - \delta_j$. Using this penalty, the algorithm enforces $\mathbb{E}_{\tau\sim\mathcal{M}}[\prod_{t=0}^{\infty}[[C^j(x_t) - \alpha_j]_+]] - \delta_j \leq 0$, which by Proposition 4.5 implies $\text{CVaR}(\beta, C^j, \mu_\gamma^\pi) \leq \alpha_j + (1-\beta)^{-1}\delta_j$. By repeating for each $j \in \{1, \ldots, m\}$, we can bound the expected severity of the constraint violations for each of the m constraints. Because the CVaR constraint is defined with x varying over μ_γ^π, the resulting guarantee is called a *near-term* or *discounted CVaR constraint.*

To obtain a tight bound on the CVaR, α_j must be set to $\text{VaR}(\beta, C^j, \mu_\gamma^\pi)$, which minimizes the function F introduced in the proof of Proposition 4.5 [83]. Unfortunately, the VaR is not known ahead of time. [26] include α_j as an optimization variable in the learning procedure, but extending their technique to the multiagent setting is not straightforward. Our approach is to include it as a tunable hyperparameter. Simulation results in Section 4.5 show that it is easy to choose α_j to give a nearly tight bound.

4.4 Primal-dual value functions

In this section, we investigate challenges with value estimation in the primal-dual regime. The fact that the reward to each agent is constantly changing (due to dual variable updates) makes it difficult to accurately estimate state values. To quantify

this decrease in accuracy, we introduce the value functions induced by the joint policy π, $\{V^i_\pi : \mathcal{X} \times \mathbb{R} \to \mathbb{R}\}_{i\in\mathcal{N}}$, $\{V^i_{R,\pi} : \mathcal{X} \to \mathbb{R}\}_{i\in\mathcal{N}}$, $V_{C,\pi} : \mathcal{X} \to \mathbb{R}^m$ where:

$$V^i_\pi(x,\lambda) = \mathbb{E}_{\tau\sim\mathcal{M}}[\prod_{t=0}^{\infty} r^i_t - \lambda^T c_t \mid x_0 = x], \tag{4.3}$$

$$V^i_{R,\pi}(x) = \mathbb{E}_{\tau\sim\mathcal{M}}[\prod_{t=0}^{\infty} r^i_t \mid x_0 = x], \quad V_{C,\pi}(x) = \mathbb{E}_{\tau\sim\mathcal{M}}[\prod_{t=0}^{\infty} c_t \mid x_0 = x]. \tag{4.4}$$

Note that c_t could be modified as indicated in Section 4.3, and the following results would hold for the modified penalty function.

Obviously, it is impossible to learn an accurate value function when λ is unknown and changing; however, simply making λ available to a value function approximator does not guarantee good generalization beyond previously seen values of λ. Having a good estimate of the *derivative* of the value function with respect to λ will ensure accuracy under small perturbations to the dual variables. Fortunately, this derivative is easy to obtain. Formally, under Assumption 4.1, we can write $V^i_\pi(x,\lambda) = V^i_{R,\pi}(x) - \lambda^T V_{C,\pi}(x)$ [94], and therefore, $\nabla_\lambda V^i_\pi(x,\lambda) = -V_{C,\pi}(x)$. By learning $V^i_{R,\pi}$ and $V_{C,\pi}$ as separate functions and then combining them using the true value of λ, we can construct a value estimate whose derivative with respect to the dual variables is as accurate as our estimate of $V_{C,\pi}$ itself. This estimate will be more robust to small changes in λ. We will refer to this type of value estimate as a *structured value function* or a *structured critic*.

Proposition 4.6. Let $\bar{c} = \mathbb{E}_{x\sim\mu^\pi_\gamma}[C(x)]$ and $\Sigma^2_C = \mathbb{E}_{x\sim\mu^\pi_\gamma}[(\bar{c} - C(x))(\bar{c} - C(x))^T]$. Suppose λ is randomly varying with mean $\bar{\lambda}$ and variance Σ^2_λ. Using a structured value function approximator can reduce the mean square temporal difference error by up to $\mathrm{Tr}[\Sigma^2_\lambda \cdot (\Sigma^2_C + \bar{c}\bar{c}^T)]$.

The proof of Proposition 4.6 is in Appendix B.1. Figure 4.5 illustrates Proposition 4.6 in a simple value estimation task with quadratic rewards, linear dynamics and policies, linear state constraints, and randomly varying λ. The *generic critic* (GC)

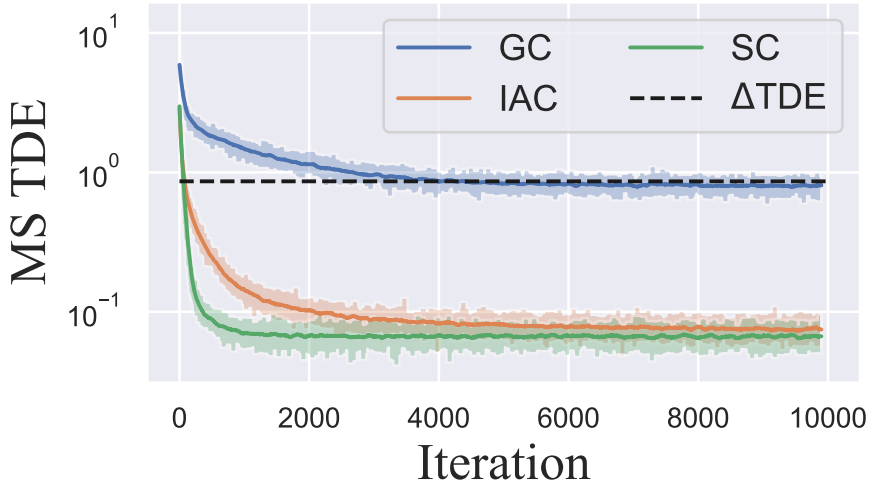

Figure 4.5: Temporal difference error trajectories in a simple policy evaluation task.

is a value function modeled as a quadratic function of the state only. The *input-augmented critic* (IAC) is a value function modeled as an unknown quadratic function of the state and dual variables, while the *structured critic* (SC) is modeled using $\hat{V}^i_\pi = \hat{V}^i_{R,\pi} - \lambda^T \hat{V}_{C,\pi}$ with quadratic $\hat{V}^i_{R,\pi}$ and linear $\hat{V}_{C,\pi}$ trained on their respective signals.

The dashed line in Figure 4.5 is at the value $\text{Tr}[\Sigma^2_\lambda \cdot (\Sigma^2_C + \bar{c}\bar{c}^T)]$ predicted in Proposition 4.6. In this simple value estimation task, high accuracy can be achieved when conditioning on the randomly varying λ; however, having an accurate estimate of $\nabla_\lambda V^i_\pi$ by using a structured critic is also shown to help. Although in practice $\bar{\lambda}$ and Σ^2_λ change over time, the simulation results in Section 4.5 confirm that using structured critics improves performance. The loss function for value function approximation is therefore given by:

$$TDE(x, x') = [R^i(x^i) + \gamma \hat{V}^i_{R,\pi}((x^i)') - \hat{V}^i_{R,\pi}(x^i)]^2 + \|C(x) + \gamma \hat{V}_{C,\pi}(x') - \hat{V}_{C,\pi}(x)\|_2^2 \tag{4.5}$$

where $x \in \mathcal{X}$ and $x' \sim f^{\pi}(x)$. Equation (4.5) is simply a sum of squared temporal difference errors over the set of $m+1$ value functions. For algorithmic details, we refer the reader to Appendix B.2.

4.5 Simulations

4.5.1 Multiagent Particle Environment

In our simulations, we sought to demonstrate the effectiveness of the penalty modifications and structured critic proposed in sections 4.3 and 4.4. We tested our findings in a modified multiagent particle environment[3] [63] with two agents pursuing individual objectives subject to a constraint on the joint state. The state of each agent is its position and velocity in $\mathbb{R}^2$, i.e. $x^i = \begin{bmatrix} y^{iT} & v^{iT} \end{bmatrix}^T$ where $y^i \in \mathbb{R}^2$ is the position and $v^i \in \mathbb{R}^2$ is the velocity of agent i. The objective of each agent is to drive its position y^i to a landmark $y^{i*} \in \mathbb{R}^2$, while making sure that the agent ensemble satisfies the safety constraint. The reward and constraint functions are given by:

$$R^i(y^i) = -\xi_i \|y^i - y^{i*}\|_2^2, \quad C(y) = \mathbf{1}^T y \tag{4.6}$$

where $\xi_i > 0$ is a constant and $y = \begin{bmatrix} y^{1T} & y^{2T} \end{bmatrix}^T$ is the position of the agent ensemble.

The landmark $y^* = \begin{bmatrix} y^{1*T} & y^{2*T} \end{bmatrix}^T$ is stationed outside of the safe region $\mathcal{S} = \{y \mid C(y) \leq 0\}$. Thus, the agents cannot both reach their goals while satisfying $C(y) \leq 0$. To train the agents to interact in this environment, we used a modified version of the EPyMARL codebase[4] [75]. We tested several MARL algorithms, including MADDPG [63], COMA [40], and MAA2C [75]. We decided to use the MAA2C algorithm because it consistently produced the best results and because as a value function-based algorithm, it provided the most straightforward route to implementing the changes proposed in Section 4.4. Details of the algorithm, pseudocode, hyperparameters, and supplementary simulation results are provided in Appendix B.2.

---

[3] Code for the environments is available at `github.com/dtabas/multiagent-particle-envs`.

[4] Code for the algorithms is available at `github.com/dtabas/epymarl`.

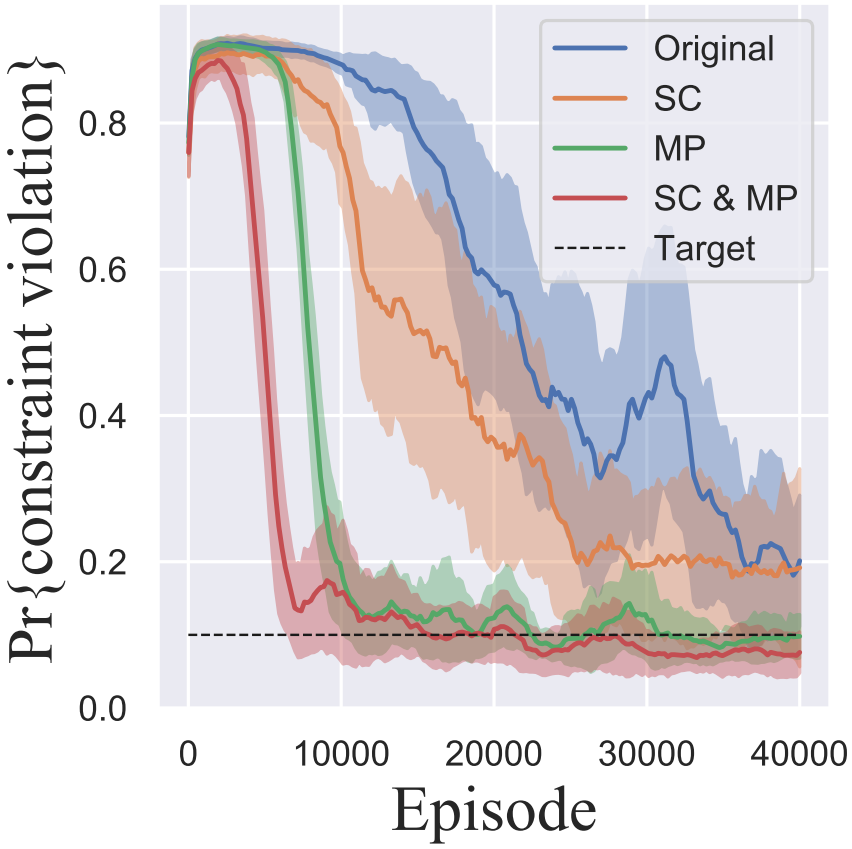

Figure 4.6: $\Pr\{C(x) \geq 0.1 \mid x \sim \mu_\gamma^\pi\}$ measured throughout training. Key: SC = structured critic, MP = modified penalty (Prop. 4.4). Both modifications speed convergence to a safe policy. The shaded region represents ± 1 standard deviation across 5 training runs.

For each risk metric described in Section 4.3, we tested the convergence of the agents to a safe policy with and without modifications to the penalty and value functions. Figure 4.6 shows the results when we make the substitution $C(x) \leftarrow I[C(x) \geq \alpha] - \delta$ in the penalty function to enforce a chance constraint, $\Pr\{C(x) \geq \alpha \mid x \sim \mu_\gamma^\pi\} \leq \delta$ with α and δ each set to 0.1. The modified penalty function performs the best as a chance constraint-enforcing signal (red and green lines in Figure 4.6). Whether or not the penalty function is modified, the structured critic finds safer policies throughout training (red vs. green and orange vs. blue lines).

Figure 4.7 shows the results when we make the substitution $C(x) \leftarrow [C(x) - \alpha]_+ - \delta$ in the penalty function to enforce the constraint $\text{CVaR}(\beta, C, \mu_\gamma^\pi) \leq \alpha + (1-\beta)^{-1}\delta$. Using the modified penalty (red and green lines in Figure 4.7) drives the CVaR upper bound (drawn in dashed lines) to the target value, and due to the choice of α, this bound is nearly tight. On the other hand, using the original penalty results in an overly conservative policy that achieves low risk at the expense of rewards (right panel). We also point out that when using the modified penalty with the structured critic, the CVaR is lower throughout training compared to when the generic critic is used, indicating improved effectiveness in enforcing limits on risk.

We chose α using the following heuristic, to make the bound on CVaR nearly tight. The "correct" value of α that would achieve a tight bound is $\text{VaR}(\beta, C, \mu_\gamma^\pi)$. Moreover, the upper bound that we used is convex and continuously differentiable in α [83]; therefore, small errors in α will lead to small errors in the upper bound on CVaR, and any approximation of the VaR will suffice. We obtained an approximation simply by running the simulation once with α set to zero and computing $\text{VaR}(\beta, C, \mu_\gamma^\pi)$ over some test trajectories. If necessary, the process could be repeated additional times. Alternatively, α could be tuned adaptively by computing VaR online, but the stability of such a procedure would need further investigation.

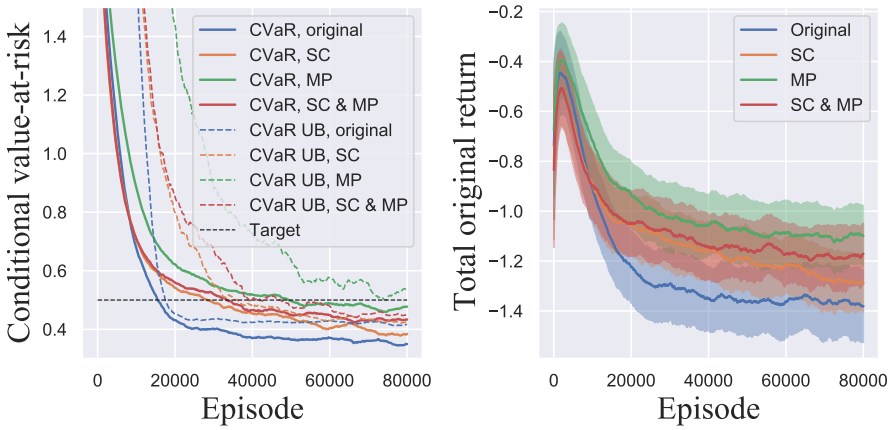

Figure 4.7: CVaR($\beta = 0.9, C, \mu^{\pi}_{\gamma}$) measured throughout training. Key: SC = structured critic, MP = modified penalty (Prop. 4.5). The dashed lines represent the CVaR upper bound used in Prop. 4.5. The panel on the right shows progress toward the original objective through the total original returns, $\sum_{i=1}^{2} \Gamma_{t=0}^{T} r_t^i$, without penalty terms. The shaded region represents ± 1 standard deviation across 5 training runs. The rewards increase then decrease because the agents first learn to navigate towards the landmark, which is outside the safe region, then learn to back off to satisfy the constraint.

4.5.2 Building energy management environment

We next tested the primal-dual method with modified penalty functions in a three-building PowerGridWorld environment with a distribution system solver [11]. Each building consists of the following behind-the-meter controllable components: an HVAC system, a PV array, and a battery energy storage system. Each building is controlled by one agent operating the three components. The objective of each agent is to operate the HVAC system in a way that satisfies a soft constraint on internal building temperature (the environment can easily be modified to include costs on energy consumption, etc). The agents must learn to operate their respective buildings in a scenario in which a sudden drop in PV availability threatens an undervoltage event at the feeder head. For additional details, we refer the reader to [11], Section 3.1.

We enforce a system-wide CVaR constraint on the voltage at the feeder head as described in Section 4.3 and train the agents using a modified version of the MADDPG algorithm [63]. The results demonstrate the sufficiency of the CVaR condition in Proposition 4.5 (Figure 4.8). Although the dual variable converges to 0, indicating that a strictly feasible equilibrium joint policy exists, the result nonetheless serves as a certificate that the CVaR constraint is satisfied. Due to the highly nonlinear system physics, necessity of the criterion in Proposition 4.5 is not guaranteed.

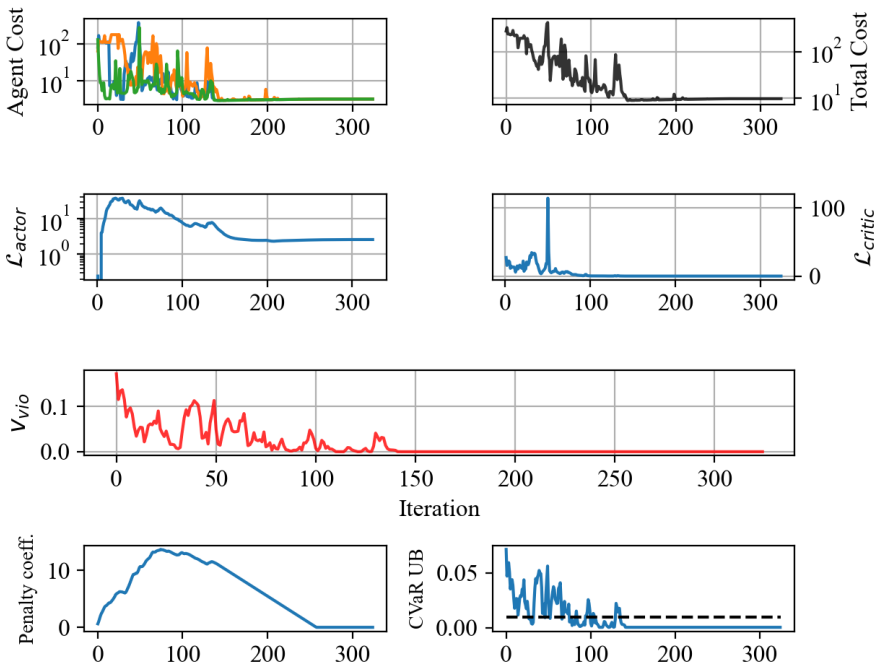

Figure 4.8: Learning curves for the 3-building scenario. The horizontal axis is in increments of 1000 training episodes, for all plots. *Top row:* Individual agent costs (left) and total cost (right), corresponding to building temperature deviations. *Second row:* average actor loss (left) and critic loss (right). *Third row:* voltage excursions from the interval $[0.95, 1.05]$ per unit (cumulative per episode). *Last row:* Dual variables and CVaR upper bound from Proposition 4.5.

Chapter 5

CONCLUSION

In Chapter 2, we proposed an efficient approach to safety-critical, data-driven control. The strategy relies on results from set-theoretic control and convex analysis to provide provable guarantees of constraint satisfaction. Importantly, the proposed policy chooses actions without solving an optimization problem, opening the door to safety-critical control in applications in which computational power may be a bottleneck. We applied the proposed controller to a frequency regulation problem, but the applications are much more wide-ranging[1].

In Chapter 3, we provided an efficient way of exploring the interior of the MPC feasible set for learning-based approximate explicit MPC, and demonstrated the performance and computational gains that can be achieved by approaching the problem from the interior. The paradigm relies on a Phase I solution that exploits the structure of the MPC problem and a Phase II solution that features a projection-free feasibility guarantee. The results compare favorably against common approaches that use unsupervised learning, as well as against the oracle itself used in supervised approaches[2].

One direction for future research on gauge map techniques is to study the generalization capabilities of policy neural networks that use a gauge map as a safety filter. If the system being controlled undergoes a change (such as a line outage), what is the resulting performance of the already-trained policy? We have reason to believe that

[1]Paragraph adapted from D. Tabas and B. Zhang, "Computationally Efficient Safe Reinforcement Learning for Power Systems," 2022 American Control Conference (ACC), Atlanta, GA, USA, 2022, pp. 3303-3310.

[2]Paragraph adapted from D. Tabas and B. Zhang, "Safe and Efficient Model Predictive Control Using Neural Networks: An Interior Point Approach," 2022 IEEE 61st Conference on Decision and Control (CDC), Cancun, Mexico, 2022, pp. 1142-1147.

policy networks designed using the principles in Chapter 2 will maintain some level of performance following such a change. The reason is that even if the set-valued map from states to safe actions changes, the optimal action from a particular state (when mapped backwards from the safe action set to the infinity norm ball) might remain relatively unchanged. Since the gauge map safety layer involves scaling and translation but no rotation, the physical meaning of the direction of a specified control action has a degree of consistency before and after a change in parameter values. Another way to achieve generalization is to condition the policy network on contextual variables (such as parameter values) and train across multiple system configurations.

Chapters 2 and 3 demonstrated how gauge maps can be used to design policy networks for frequency regulation. However, the ability to design neural networks whose outputs are constrained to nearly arbitrary full-dimensional polytopes has far-reaching implications for data-driven control and optimization. The technique has already been picked up and studied for nonlinear control [100], parametric optimization [58], and solving robust versions of the DC optimal power flow problem [105]. New advances in convex restrictions for AC optimal power flow [104], when combined with gauge map techniques, promise to yield new and powerful tools for power system operations.

In Chapter 4, we studied the effect of primal-dual algorithms on the structure of constrained MARL problems. First, we used the occupation measure to study the effect of the penalty term on safety. We showed that using the constraint function as the penalty enforces safety only in expectation, but by making simple modifications to the penalty term, one may enforce meaningful probabilistic safety guarantees, namely, chance and CVaR constraints. These risk metrics are defined over the occupation measure, leading to notions of safety in the near term. Next, we studied the effect of the penalty term on the value function. When the dual variable and constraint evaluation signals are available, it is easy to model the relationship between the penalty term and the value function. By exploiting this structure, the accuracy of

the value function can be improved. We demonstrated the usefulness of both of these insights in simulations, showing that convergence to a low-risk policy is accelerated[3].

One direction for future research in primal-dual based MARL algorithms is the question of convergence. The work in Chapter 4 studied the behavior of learned policies assuming that some equilibrium had been reached. However, it remains to show under which conditions an equilibrium will actually be reached, as well as whether that equilibrium will represent a social optimum. Some hardness results [30] indicate that these questions will have to be answered for particular problem instances. Also, some recent counterexamples [19] suggest that optimal constraint-satisfying policies must be conditioned on dual variables updated in real time, contradicting previous results on the convergence of primal-dual RL algorithms [15, 94]. In terms of social optimality, the objective function for the dual problem can be modified in order to incorporate notions of fairness, such as enforcing equal contribution to constraint violations from all agents. Taken together, such results will greatly improve the applicability of MARL in constrained, non-cooperative environments.

[3]Paragraph adapted from Tabas, D., Zamzam, A.S. and Zhang, B. (2023). Interpreting Primal-Dual Algorithms for Constrained Multiagent Reinforcement Learning. *Proceedings of The 5th Annual Learning for Dynamics and Control Conference*, in *Proceedings of Machine Learning Research* 211:1205-1217.

ACKNOWLEDGEMENTS

I would like to begin by thanking my advisor, Prof. Baosen Zhang, for being an invaluable mentor and resource. I am grateful for the level of support and academic freedom he provided throughout my studies, and will always hold on to his advice to "just try more things" whether or not I happen to be stuck on a problem. I would also like to thank the other UW faculty members who had an impact on my education: Daniel Kirschen, Lillian Ratliff, Maryam Fazel, Sam Burden, Brian Johnson, Behçet Açikmeşe, Kevin Jamieson, Jamie Morgenstern, and James Burke for their exceptional teaching capabilities, individual attention to students, and thought-provoking discussions.

Next, I would like to thank my internship mentors: Aftab Alam (CAISO), Ahmed Zamzam (NREL), and Anup Menon (Tesla). During each of my internships, I was given substantial responsibilities and full freedom to pursue solutions as I saw fit. My insistence on understanding and defining the problem at hand was always met with patience and encouragement, and I left each internship with deeper knowledge of a different aspect of modern power systems.

Special thanks go out to my labmates, classmates, and collaborators: Ling Zhang, Wenqi Cui, Jiayi Li, Trager Joswig-Jones, Matt Motoki, Yan Jiang, Yize Chen, Tinu Ademola-Idowu, Yuanyuan Shi, Chase Dowling, Lane Smith, Gord Steven, Daniel Olsen, Ryan Elliott, Mareldi Ahumada-Parás, Jackie Baum, Nina Vincent, Max Zhelyeznyakov, Rahul Mallik, Sarah Li, Dan Calderone, Satpreet Singh, Mitas Ray, Leo Zheng, Adhyyan Narang, Patrick Emami, Xiangyu Zhang, and others. From meetings and homework help to reading groups and late-night gripe sessions, I learned so much from each of them.

I would also like to thank my family for their unwavering support, for visiting me in the Pacific Northwest and joining me on road trips across the country, and for listening as I tried to explain neural networks, chance constraints, and the reasons why ChatGPT won't turn into Skynet.

Next, I would like to thank my friends: thank you for both the type I fun and the type II fun, and for helping me foster a sense of community and belonging. Although I can't possibly thank everyone individually here, I want to give a special shoutout to Gabe Mantegna for being my friend, advocate, adventure buddy, energy colleague, and co-investigator into the "permanent problem" of sustainability[4], and for introducing me to the field of sequential decision analytics[5]. Thank you especially to all my housemates over the years for making 4304 such a great place to live, and to everyone in the power systems community who welcomed me and inspired me.

Finally, I would like to acknowledge the various institutions around the state of Washington that helped make my time as a graduate student so special: NPS, USFS, WA DNR, Northwest Avalanche Center, Evergreen Mountain Bike Alliance, Race Cascadia, American Whitewater, Washington Trails Association, Crystal Mountain, the Summit at Snoqualmie, Seattle Parks & Rec, Burke Gilman Brewing, El Camion, Un Bien, Met Market, Winco, Bryant Corner Cafe, etc. I would also like to acknowledge the tribes of Washington, the original caretakers of the land on which I lived, studied, and recreated. Washington is the perfect place to do a PhD and I'm grateful to have called this state home for the past five years.

---

[4] `niskanencenter.org/the-permanent-problem`

[5] `castle.princeton.edu/sda`

BIBLIOGRAPHY

[1] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 22–31. PMLR, 06–11 Aug 2017.

[2] Atinuke Ademola-Idowu and Baosen Zhang. Frequency stability using mpc-based inverter power control in low-inertia power systems. *IEEE Transactions on Power Systems*, 36(2):1628–1637, 2020.

[3] U.S. Energy Information Administration. November 2023 monthly energy review, 2023.

[4] Akshay Agrawal, Brandon Amos, Shane Barratt, Stephen Boyd, Steven Diamond, and J. Zico Kolter. Differentiable convex optimization layers. *Advances in Neural Information Processing Systems*, 32(NeurIPS), 2019.

[5] Bernt M. Åkesson and Hannu T. Toivonen. A neural network model predictive controller. *J. Process Control*, 16(9):937–946, 2006.

[6] Alessandro Alessio and Alberto Bemporad. A Survey on Explicit Model Predictive Control. In L. Magni, D.M. Raimondo, and F. Allgöwer, editors, *Nonlinear Model Predictive Control*. Springer, Berlin, Heidelberg, 2009.

[7] Anil Aswani, Humberto Gonzalez, S. Shankar Sastry, and Claire Tomlin. Provably safe and robust learning-based model predictive control. *Automatica*, 49(5):1216–1226, 2013.

[8] A.G. Baydin, Alexey Andreyevich Radul, Barak A Pearlmutter, and Jeffrey Mark Siskind. Automatic Differentiation in Machine Learning: a Survey. *Journal of Machine Learning Research*, 18:1–43, 2018.

[9] Alberto Bemporad and Manfred Morari. Robust model predictive control: A survey. In *Robustness in identification and control*, pages 207–226. Springer, London, 1999.

[10] Dimitri P Bertsekas. *Reinforcement Learning and Optimal Control*. Athena Scientific, 2019.

[11] David Biagioni, Xiangyu Zhang, Dylan Wald, Deepthi Vaidhynathan, Rohit Chintala, Jennifer King, and Ahmed S. Zamzam. PowerGridworld: A Framework for Multi-Agent Reinforcement Learning in Power Systems. *e-Energy 2022 - Proceedings of the 2022 13th ACM International Conference on Future Energy Systems*, pages 565–570, 2022.

[12] F. Blanchini and A. Megretski. Robust state feedback control of LTV systems: Nonlinear is better than linear. *IEEE Tranactions on Automatic Control*, 44(4):2347–2352, 1999.

[13] Franco Blanchini and Stefano Miani. *Set-theoretic methods in control*. Birkhauser, 2015.

[14] Toni Barjas Blanco, Mark Cannon, and Bart De Moor. On efficient computation of low-complexity controlled invariant sets for uncertain linear systems. *International Journal of Control*, 83(7):1339–1346, 2010.

[15] V. S. Borkar. An actor-critic algorithm for constrained Markov decision processes. *Systems and Control Letters*, 54(3):207–213, 2005.

[16] Vivek S. Borkar and Abhay G. Bhatt. Occupation Measures for Controlled Markov Processes: Characterization and Optimality. *The Annals of Probability*, 24(3):1531–1562, 1996.

[17] S. Boyd, Laurent El Ghaoui, Eric Feron, and Venkataramanan Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*, volume 15. Society for Industrial and Applied Mathematics, Philadelphia, 1994.

[18] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2009.

[19] Miguel Calvo-Fullana, Santiago Paternain, Luiz F. O. Chamon, and Alejandro Ribeiro. State Augmented Constrained Reinforcement Learning: Overcoming the Limitations of Learning with Rewards. *arXiv: 2102.11941*, 2021.

[20] Steven Chen, Kelsey Saulnier, Nikolay Atanasov, Daniel D. Lee, Vijay Kumar, George J. Pappas, and Manfred Morari. Approximating Explicit Model Predictive Control Using Constrained Neural Networks. *Proc. Am. Control Conf.*, 2018-June:1520–1527, 2018.

[21] Xin Chen, Guannan Qu, Yujie Tang, Steven Low, and Na Li. Reinforcement learning for decision-making and control in power systems: Tutorial, review, and vision. *arXiv preprint: arXiv 2102.01168*, 2021.

[22] Yu Christine Chen, Xichen Jiang, and Alejandro D. Domínguez-García. Impact of power generation uncertainty on power system static performance. In *NAPS 2011 - 43rd North American Power Symposium*, 2011.

[23] Yu Jia Chen, Deng Kai Chang, and Cheng Zhang. Autonomous Tracking Using a Swarm of UAVs: A Constrained Multi-Agent Reinforcement Learning Approach. *IEEE Transactions on Vehicular Technology*, 69(11):13702–13717, 2020.

[24] Yuxiao Chen, James Anderson, Karanjit Kalsi, Aaron D. Ames, and Steven H. Low. Safety-critical control synthesis for network systems with control barrier functions and assume-guarantee contracts. *IEEE Transactions on Control of Network Systems*, 8(1):487–499, 2021.

[25] Richard Cheng, Gábor Orosz, Richard M. Murray, and Joel W. Burdick. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. In *33rd AAAI Conference on Artificial Intelligence*, pages 3387–3395, 2019.

[26] Yinlam Chow, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone. Risk-constrained reinforcement learning with percentile risk criteria. *Journal of Machine Learning Research*, 18:1–51, 2018.

[27] Wenqi Cui, Yan Jiang, and Baosen Zhang. Reinforcement learning for optimal primary frequency control: A lyapunov approach. *IEEE Transactions on Power Systems*, 38:1676–1688, 3 2023.

[28] Wenqi Cui, Jiayi Li, and Baosen Zhang. Decentralized safe reinforcement learning for inverter-based voltage control. *Electric Power Systems Research*, 211(October 2021):108609, 2022.

[29] Wenqi Cui and Baosen Zhang. Lyapunov-regularized reinforcement learning for power system transient stability. *IEEE Control Systems Letters*, 2021.

[30] Constantinos Daskalakis, Noah Golowich, and Kaiqing Zhang. The complexity of markov equilibrium in stochastic games. In Gergely Neu and Lorenzo Rosasco, editors, *Proceedings of Thirty Sixth Conference on Learning Theory*, volume 195 of *Proceedings of Machine Learning Research*, pages 4180–4234. PMLR, 12–15 Jul 2023.

[31] Ying Ding, Liang Wang, Yongwei Li, and Daoliang Li. Model predictive control and its application in agriculture: A review. *Computers and Electronics in Agriculture*, 151:104–117, 2018.

[32] Alexander Domahidi, Melanie N. Zeilinger, Manfred Morari, and Colin N. Jones. Learning a feasible and stabilizing explicit model predictive control law by robust optimization. In *Proc. IEEE Conf. Decision Control*, pages 513–519. IEEE, 2011.

[33] Priya L. Donti, Melrose Roderick, Mahyar Fazlyab, and J. Zico Kolter. Enforcing robust control guarantees within neural network policies. In *International Conference on Learning Representations*, pages 1–26, 2021.

[34] Florian Dörfler, Mihailo R Jovanović, Michael Chertkov, and Francesco Bullo. Sparsity-promoting optimal wide-area control of power networks. *IEEE Transactions on Power Systems*, 29(5):2281–2291, 2014.

[35] Jan Drgona, Karol Kis, Aaron Tuor, Draguna Vrabie, and Martin Klauco. Differentiable Predictive Control: Deep Learning Alternative to Explicit Model Predictive Control for Unknown Nonlinear Systems. *arXiv:2011.03699*, 2020.

[36] Ahmed El-Guindy, Yu Christine Chen, and Matthias Althoff. Compositional transient stability analysis of power systems via the computation of reachable sets. *Proceedings of the American Control Conference*, pages 2536–2543, 2017.

[37] Ahmed El-Guindy, Konstantin Schaab, Bastian Schurmann, Olaf Stursberg, and Matthias Althoff. Formal LPV control for transient stability of power systems. *IEEE Power and Energy Society General Meeting*, 2018-Janua:1–5, 2018.

[38] Marcello Farina, Luca Giulioni, and Riccardo Scattolini. Stochastic linear Model Predictive Control with chance constraints - A review. *J. Process Control*, 44:53–67, 2016.

[39] Carlo Filippi. An algorithm for approximate multiparametric linear programming. *Journal of Optimization Theory and Applications*, 120(1):73–95, 2004.

[40] Jakob N. Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. *32nd AAAI Conference on Artificial Intelligence*, pages 2974–2982, 2018.

[41] Javier García and Fernando Fernández. A Comprehensive Survey on Safe Reinforcement Learning. *Journal of Machine Learning Research*, 16:1437–1480, 2015.

[42] Mevludin Glavic. (Deep) Reinforcement learning for electric power system control and related problems: A short review and perspectives. *Annual Reviews in Control*, 48:22–35, 2019.

[43] Alexandra Grancharova and Tor A. Johansen. Computation, approximation and stability of explicit feedback min-max nonlinear model predictive control. *Automatica*, 45(5):1134–1143, 2009.

[44] Sebastien Gros, Mario Zanon, and Alberto Bemporad. Safe reinforcement learning via projection on a safe set: How to achieve optimality? *IFAC-PapersOnLine*, 53(2):8076–8081, 2020.

[45] Joao P Hespanha. *Linear systems theory*. Princeton university press, 2018.

[46] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[47] D. Hrovat, S. Di Cairano, H. E. Tseng, and I. V. Kolmanovsky. The development of Model Predictive Control in automotive industry: A survey. *Proc. IEEE Int. Conf. on Control Applications*, pages 295–302, 2012.

[48] Qiuhua Huang, Renke Huang, Weituo Hao, Jie Tan, Rui Fan, and Zhenyu Huang. Adaptive power system emergency control using deep reinforcement learning. *IEEE Transactions on Smart Grid*, 11:1171–1182, 3 2020.

[49] Tor A. Johansen. Approximate explicit receding horizon control of constrained nonlinear systems. *Automatica*, 40(2):293–300, 2004.

[50] C. N. Jones, M. Barić, and M. Morari. Multiparametric linear programming with applications to control. *European Journal of Control*, 13(2-3):152–170, 2007.

[51] Hanna Krasowski, Jakob Thumm, Marlon Müller, Lukas Schäfer, Xiao Wang, and Matthias Althoff. Provably safe reinforcement learning: Conceptual analysis, survey, and benchmarking. *Transactions on Machine Learning Research*, 2023.

[52] Benjamin Kroposki, Brian Johnson, Yingchen Zhang, Vahan Gevorgian, Paul Denholm, Bri Mathias Hodge, and Bryan Hannegan. Achieving a 100% Renewable Grid: Operating Electric Power Systems with Extremely High Levels of Variable Renewable Energy. *IEEE Power and Energy Magazine*, 15(2):61–73, 2017.

[53] P. Kundur, N.J. Balu, and M.G. Lauby. *Power system stability and control*. McGraw-Hill, New York, 7 edition, 1994.

[54] W. Langson, I. Chryssochoos, S. V. Raković, and D. Q. Mayne. Robust model predictive control using tubes. *Automatica*, 40(1):125–133, 2004.

[55] Abdul Latif, SM Suhail Hussain, Dulal Chandra Das, and Taha Selim Ustun. State-of-the-art of controllers and soft computing techniques for regulated load frequency management of single/multi-area traditional and renewable energy based power systems. *Applied Energy*, 266:114858, 2020.

[56] Donghwan Lee, Hyungjin Yoon, and Naira Hovakimyan. Primal-Dual Algorithm for Distributed Reinforcement Learning: Distributed GTD. *Proceedings of the IEEE Conference on Decision and Control*, 2018-Decem(Cdc):1967–1972, 2018.

[57] Wonyeol Lee, Hangyeol Yu, Xavier Rival, and Hongseok Yang. On correctness of automatic differentiation for non-differentiable functions. In *Advances in Neural Information Processing Systems*, 2020.

[58] Meiyi Li, Soheil Kolouri, and Javad Mohammadi. Learning to solve optimization problems with hard linear constraints. *IEEE Access*, 11, 2023.

[59] Wenhao Li, Bo Jin, Xiangfeng Wang, Junchi Yan, and Hongyuan Zha. F2A2: Flexible Fully-decentralized Approximate Actor-critic for Cooperative Multi-agent Reinforcement Learning. *arXiv: 2004.11145*, pages 1–42, 2020.

[60] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, 2016.

[61] Chengyuan Liu and Imad M. Jaimoukha. The computation of full-complexity polytopic robust control invariant sets. *Proceedings of the IEEE Conference on Decision and Control*, 54rd IEEE(Cdc):6233–6238, 2015.

[62] Chengyuan Liu, Furqan Tahir, and Imad M. Jaimoukha. Full-complexity polytopic robust control invariant sets for uncertain linear discrete-time systems. *International Journal of Robust and Nonlinear Control*, 29(11):3587–3605, 2019.

[63] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In *31st Conference on Neural Information Processing Systems*, 6 2017.

[64] Songtao Lu, Kaiqing Zhang, Tianyi Chen, Tamer Başar, and Lior Horesh. Decentralized Policy Gradient Descent Ascent for Safe Multi-Agent Reinforcement Learning. *35th AAAI Conference on Artificial Intelligence, AAAI 2021*, 10A:8767–8775, 2021.

[65] John Lygeros. On reachability and minimum cost optimal control. *Automatica*, 40(6):917–927, 2004.

[66] Haitong Ma, Jianyu Chen, Shengbo Eben, Ziyu Lin, Yang Guan, Yangang Ren, and Sifa Zheng. Model-based Constrained Reinforcement Learning using Generalized Control Barrier Function. *IEEE International Conference on Intelligent Robots and Systems*, pages 4552–4559, 2021.

[67] Jan Machowski, Janusz W. Bialek, and James R Bumby. *Power System Dynamics: Stability and Control*. Wiley, 2 edition, 2008.

[68] E. T. Maddalena, C. G.S. da Moraes, G. Waltrich, and C. N. Jones. A neural network architecture to learn explicit MPC controllers from data. *IFAC-PapersOnLine*, 53(2):11362–11367, 2020.

[69] John N. Maidens, Shahab Kaynama, Ian M. Mitchell, Meeko M.K. Oishi, and Guy A. Dumont. Lagrangian methods for approximating the viability kernel in high-dimensional systems. *Automatica*, 49(7):2017–2029, 2013.

[70] Ali Mesbah. Stochastic model predictive control: An overview and perspectives for future research. *IEEE Control Systems Magazine*, 36(6):30–44, 2016.

[71] Ian M Mitchell, Alexandre M Bayen, and Claire J Tomlin. A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games. *IEEE Transactions on automatic control*, 50(7):947–957, 2005.

[72] Miguel Molina-Solana, María Ros, M. Dolores Ruiz, Juan Gómez-Romero, and M. J. Martin-Bautista. Data science for building energy management: A review. *Renewable and Sustainable Energy Reviews*, 70(August 2015):598–609, 2017.

[73] T Nguyen and F Jabbari. Disturbance Attenuation for Systems with Input Saturation: An LMI Approach. Technical Report 4, Department of Mechanical and Aerospace Engineering, University of California, Irvine, 1999.

[74] Jon O'Sullivan, Alan Rogers, Damian Flynn, Paul Smith, Alan Mullane, and Mark O'Malley. Studying the maximum instantaneous non-synchronous generation in an Island system-frequency stability challenges in Ireland. *IEEE Transactions on Power Systems*, 29(6):2943–2951, 2014.

[75] Georgios Papoudakis, Filippos Christianos, Lukas Schäfer, and Stefano V. Albrecht. Benchmarking Multi-Agent Deep Reinforcement Learning Algorithms in Cooperative Tasks. In *35th Conference on Neural Information Processing Systems*, 2020.

[76] T. Parisini and R. Zoppoli. A Receding-Horizon Regulator for Nonlinear Systems and a Neural Approximation. *Automatica*, 31(10):1443–1451, 1995.

[77] P. Parnika, Raghuram Bharadwaj Diddigi, Sai Koti Reddy Danda, and Shalabh Bhatnagar. Attention actor-critic algorithm for multi-agent constrained co-operative reinforcement learning. *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, 3(Aamas 2021):1604–1606, 2021.

[78] Santiago Paternain, Miguel Calvo-Fullana, Luiz F.O. Chamon, and Alejandro Ribeiro. Safe Policies for Reinforcement Learning via Primal-Dual Methods. *IEEE Transactions on Automatic Control*, 9286(c):1–16, 2022.

[79] Santiago Paternain, Luiz F.O. Chamon, Miguel Calvo-Fullana, and Alejandro Ribeiro. Constrained reinforcement learning has zero duality gap. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[80] Aisling Pigott, Constance Crozier, Kyri Baker, and Zoltan Nagy. Gridlearn: Multiagent reinforcement learning for grid-aware building energy management. *Electric Power Systems Research*, 213, 12 2022.

[81] Warren B. Powell. *Reinforcement Learning and Stochastic Optimization: A Unified Framework for Sequential Decisions*. Wiley, 2022.

[82] James B. Rawlings, David Q. Mayne, and Moritz M. Diehl. *Model Predictive Control: Theory and Design*. Nob Hill, Santa Barbara, CA, 2 edition, 2019.

[83] R. Tyrrell Rockafellar and Stanislav Uryasev. Optimization of Conditional Value-at-Risk. *Journal of Risk*, 2:21–42, 2000.

[84] R. Tyrrell Rockafellar and Stanislav Uryasev. Conditional value-at-risk for general loss distributions. *Journal of Banking and Finance*, 26(7):1443–1471, 2002.

[85] Sadra Sadraddini and Russ Tedrake. Linear Encodings for Polytope Containment Problems. *Proc. IEEE Conf. Decision Control*, pages 4367–4372, 2019.

[86] M. Bahadır Saltık, Leyla Özkan, Jobert H.A. Ludlage, Siep Weiland, and Paul M.J. Van den Hof. An outlook on robust model predictive control algorithms: Reflections on performance and computational aspects. *J. Process Control*, 61:77–102, 2018.

[87] Yuanyuan Shi, Guannan Qu, Steven Low, Anima Anandkumar, and Adam Wierman. Stability constrained reinforcement learning for real-time voltage control. In *Proceedings of the American Control Conference*, volume 2022-June, 2022.

[88] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. *31st International Conference on Machine Learning, ICML 2014*, 1:605–619, 2014.

[89] Jørgen Spjøtvold, Petter Tøndel, and Tor A. Johansen. A method for obtaining continuous solutions to multiparametric linear programs. *IFAC Proc. Volumes (IFAC-PapersOnline)*, 38(1):253–258, 2005.

[90] Mandyam Dhati Srinath, PK Rajasekaran, and Ramanarayanan Viswanathan. *Introduction to statistical signal processing with applications*. Prentice-Hall, Inc., 1995.

[91] Daniel Tabas and Baosen Zhang. Computationally Efficient Safe Reinforcement Learning for Power Systems. *arXiv:2110.10333*, 2021.

[92] Daniel Tabas and Baosen Zhang. Computationally Efficient Safe Reinforcement Learning for Power Systems. In *Proceedings of the American Control Conference*, pages 3303–3310. American Automatic Control Council, 2022.

[93] Furqan Tahir. Efficient computation of Robust Positively Invariant sets with linear state-feedback gain as a variable of optimization. *7th International Conference on Electrical Engineering, Computing Science and Automatic Control, 2010*, pages 199–204, 2010.

[94] Chen Tessler, Daniel J. Mankowitz, and Shie Mannor. Reward constrained policy optimization. In *7th International Conference on Learning Representations, ICLR 2019*, 5 2019.

[95] Kim Peter Wabersich and Melanie N. Zeilinger. A predictive safety filter for learning-based control of constrained nonlinear dynamical systems. *Automatica*, 129:109597, 2021.

[96] Yang Wang and Stephen Boyd. Fast model predictive control using online optimization. *IEEE Trans. Control Syst. Technol.*, 18(2):267–278, 2010.

[97] Hongyu Wu, Ibrahim Krad, Anthony Florita, Bri Mathias Hodge, Eduardo Ibanez, Jie Zhang, and Erik Ela. Stochastic Multi-Timescale Power System Operations with Variable Wind Generation. *IEEE Transactions on Power Systems*, 32(5):3325–3337, 2017.

[98] Bolun Xu, Yuanyuan Shi, Daniel S Kirschen, and Baosen Zhang. Optimal battery participation in frequency regulation markets. *IEEE Transactions on Power Systems*, 33(6):6715–6725, 2018.

[99] Ziming Yan and Yan Xu. Data-driven load frequency control for stochastic power systems: A deep reinforcement learning method with continuous action search. *IEEE Transactions on Power Systems*, 34:1653–1656, 3 2019.

[100] Shuo Yang, Shaoru Chen, Victor M. Preciado, and Rahul Mangharam. Differentiable safe controller design through control barrier functions. *IEEE Control Systems Letters*, 7, 2023.

[101] Haeun Yoo, Victor M. Zavala, and Jay H. Lee. A Dynamic Penalty Function Approach for Constraint-Handling in Reinforcement Learning. *IFAC-PapersOnLine*, 54(3):487–491, 2021.

[102] Ming Yu, Zhuoran Yang, Mladen Kolar, and Zhaoran Wang. Convergent policy optimization for safe reinforcement learning. *Advances in Neural Information Processing Systems*, 32(NeurIPS), 2019.

[103] Melanie Nicole Zeilinger, Colin Neil Jones, and Manfred Morari. Real-time suboptimal model predictive control using a combination of explicit MPC and online optimization. *IEEE Trans. Autom. Control*, 56(7):1524–1534, 2011.

[104] Ling Zhang, Daniel Tabas, and Baosen Zhang. Convex restriction of feasible sets for ac radial networks. *arXiv:2310.00549*, 9 2023.

[105] Ling Zhang, Daniel Tabas, and Baosen Zhang. An efficient learning-based solver for two-stage dc optimal power flow with feasibility guarantees. *arXiv:2304.01409*, 2023.

[106] Yichen Zhang, Yan Li, Kevin Tomsovic, Seddik Djouadi, and Meng Yue. Review on Set-Theoretic Methods for Safety Verification and Control of Power System. *IET Energy Systems Integration*, pages 2–12, 2020.

[107] Zidong Zhang, Dongxia Zhang, and Robert C Qiu. Deep reinforcement learning for power system: An overview. *CSEE Journal of Power and Energy Systems*, 6(1):213–225, 2019.

[108] Changhong Zhao, Ufuk Topcu, Na Li, and Steven Low. Design and stability of load-side primary frequency control in power systems. *IEEE Transactions on Automatic Control*, 59(5):1177–1189, 2014.

[109] Liyuan Zheng, Yuanyuan Shi, Lillian J. Ratliff, and Baosen Zhang. Safe reinforcement learning of control-affine systems with vertex networks. *arXiv preprint: arXiv 2003.09488v1*, 2020.

[110] Wei Zhou, Dong Chen, Jun Yan, Zhaojian Li, Huilin Yin, and Wanchen Ge. Multi-agent reinforcement learning for cooperative lane changing of connected and autonomous vehicles in mixed traffic. *Autonomous Intelligent Systems*, 2(1), 2022.

Appendix A

ADDITIONAL RESULTS FOR CHAPTER 2[1]

A.1 Derivation of (2.11)

Let $\mathcal{Q} = \{x \in \mathbb{R}^n \mid Fx \leq g\}$ be a C-set, where $F \in \mathbb{R}^{r\times n}$, $g \in \mathbb{R}^r$, F_i^T denotes the ith row of F, and g_i denotes the ith element of g. The gauge function $\gamma_{\mathcal{Q}}(v)$ is computed as follows.

$$\gamma_{\mathcal{Q}}(v) = \inf\{\lambda \geq 0 \mid v \in \lambda\mathcal{Q}\} \tag{A.1}$$

$$= \inf\{\lambda \geq 0 \mid \frac{1}{\lambda}F_i^T v \leq g_i, i = 1, \ldots, r\} \tag{A.2}$$

$$= \inf\{\lambda \geq 0 \mid \lambda \geq \frac{F_i^T v}{g_i}, i = 1, \ldots, r\} \tag{A.3}$$

$$= \max\{0, \max_i\{\frac{F_i^T v}{g_i}\}\}. \tag{A.4}$$

We now argue that $\max_i\{\frac{F_i^T v}{g_i}\} \geq 0$. If $F_i^T v < 0$ for all i, then $\mathcal{Q}$ is unbounded in the direction of v and $\mathcal{Q}$ cannot be a C-set, a contradiction. Further, since $0 \in$ **int** $(\mathcal{Q})$, it must hold that $g_i > 0$ for all i. Therefore, there exists i such that $\frac{F_i^T v}{g_i} \geq 0$. □

A.2 Additional lemmas

The following lemma will be used in the proofs of Lemma 2.2 and Theorem 2.1.

Lemma A.1. *Under the assumptions of Theorem 2.1, the safe action set $\Omega(x_t)$ is a polytope for all $x_t \in \mathcal{S}$.*

[1]Adapted from D. Tabas and B. Zhang, "Computationally Efficient Safe Reinforcement Learning for Power Systems," 2022 American Control Conference (ACC), Atlanta, GA, USA, 2022, pp. 3303-3310.

Proof. Starting from (2.1), (2.5), and (2.7), the safe action set is

$$\Omega(x_t) = \{u_t \in \mathcal{U} \mid -\bar{s} \leq V_s x_{t+1} \leq \bar{s},\ \forall\ d_t \in \mathcal{D}\} \tag{A.5}$$

$$\begin{aligned} &= \{u_t \in \mathcal{U} \mid -\bar{s}_i - \min_{d\in\mathcal{D}} V_s^{(i)T} E d \\ &\qquad \leq V_s^{(i)T}(Ax_t + Bu_t) \\ &\qquad \leq \bar{s}_i - \max_{d\in\mathcal{D}} V_s^{(i)T} E d, \\ &\qquad \forall\ i = 1, \ldots, r\} \end{aligned} \tag{A.6}$$

where $\bar{s}_i$ is the ith element of $\bar{s}$ and $V_s^{(i)T}$ is the ith row of V_s. Since the min and max terms evaluate to constant scalars for each i, and since x_t is fixed, (A.6) is a set of linear inequalities in u_t, making $\Omega(x_t)$ a polytope [13]. □

A.3 Proof of Lemma 2.1

We will prove the more general case in which $\mathbb{B}_\infty$ is replaced by any polytopic C-set. Let $\mathcal{P}$ and $\mathcal{Q}$ be two polytopic C-sets, and define the gauge map from $\mathcal{P}$ to $\mathcal{Q}$ as $G(v|\mathcal{P}, \mathcal{Q}) = \frac{\gamma_\mathcal{P}(v)}{\gamma_\mathcal{Q}(v)} \cdot v$. We will prove that G is a bijection from $\mathcal{P}$ to $\mathcal{Q}$. The proof is then completed by noting that $\gamma_{\mathbb{B}_\infty}$ is the same as the ∞-norm.

To prove injectivity, we fix $v_1, v_2 \in \mathcal{P}$ and show that if $G(v_1|\mathcal{P}, \mathcal{Q}) = G(v_2|\mathcal{P}, \mathcal{Q})$ then $v_1 = v_2$. Assume $G(v_1|\mathcal{P}, \mathcal{Q}) = G(v_2|\mathcal{P}, \mathcal{Q})$. Then v_1 and v_2 must be nonnegative scalar multiples of each other, i.e. $v_2 = \beta v_1$ for some $\beta \geq 0$. Making this substitution and applying positive homogeneity of the gauge function [13] yields

$$G(v_2|\mathcal{P}, \mathcal{Q}) = \frac{\gamma_\mathcal{P}(v_2)}{\gamma_\mathcal{Q}(v_2)} v_2 = \frac{\gamma_\mathcal{P}(v_1)}{\gamma_\mathcal{Q}(v_1)} v_2. \tag{A.7}$$

Noting that $G(v_1|\mathcal{P}, \mathcal{Q}) = \frac{\gamma_\mathcal{P}(v_1)}{\gamma_\mathcal{Q}(v_1)} v_1$, we conclude that $\beta = 1$, thus $v_1 = v_2$.

To prove surjectivity, fix $w \in \mathcal{Q}$. We must find $v \in \mathcal{P}$ such that $G(v|\mathcal{P}, \mathcal{Q}) = w$. Since $\mathcal{P}$ and $\mathcal{Q}$ are C-sets, each set contains an open ball around the origin, thus $\mathcal{P}$ and $\mathcal{Q}$ each contain all directions at sufficiently small magnitude. Choose v in the

same direction as w such that $\gamma_{\mathcal{P}}(v) = \gamma_{\mathcal{Q}}(w)$. Since $w \in \mathcal{Q}$, $v \in \mathcal{P}$. Then, we have

$$G(v|\mathcal{P}, \mathcal{Q}) = \frac{\gamma_{\mathcal{P}}(v)}{\gamma_{\mathcal{Q}}(v)} v \tag{A.8}$$

$$= \frac{\gamma_{\mathcal{Q}}(w)}{\gamma_{\mathcal{Q}}(v)} v. \tag{A.9}$$

Since v and w are in the same direction, $\frac{v}{\gamma_{\mathcal{Q}}(v)} = \frac{w}{\gamma_{\mathcal{Q}}(w)}$. Making this substitution completes the proof. □

A.4 Proof of Lemma 2.2

Let **int** and **bd** denote the interior and boundary of a set, and rewrite (A.6) as $\Omega(x) = \{u \in \mathbb{R}^m \mid Hu \leq h, F(Ax + Bu) \leq g\}$. Fix $x \in \textbf{int}\ (\mathcal{S})$ and let $u^* = Kx$. By Lemma A.1, $\Omega(x)$ is convex and compact. To fulfill the properties of a C-set, it remains to show that $u^* \in \textbf{int}\ (\Omega(x))$. Since $\pi_0 \in \Pi$, $u^* \in \Omega(x)$. Assume for the sake of contradiction that $u^* \in \textbf{bd}(\Omega(x))$. Then either $F_i^T(A + BK)x = g_i$ or $H_j^T Kx = h_j$ for some i or j, where the subscript denotes a row index. Suppose without loss of generality that the former holds, i.e. $F_i^T(A+BK)x = g_i$ for some i. Since $x \in \textbf{int}\ (\mathcal{S})$, there exists $\varepsilon \in (0, 1)$ and $\alpha = [1 + \varepsilon \cdot \textbf{sign}(g_i)]$ such that $y = \alpha x$ is also in **int** $(\mathcal{S})$. The set $\Omega(y)$ is contained in the halfspace $\{u \mid F_i^T(Ay + Bu) \leq g_i\}$. Evaluating this inequality with $u = Ky$, we have $F_i^T(A + BK)y = \alpha F_i^T(A + BK)x = \alpha g_i > g_i$, thus $Ky \notin \Omega(y)$ even though $y \in \mathcal{S}$, contradicting the assumption that $\pi_0 \in \Pi$. We conclude that $u^* \notin \textbf{bd}(\Omega(x))$. Since $u^* \in \Omega(x)$, u^* must be an element of **int** $(\Omega(x))$. □

A.5 Proof of Theorem 2.1

1. It suffices to show that the gauge map from $\mathbb{B}_\infty$ to $\hat{\Omega}_t$ is well-defined on **int** $(\mathcal{S})$. This is a direct result of Lemma 2.2.

2. By Lemmas 2.2 and A.1, $\hat{\Omega}_t$ is a polytopic C-set. By (2.11), $\gamma_{\hat{\Omega}_t}$ (and π_θ) can be computed in closed form.

3. Let $\mathcal{P} = \{\mathbf{z} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{z} \leq \mathbf{b}\}$ and $\mathcal{Q} = \{\mathbf{z} \in \mathbb{R}^n \mid \mathbf{C}\mathbf{z} \leq \mathbf{d}\}$ with $\mathbf{A} \in \mathbb{R}^{m\times n}, \mathbf{b} \in \mathbb{R}^m_{++}, \mathbf{C} \in \mathbb{R}^{k\times n}$, and $\mathbf{d} \in \mathbb{R}^k_{++}$. Let $\mathcal{A}_{ij}$ be the polytope described as $\{\mathbf{z} \in \mathcal{P} \mid i \in \arg\max_{i=1,\cdots,m} \frac{\mathbf{a}_i^T\mathbf{z}}{b_i}, j \in \arg\max_{j=1,\cdots,k} \frac{\mathbf{c}_j^T\mathbf{z}}{d_j}\}$. The set $\{\mathcal{A}_{ij} \mid i \in 1,\cdots,m,\ j \in 1,\cdots,k\}$ forms a polyhedral partition of $\mathcal{P}$, and the gauge map is an analytic function on the interior of each $\mathcal{A}_{ij}$ except when $\mathbf{c}_j^T\mathbf{z} = 0$ or $\mathbf{z} = 0$. Specifically, the gauge map on the interior of $\mathcal{A}_{ij} \subseteq \mathcal{P}$ can be written as $G(\mathbf{z} \mid \mathcal{P}, \mathcal{Q}) = \frac{\mathbf{a}_i^T\mathbf{z}/b_i}{\mathbf{c}_j^T\mathbf{z}/d_j}\mathbf{z}$.

   For any $j \in 1,\cdots,k$, $\mathbf{c}_j^T\mathbf{z} = 0$ if and only if $\mathbf{z} = 0$: since Q forms a full-dimensional and bounded polytope, $\mathbf{C}$ must be full-rank and tall ($k > n$). Thus, $\mathbf{C}\mathbf{z} = 0$ if and only if $\mathbf{z} = 0$.

   We can now justify the choice $G(0 \mid \mathcal{P}, \mathcal{Q}) := 0$ as follows. Let $\mathbf{z} = \alpha\mathbf{h}$ for some $\alpha > 0$ and $\mathbf{h} \in \mathbb{R}^n\backslash\{0\}$. There exist some (i, j) and sufficiently small $\varepsilon > 0$ such that $\mathbf{z} \in \mathcal{A}_{ij}\ \forall\ \alpha \in (0, \varepsilon)$. The limit of $\frac{\mathbf{a}_i^T\mathbf{z}/b_i}{\mathbf{c}_j^T\mathbf{z}/d_j}\mathbf{z}$ as $\alpha \to 0$ evaluates to $0 \in \mathbb{R}^n$.

   By the above analysis, the gauge map is *piecewise analytic under analytic partition* (PAP) on $\mathcal{P}$ which implies desirable properties for automatic differentiation [57]. Specifically, PAP functions can be composed with one another (they obey a chain rule), they are differentiable almost everywhere (except possibly on a set of measure zero), and standard automatic differentiation tools will compute the derivatives at all points where the function is differentiable[2].

4. This is due to the fact that ψ_θ is a universal function approximator for functions from $\mathcal{S}$ to $\mathbb{B}_\infty$ [46]. By (2.13) and Lemma 2.1, π_θ approximates any function in Π.

---

[2]Adapted from Zhang, L., Tabas, D., and Zhang, B. (2023). An Efficient Learning-Based Solver for Two-Stage DC Optimal Power Flow with Feasibility Guarantees. *arXiv: 2304.01409*

Appendix B

ADDITIONAL RESULTS FOR CHAPTER 4[1]

B.1 Theoretical results

B.1.1 Proof of Proposition 4.1

Applying the definition of μ_γ^π, we have $\int_{\mathcal{X}} \mu_\gamma^\pi(x)dx = \int_{\mathcal{X}} \Gamma_{t=0}^{\infty} p_t^\pi(x)dx$. Using the Dominated Convergence Theorem, we can exchange the order of the sum and integral. Each individual p_t^π integrates to 1. The geometric sum property ensures that the resulting expression evaluates to 1.

B.1.2 Proof of Proposition 4.2

1. By definition, we have $\lim_{\gamma\to 0^+} \mu_\gamma^\pi(x) = \lim_{\gamma\to 0^+} \Gamma_{t=0}^{\infty} p_t^\pi(x)$. Using Tannery's theorem, we can exchange the order of the limit and the infinite sum. The zeroth term in the sum evaluates to $p_0(x)$ and all other terms evaluate to 0.

2. Assume $\lim_{t\to\infty} p_t^\pi$ exists, and denote it p_∞^π. Using the triangle inequality, we have

$$|\mu_\gamma^\pi(x) - p_\infty^\pi(x)| \leq \sum_{t=0}^{\infty} |p_t^\pi(x) - p_\infty^\pi(x)| \tag{B.1}$$

$$= \sum_{t=0}^{N} |p_t^\pi(x) - p_\infty^\pi(x)| + \sum_{t=N+1}^{\infty} |p_t^\pi(x) - p_\infty^\pi(x)| \tag{B.2}$$

for some $N \in \mathbb{N}$. Since $p_t^\pi(x) \to p_\infty^\pi(x)$, we can choose N large enough to make the second term in (B.2) arbitrarily small. Then, using boundedness of p_t^π for all t, we can take $\gamma \to 1^-$ to make the first term arbitrarily small.

---

[1]Adapted from Tabas, D., Zamzam, A. S., and Zhang, B. (2023). Interpreting Primal-Dual Algorithms for Constrained MARL. *arXiv: 2211.16069*.

B.1.3 Proof of Proposition 4.3

By the geometric sum property, we have $T_2(\gamma, \varepsilon) = \min\{K \in \mathbb{N} : \prod_{t=0}^{K-1}[1] \geq 1-\varepsilon\} = \min\{K \in \mathbb{N} : 1-\gamma^K \geq 1-\varepsilon\} = \min\{K \in \mathbb{N} : K \geq \frac{\log \varepsilon}{\log \gamma}\} = \left\lceil \frac{\log \varepsilon}{\log \gamma} \right\rceil$. The termination time follows a geometric distribution with parameter $(1-\gamma)$, and thus has expected value $\frac{1}{1-\gamma}$. Setting $T_2(\gamma, \varepsilon) = T_1(\gamma)$ and solving for ε (ignoring the integer constraint) yields $\varepsilon = \gamma^{\frac{1}{1-\gamma}}$. Finally, taking $\lim_{\gamma \to 1} \gamma^{\frac{1}{1-\gamma}}$ yields $\frac{1}{e}$.

B.1.4 Proof of Proposition 4.6

Let $x \sim \mu_\gamma^\pi$, $x' \sim f^\pi(x)$, $\bar{c} = \mathbb{E}_{x \sim \mu_\gamma^\pi}[C(x)]$, and $\Sigma_C^2 = \mathbb{E}_{x \sim \mu_\gamma^\pi}[(\bar{c} - C(x))(\bar{c} - C(x))^T]$. Suppose λ is randomly distributed with mean $\bar{\lambda}$ and variance Σ_λ^2. For any value function approximator $\hat{V}_\pi^i$, assume λ and $\hat{V}_\pi^i$ are independent. Let $\eta = \begin{bmatrix} 1 & \lambda^T \end{bmatrix}^T$, $d = \begin{bmatrix} R^i(x) & C(x)^T \end{bmatrix}^T$, $\hat{V}_\pi^i : \mathcal{X} \to \mathbb{R}$, $\hat{V}_{R,\pi}^i : \mathcal{X} \to \mathbb{R}$, and $\hat{V}_{C,\pi} : \mathcal{X} \to \mathbb{R}^m$. Let $\mathcal{D}$ be a dataset of trajectories sampled from $\mathcal{M}$ that is used to train $\hat{V}_\pi^i$, $\hat{V}_{R,\pi}^i$, and $\hat{V}_{C,\pi}$. The mean square temporal difference error achieved by using a generic value function is

$$MSTDE_1 = \mathbb{E}_{x,x',\lambda,\mathcal{D}}[(\eta^T d + \gamma \hat{V}_\pi^i(x') - \hat{V}_\pi^i(x))^2] \tag{B.3}$$

while the error achieved using the structured value function is

$$MSTDE_2 = \mathbb{E}_{x,x',\mathcal{D}}[(\eta^T d + \gamma[\hat{V}_{R,\pi}^i(x') - \lambda^T \hat{V}_{C,\pi}(x')] - [\hat{V}_{R,\pi}^i(x) - \lambda^T \hat{V}_{C,\pi}(x)])^2]. \tag{B.4}$$

Note that in (B.4) we do not take the expectation over λ since the dual variables are made available to this function approximator.

Begin with the states and dual variables fixed at $(\bar{x}, \bar{x}', \bar{\lambda})$. Let $\hat{g}(\bar{x}, \bar{x}') = \begin{bmatrix} \hat{V}_{R,\pi}^i(\bar{x}) & \hat{V}_{C,\pi}(\bar{x})^T \end{bmatrix}^T - \gamma \begin{bmatrix} \hat{V}_{R,\pi}^i(\bar{x}') & \hat{V}_{C,\pi}(\bar{x}')^T \end{bmatrix}^T$ and $\hat{h}(\bar{x}, \bar{x}') = \hat{V}_\pi^i(\bar{x}) - \gamma \hat{V}_\pi^i(\bar{x}')$. Then, suppressing the arguments $(\bar{x}, \bar{x}')$ and setting $\bar{\eta} = \begin{bmatrix} 1 & -\bar{\lambda}^T \end{bmatrix}^T$, we can write the squared temporal difference error at $(\bar{x}, \bar{x}', \bar{\lambda})$ as

$$STDE_1(\bar{\eta}) = \mathbb{E}_{\mathcal{D}}[(\bar{\eta}^T d - \hat{h})^2], \tag{B.5}$$

$$STDE_2(\bar{\eta}) = \mathbb{E}_{\mathcal{D}}[(\bar{\eta}^T d - \bar{\eta}^T \hat{g})^2]. \tag{B.6}$$

The loss function used to train $\hat{V}^i_{R,\pi}$ and $\hat{V}_{C,\pi}$ is

$$\mathbb{E}_{\mathcal{D}}[\|d - \hat{g}\|^2]. \tag{B.7}$$

Since d is a deterministic function of x, (B.7) can be decomposed into bias and variance terms:

$$\begin{aligned}
\mathbb{E}_{\mathcal{D}}[\|d - \hat{g}\|^2] &= \mathbb{E}_{\mathcal{D}}[\sum_{k=0}^{m}(d_k - \hat{g}_k)^2] && \text{(B.8)}\\
&= \sum_{k=0}^{m}\mathbb{E}_{\mathcal{D}}[(d_k - \hat{g}_k)^2] && \text{(B.9)}\\
&= \sum_{k=0}^{m}[(d_k - \mathbb{E}_{\mathcal{D}}\hat{g}_k)^2 + \mathbb{E}_{\mathcal{D}}[(\hat{g} - \mathbb{E}_{\mathcal{D}}\hat{g})^2]] && \text{(B.10)}\\
&:= \sum_{k=0}^{m}[b_k^2 + \sigma_k^2] && \text{(B.11)}\\
&:= \mathrm{Tr}[bb^T + \Sigma^2] && \text{(B.12)}
\end{aligned}$$

where $k = 0$ corresponds to the reward signal and $k = 1, \ldots, m$ corresponds to the cost signals.

Following a similar line of reasoning, we can use (B.12) to rewrite (B.6) as

$$STDE_2(\bar{\eta}) = \mathrm{Tr}[(bb^T + \Sigma^2)(\bar{\eta}\bar{\eta}^T)]. \tag{B.13}$$

For the sake of argument, we assume that $\hat{g}$ and $\hat{h}$ achieve the same performance at (x, x', λ), that is,

$$STDE_1(\bar{\eta}) = STDE_2(\bar{\eta}) = \mathrm{Tr}[(bb^T + \Sigma^2)(\bar{\eta}\bar{\eta}^T)] \tag{B.14}$$

where $\mathrm{Tr}[(bb^T)(\bar{\eta}\bar{\eta}^T)]$ and $\mathrm{Tr}[\Sigma^2\bar{\eta}\bar{\eta}^T]$ reflect the bias squared and variance terms, respectively. How do $STDE_1$ and $STDE_2$ change when λ is allowed to vary? Using the generic estimator, the noise in λ will introduce some amount of *irreducible error* into $STDE_1$. On the other hand, using $\lambda = \bar{\lambda} + \Delta\lambda$ in our proposed estimator will change the bias and variance terms in $STDE_2$ while the irreducible error remains at

zero (since there is no uncertainty when $\Delta\lambda$ is known). Setting $\Delta\eta = \begin{bmatrix} 0 & -\Delta\lambda^T \end{bmatrix}^T$, the temporal difference errors at $(\bar{x}, \bar{x}', \bar{\lambda} + \Delta\lambda)$ are

$$STDE_1(\bar{\eta} + \Delta\eta) = \mathrm{Tr}[(bb^T + \Sigma^2)(\bar{\eta}\bar{\eta}^T)] + (\Delta\eta^T d)^2, \tag{B.15}$$

$$STDE_2(\bar{\eta} + \Delta\eta) = \mathrm{Tr}[(bb^T + \Sigma^2)((\bar{\eta} + \Delta\eta)(\bar{\eta} + \Delta\eta)^T)]. \tag{B.16}$$

Taking the expectation over $\Delta\lambda$ which has a mean of zero and a variance of Σ_λ^2, and setting $\Sigma_\eta^2 = \begin{bmatrix} 0 & 0 \\ 0 & \Sigma_\lambda^2 \end{bmatrix}$, yields

$$\mathbb{E}_{\Delta\lambda}[STDE_1(\bar{\eta} + \Delta\eta) - STDE_2(\bar{\eta} + \Delta\eta)] = \mathrm{Tr}[\Sigma_\eta^2(dd^T - bb^T - \Sigma^2)] \tag{B.17}$$

$$= \mathrm{Tr}[\Sigma_\lambda^2(cc^T - \tilde{b}\tilde{b}^T - \tilde{\Sigma}^2)] \tag{B.18}$$

where $\tilde{b} = (c - \mathbb{E}_{\mathcal{D}}\hat{g}_C)$, $\tilde{\Sigma}^2 = \mathbb{E}_{\mathcal{D}}[(\hat{g}_C - \mathbb{E}_{\mathcal{D}}\hat{g}_C)^2]$, and $\hat{g}_C = \hat{V}_{C,\pi}(x) - \gamma\hat{V}_{C,\pi}(x')$. Note that $\mathbb{E}_{\mathcal{D}}[\|c - \hat{g}_C\|^2] = \mathrm{Tr}[\tilde{b}\tilde{b}^T + \tilde{\Sigma}^2]$. Taking $\tilde{b}, \tilde{\Sigma}^2 \to 0$ as the accuracy of $\hat{g}_C$ improves, (B.18) can be estimated as

$$\mathrm{Tr}[\Sigma_\lambda^2 cc^T]. \tag{B.19}$$

Taking the expectation over $c \sim C(x), x \sim \mu_\gamma^\pi$ yields the final result.

B.2 Simulation details

B.2.1 Algorithm

The Constrained Multiagent Advantage Actor Critic (C-MAA2C) algorithm is shown in Algorithm 1. The main differences from the basic MAA2C algorithm are the penalty modifications in lines 9 and 11, the use of vector-valued value functions $\hat{V}^i : \mathcal{X} \to \mathbb{R}^{m+1}$ (one per agent in the noncooperative setting), and the dual update.

There are two apparent differences between Algorithm 1 and the concepts described in the main text. The first is that Algorithm 1 uses n-step returns in the advantage function, whereas Section 4.4 only considers one-step returns. We resolve

this discrepancy by revisiting the proof of Proposition 4.6. First, note that the coefficients η can be factored out of the returns just like they are factored out of the rewards. Thus, the proof only requires slight modifications up to the last line, Equation (B.19). Using returns instead of rewards in (B.19) will lead to a different numerical result but the conclusion (justification for using a structured value function) will be the same.

The second apparent difference is the fact that Algorithm 1 considers finite-horizon episodic tasks, thus the primal-dual algorithm will enforce $\mathbb{E}_{\tau\sim\mathcal{M}}[\prod_{t=0}^{T} c_t] \leq 0$. Due to the finite horizon, we cannot directly use the occupation measure to interpret the meaning of this constraint. However, we can define the occupation measure over a finite horizon as

$$\mu_{\gamma,T}^{\pi}(x) = \frac{1}{1-\gamma^{T+1}} \sum_{t=0}^{T} p_t^{\pi}(x). \tag{B.20}$$

It is easy to show that $\mu_{\gamma,T}^{\pi}$ is nonnegative and integrates to unity over $\mathcal{X}$. We can use $\mu_{\gamma,T}^{\pi}$ in place of μ_{γ}^{π} everywhere in order to interpret discounted sum constraints and to generate probabilistic constraints in finite-horizon episodic tasks. The statements $\mathbb{E}_{\tau\sim\mathcal{M}}[\prod_{t=0}^{T} c_t] \leq 0$, $\mathbb{E}_{\tau\sim\mathcal{M}}[(1-\gamma^{T+1})^{-1} \prod_{t=0}^{T} c_t] \leq 0$, and $\mathbb{E}_{x\sim\mu_{\gamma,T}^{\pi}}[C(x)] \leq 0$ are equivalent. Note that the effective horizon discussed in Section 4.2 may be shorter than the horizon length T.

B.2.2 Hyperparameters

Simulation hyperparameters are listed in Table B.1.

Table B.1: Simulation hyperparameters.

Simulation	
Episode length	25
Number of episodes	$\{4, 8\} \times 10^4$
Number of trials per configuration	5
RL algorithm	
Discount factor γ	0.99
Actor learning rate ζ_θ	3×10^{-4}
Critic learning rate ζ_ω	3×10^{-4}
Dual update step size ζ_λ	1×10^{-4}
Optimizer	$\text{Adam}(\beta_{\text{Adam}} = (0.9, 0.999))$
n-step return horizon κ	5
Constraint enforcement	
$\lambda_{\max}$	10
Risk level β	0.9
"LHS tolerance" α:	
Average constraints	N/A
Chance constraints	0.1
CVaR constraints	0.2
"RHS tolerance" δ:	
Average constraints	0
Chance constraints	0.1

Continued on next page

Table B.1 – continued from previous page

CVaR constraints	5×10^{-3}
Actors	
Policy architecture	Multi-layer perceptron
Number of hidden layers	2
Hidden layer width	64
Hidden layer activation	ReLU
Output layer activation	Linear
Action selection	Categorical sampling
Parameter sharing	No
Critics	
Critic architecture	Multi-layer perceptron
Number of hidden layers	2
Hidden layer width	64
Hidden layer activation	ReLU
Output layer activation	Linear
Target network update interval	200 episodes
Parameter sharing	No

B.2.3 Additional simulation results

Here, we provide some additional results to supplement the findings in Section 4.5. First, we compared the convergence to a safe policy under the original discounted sum constraint and found that similar to the results for the other types of constraints, the structured critic demonstrates a better safety margin throughout training. This is illustrated in Figure B.1.

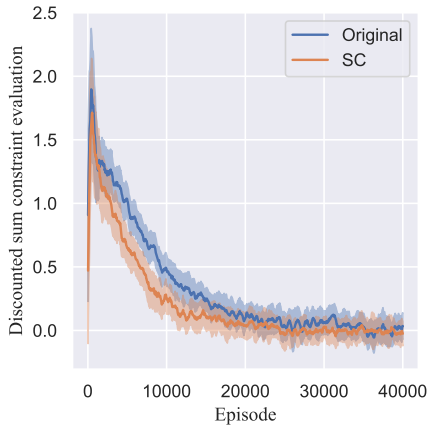

Figure B.1: Evaluation of the discounted sum constraint throughout training, showing that the structured critic helps the actor to find safer policies faster. Each line and shaded region represents the mean and standard deviation over 5 training runs. Key: SC = structured critic.

Next, we provide a closer look at the accuracy of the CVaR upper bound provided in Proposition 4.5, and illustrated using dashed lines in the left panel of Figure 4.7. Table B.2 shows that in all four configurations in which the CVaR was evaluated, the upper bound is a fairly accurate estimate. The results from Section 4.5 show that this upper bound can be used to drive the actual CVaR below a target value. Although using a structured critic with modified penalty function yielded the most accurate CVaR UB, the accuracy in all four configurations could be improved by

making further adjustments to the tolerance α. The error is reported for policies tested at the end of the training phase.

Penalty function	Critic	CVaR UB error
$C(x)$	Generic	18.3%
$C(x)$	Structured	11.8%
$[C(x) - \alpha]_+ - \delta$	Generic	7.6%
$[C(x) - \alpha]_+ - \delta$	Structured	3.7%

Table B.2: Accuracy of CVaR upper bound.

Algorithm 1 C-MAA2C with probabilistic safety & structured value functions

Input discount factor γ, learning rates $\zeta_\theta,\ \zeta_\omega,\ \zeta_\lambda$, n-step return horizon κ, tol. α and δ, mult. limit $\lambda_{\max}$, ep. len. T, num. ep. K, risk metric (avg, chance, CVaR)
Initialize actor params $\{\theta^i\}_{i\in\mathcal{N}}$, critic params $\{\omega^i\}_{i\in\mathcal{N}}$, policies $\pi^i(\cdot \mid \theta^i) : \mathcal{X}_i \to \Delta_{\mathcal{U}_i}$, value estimates $\hat{V}^i(\cdot \mid \omega^i) : \mathcal{X} \to \mathbb{R}^{m+1}$, dual variables $\lambda \in \mathbb{R}^m$
for $k = 1, 2, \ldots, K$ **do**
  Initialize $x_0 \sim p_0$ ▷ Run 1 episode
  for $t = 0, 1, \ldots T$ **do**
    Sample $u_t^i \sim \pi^i(\cdot \mid x_t^i, \theta^i)$ for $i \in \mathcal{N}$
    Receive $\{r_t^i\}_{i\in\mathcal{N}},\ c_t, x_{t+1}$
    if risk metric = chance **then**
      $c_t \leftarrow I[c_t \geq \alpha] - \delta$ ▷ Proposition 4.4
    else if risk metric = CVaR **then**
      $c_t \leftarrow [c_t - \alpha]_+ - \delta$ ▷ Proposition 4.5
    end if
    Let $d_t^i = \begin{bmatrix} r_t^i & c_t^T \end{bmatrix}^T$ for $i \in \mathcal{N}$
  end for
  for $i \in \mathcal{N}$ **do**
    for $t = 0, 1, \ldots, T$ **do**
      $N = \min\{T, t + \kappa\}$
      $D_t^i = \sum_{n=t}^{N-1} \gamma^{n-t} d_n^i + \gamma^{N-t} \hat{V}^i(x_N \mid \omega^i)$ ▷ Compute n-step returns
      $A_t^i = \eta^T (D_t^i - \hat{V}^i(x_t \mid \omega^i))$ where $\eta := \begin{bmatrix} 1 & -\lambda^T \end{bmatrix}^T$ ▷ Compute advantages
    end for
    $\theta^i \leftarrow \theta^i + \zeta_\theta \sum_{t=0}^{T} A_t^i \nabla_{\theta^i} \log \pi^i(u_t^i \mid x_t^i, \theta^i)$ ▷ Actor update
    $\omega^i \leftarrow \omega^i - \zeta_\omega \nabla_{\omega^i} \sum_{t=0}^{T} \|D_t^i - \hat{V}^i(x_t \mid \omega^i)\|_2^2$ ▷ Critic update
  end for
  $\lambda \leftarrow \lambda + \zeta_\lambda \prod_{t=0}^{T} c_t$ ▷ Dual update
  $\lambda \leftarrow \min\{[\lambda]_+, \lambda_{\max}\}$
end for

VITA

Daniel Tabas was born and raised in Pittsburgh, Pennsylvania. He received his PhD and Master's degrees in Electrical Engineering from the University of Washington in 2024 and 2020, respectively, and his Bachelor's degree in Mechanical Engineering from Johns Hopkins University in 2018. Throughout his studies he also worked at Tesla Motors (Palo Alto, CA), the National Renewable Energy Laboratory (Golden, CO), the California ISO (Folsom, CA), the University of Hawaii (Manoa, HI), the Swiss Federal Institute of Technology (Lausanne, CH), and the National University of Ireland (Maynooth, IE). From 2020 to 2023 he was a recipient of the National Science Foundation Graduate Research Fellowship. In his free time, he enjoys going outside and being with friends.