Constrained Meta-Reinforcement Learning for Adaptable Safety Guarantee with Differentiable Convex Programming

Minjae Cho[1], **Chuangchuang Sun**[2]

[1]Department of Mechanical Engineering, Mississippi State University, MS 39762
[2]Department of Aerospace Engineering, Mississippi State University, MS 39762
mc3216@msstate.edu, csun@ae.msstate.edu

Abstract

Despite remarkable achievements in artificial intelligence, the deployability of learning-enabled systems in high-stakes real-world environments still faces persistent challenges. For example, in safety-critical domains like autonomous driving, robotic manipulation, and healthcare, it is crucial not only to achieve high performance but also to comply with given constraints. Furthermore, adaptability becomes paramount in non-stationary domains, where environmental parameters are subject to change. While safety and adaptability are recognized as key qualities for the new generation of AI, current approaches have not demonstrated effective adaptable performance in constrained settings. Hence, this paper breaks new ground by studying the unique challenges of ensuring safety in *non-stationary* environments by solving constrained problems through the lens of the meta-learning approach (learning-to-learn). While unconstrained meta-learning already encounters complexities in end-to-end differentiation of the loss due to the bi-level nature, its constrained counterpart introduces an additional layer of difficulty, since the constraints imposed on task-level updates complicate the differentiation process. To address the issue, we first employ successive convex-constrained policy updates across multiple tasks with differentiable convex programming, which allows meta-learning in constrained scenarios by enabling end-to-end differentiation. This approach empowers the agent to rapidly adapt to new tasks under non-stationarity while ensuring compliance with safety constraints. We also provide a theoretical analysis demonstrating guaranteed monotonic improvement of our approach, justifying our algorithmic designs. Extensive simulations across diverse environments provide empirical validation with significant improvement over established benchmarks.

Introduction

Artificial intelligence (AI) has made significant progress in the past few decades, ranging from mastering board games (AlphaGo (Silver et al. 2016)), and predicting protein structure (AlphaFold2 (Jumper et al. 2021)) to generating human-like texts (GPT-4 (Brown et al. 2020)). Though it has the potential to revolutionize human society like electricity did about one hundred years ago, currently its real-world impact in high-stakes scenarios is still yet proven beyond games. It is observed that, despite those significant successes, deployable Learning-Enabled Systems (LES, (Marcus and Davis 2019)) are far less pervasive. One of the most critical concerns among others towards deployability is safety. In many scenarios, the safety of LES is not compromisable, especially those with humans in the loop. For example, autonomous driving vehicles should guarantee the safety of the drivers and other entities by following the driving rules and operating under various internal and external disturbances, such as partial sensor dysfunction and weather conditions. In healthcare and medicine, the treatment should guarantee that the side effects should not exceed the prescribed threshold. Therefore, learning-enable components should rigorously guarantee safety, and failing to do so can result in undesirable or even disastrous outcomes.

In this paper, we focus on the safety issues of reinforcement learning (RL), a popular framework for sequential decision-making. A significant amount of effort has been made to advance safe RL. For example, constrained reinforcement learning (Chow et al. 2017; Achiam et al. 2017) offers a compelling solution for training policy safely and responsibly by complying with safety constraints. However, there are still major gaps toward deployability in more restrictive environmental assumptions. Consider a *non-stationary* environment with dynamically changing specifications, including the safety criterion. In this case, safety-aware RL policies trained point-wisely with fixed tasks are likely to violate safety constraints in different task settings. In other words, for AI to truly mirror human intelligence, it must possess the ability to adapt quickly to new tasks under constraints. Therefore, it is crucial to develop learning algorithms that enable LES to rapidly adapt while adhering to safety specifications. As it is still challenging for existing safe learning approaches, our goal here is to bridge such a gap by achieving a fast adaptation regarding both performance and safety guarantees in non-stationary environments.

Specifically, we investigate fast-adapting safe RL through the lens of meta-learning, which admits a bi-level structure. Constrained Policy Optimization (CPO, (Achiam et al. 2017)) is employed as the base module for the updates of task-specific parameters and meta parameters at the inner and outer levels, respectively. However, it is well-known that unconstrained meta-learning already admits complex differ-

entiation of the loss function with respect to the meta parameter. This issue is even worse in constrained meta-learning since the inner-level updates under constraints complicate the differentiation process even further. To tackle this challenge, we use CPO which convexifies the constrained policy learning within a trust region for policy updates. Additionally, to facilitate efficient end-to-end differentiation for effective meta-training, we employed Differentiable Convex Optimization (DCO, (Agrawal et al. 2019)) to bridge the inner-level and outer-level updates. As convexity enhances the efficiency of both forward pass and backpropagation, this framework will support adaptable safety guarantees at scale under non-stationarity. This will be further explored in Section , where we delve into solving a constrained meta-learning problem for unprecedented testing tasks in non-stationary environments. To the best of our knowledge, this is the first attempt to build such a framework, providing a promising solution for fast adaptation with safety specifications.

Our main contributions are listed as follows.

- Building a novel architecture by integrating constrained RL into the meta-learning framework, enhancing the ability to provide adaptable safety guarantees.
- Developing a practical method to solve constrained metal-RL via successive convexification and DCO for end-to-end trainability.
- Conducting a thorough evaluation of the Meta-CPO algorithm, which outperforms the benchmarks regarding performance and safety satisfaction.

Preliminaries

Constrained Reinforcement Learning

A Markov Decision Process (MDP) serves as a mathematical framework for modeling sequential decision-making problems. It is composed of several key components represented as a tuple: (S, A, R, P, μ). Here, S denotes the set of states, A represents the set of actions, $R : S \times A \times S \to \mathbb{R}$ signifies the reward function, $P : S \times A \times S \to [0, 1]$ represents the transition probability function. Specifically, $P(s'|s, a)$ indicates the probability of transitioning to state s' given the action a the agent took in previous state s. Additionally, $\mu : S \to [0, 1]$ denotes the starting state distribution. Within an MDP, a policy $\pi : S \to P(A)$ refers to a mapping from states to probability distributions over actions. The notation $\pi(s|a)$ implies the probability of selecting action a in state s.

While RL only aims to maximize a cumulative discounted reward $J_R(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t R(s_t, a_t)\right]$, constrained RL will additionally enforce a cumulative discounted cost constraint as $J_C(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t C(s_t, a_t)\right] \le h$, where h is the safety threshold, $C(s_t, a_t)$ is the cost function, γ the discount factor, τ is the trajectory $\tau = (s_0, a_0, s_1, \dots)$, and $\tau \sim \pi$ means that the trajectory distribution depends on π in the following way: $s_0 \sim \mu, a_t \sim \pi(a_t|s_t), s_{t+1} \sim P(s_{t+1}|s_t, a_t)$.

To guide effective policy learning, one can use either an action-value function, $Q_R^\pi(s, a)$, or a state-value function, $V_R^\pi(s)$. These functions estimate the expected future return based on the current state and chosen action (action-value) or just the state itself (state-value). Their formal definitions are $Q_R^\pi(s, a) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t R(s_t, a_t)|s_0 = s, a_0 = a\right]$ for action-value function and $V_R^\pi(s) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t R(s_t, a_t)|s_0 = s\right]$ for state-value function. In analogy, the action and state value functions for the cost, $Q_C^\pi(s, a) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t C(s_t, a_t)|s_0 = s, a_0 = a\right]$ and $V_C^\pi(s) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t C(s_t, a_t)|s_0 = s\right]$, are defined similarly.

Kakade and Langford (Kakade and Langford 2002) give an identity to express the performance measure of policy π' in terms of the advantage function over another policy π :

$$J_R(\pi') - J_R(\pi) = \frac{1}{1-\gamma}\mathbb{E}_{\substack{s\sim d^{\pi'}\\ a\sim\pi'}}\left[A_R^\pi(s, a)\right] \quad (1)$$

where $d^{\pi'}$ is the discounted future state distribution of policy π', and (1) still depends on expectation of π'. Moreover, $A_R^\pi(s, a)$ is the reward advantage function defined as $A_R^\pi(s, a) \coloneqq Q_R^\pi(s, a) - V_R^\pi(s)$. Similarly, we have the cost advantage function as $A_C^\pi(s, a) = Q_C^\pi(s, a) - V_C^\pi(s)$.

Extending upon (1), subsequent research (Schulman et al. 2015) defined a surrogate function, $M(\pi)$, of (1) to bound the policy improvement, replacing dependencies on π' to π:

$$\begin{aligned} J_R(\pi') - J_R(\pi) &\ge M(\pi') - M(\pi) \\ &= L_\pi(\pi') - CD_{KL}^{\max}(\pi, \pi') \end{aligned} \quad (2)$$

where $L_\pi(\pi') = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d_{a\sim\pi}^{\pi}}\left[A_R^\pi(s, a)\right]$ and $CD_{KL}^{\max}(\pi, \pi')$ represents the error bound. It's worth noting that by decreasing the step size, $D_{KL}^{\max}$, we can minimize the error on a smaller scale, ensuring trust region updates that enforce the right-hand side (RHS) to always be positive. Constrained Policy Optimization (CPO, (Achiam et al. 2017)) further extends this framework by introducing extra inequality constraints. This ensures the preservation of the guarantee of monotonic improvement while accommodating the additional constraint. Consequently, the resultant policies not only aim for optimal performance but also align with specified safety requirements. Building upon (2), we present a theoretical analysis that extends this methodology to a meta-learning setting.

Constrained Policy Optimization (CPO)

Constrained RL methods (e.g., CPO) aim to balance the trade-off between achieving goals and ensuring safety in critical RL tasks, like industrial robot operation and autonomous driving. Leveraging the trust region for guaranteed monotonic improvement, CPO offers intuitive analytical solutions that excel at optimizing policies for a balance of rewards and safety constraints by solving a constrained optimization problem. For CPO, its update procedure is depicted in Figure 1 and the detailed update rule is formulated as follows

$$\begin{aligned} \theta^{k+1} = \underset{\theta}{\operatorname{argmax}} \quad & g_\theta^T(\theta - \theta^k) \\ \text{s.t.} \quad & \frac{1}{2}(\theta - \theta^k)^T H(\theta - \theta^k) \le \delta \\ & b_i + a_\theta^T(\theta - \theta^k) \le 0 \end{aligned} \quad (3)$$

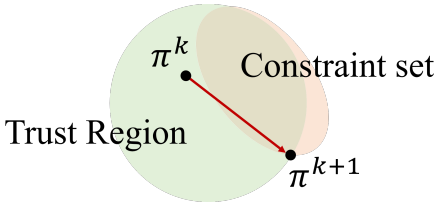


Figure 1: Update procedures for CPO (Achiam et al. 2017). CPO computes the update by simultaneously considering the trust region (light green) and the constraint set (light orange). Figure adopted from (Yang et al. 2020)

with definitions: $g_\theta = \nabla_\theta \mathbb{E}_{s\sim d^\pi}\left[A_R^\pi(s,a)\right]$ is the gradient of the reward advantage function, $a_\theta = \nabla_\theta \mathbb{E}_{s\sim d^\pi}\left[A_C^\pi(s,a)\right]$ is the gradient of the cost advantage function, H is Hessian matrix of KL-divergence $\frac{\partial^2 \bar{D}_{KL}}{\partial \theta^2}$, and $b_i = J_i^C(\pi) - h$.

In case (3) is infeasible, CPO instead solves

$$\begin{aligned} \theta^{k+1} = \underset{\theta}{\text{argmin}} \quad & a_\theta^T(\theta - \theta^k) \\ \text{s.t.} \quad & \frac{1}{2}(\theta-\theta^k)^T H(\theta-\theta^k) \le \delta \end{aligned} \tag{4}$$

by solely decreasing the constraint value within the trust region. We will also use the following notation $\|\theta-\theta^k\|_H^2 \coloneqq (\theta-\theta^k)^T H(\theta-\theta^k)$ for simplicity.

In our approach, CPO serves as the primary update rule for the agents, both meta- and local ones. Because the objective of CPO is to optimize the policy while ensuring safety satisfaction, it is a good fit to be the base learner towards constrained meta-policy learning under non-stationary.

Differentiable Convex Optimization

Differentiable Convex Optimization (DCO, (Agrawal et al. 2019)) is a framework that aims to facilitate the differentiation of convex optimization problems. The DCO layers provide a framework for expressing and solving convex optimization problems in a way that allows for efficient gradient computations and integration with deep learning models. These layers offer a differentiable representation of convex optimization problems, enabling the computation of gradients with respect to the optimization variables. By utilizing affine-solver-affine (ASA) composition and employing canonicalization techniques, DCO layers ensure straightforward computation of gradients through the backward pass. This advancement allows for the integration of convex optimization with deep learning models, enabling meta-learning with nested convex optimization modules with learnable parameters. Specifically, the ASA consists of taking the optimization problem's objective and constraints and mapping them to a cone program. For the following general quadratic programming (QP)

$$\min_x \ \frac{1}{2}x^T Q x + q^T x \text{ s.t. } Ax = b, Gx \le h, \tag{5}$$

we can write the Lagrangian function of the problem as:

$$L(z,\nu,\lambda) = \frac{1}{2}z^T Q z + q^T z + \nu^T(Az-b) + \lambda^T(Gz-h) \tag{6}$$

where ν are the dual variables on the equality constraints and $\lambda \ge 0$ are the dual variables on the inequality constraint. Using the KKT conditions for stationarity, primal feasibility, and complementary slackness.

$$\begin{aligned} Qz^\star + q + A^T\nu^\star + G^T\lambda^\star &= 0 \\ Az^\star - b &= 0 \\ D\left(\lambda^\star\right)\left(Gz^\star - h\right) &= 0 \end{aligned} \tag{7}$$

By differentiating these conditions, we can shape the Jacobian of the problem as follows.

$$\begin{bmatrix} d_z \\ d_\lambda \\ d_\nu \end{bmatrix} = \begin{bmatrix} Q & G^T D\left(\lambda^\star\right) & A^T \\ G & D\left(Gz^\star - h\right) & 0 \\ A & 0 & 0 \end{bmatrix}^{-1} \begin{bmatrix} \left(\frac{\partial \ell}{\partial z^\star}\right)^T \\ 0 \\ 0 \end{bmatrix}$$

Furthermore, via chain rule, we can get the derivatives of any loss function of interest regarding any of the parameters in the QP.

Constrained Meta-Reinforcement Learning

To leverage previous learning experiences, it is crucial to train a model that can adapt to multiple tasks rather than only being optimized for a single task. This is where meta-learning comes into play. Model-agnostic meta-learning (MAML) (Finn, Abbeel, and Levine 2017) has emerged as a powerful technique for training models with generalization capabilities over unseen tasks, demonstrating its potential for few-shot adaptation in both supervised learning and reinforcement learning domains.

In MAML, there are two key components: the meta-learner (θ) and the local-learner (ϕ). Meta-learner could be a distinct model parameter or could be other adaptable parameters such as learning rate and, in reinforcement learning, discount factor γ (Hospedales et al. 2022). The goal of meta-learning is to train the meta-learner in a way that it can quickly adapt to new tasks with minimal training, while the local learner represents the updated parameters obtained through one or more updates on a specific task. By averaging the updates of the local-learner across multiple tasks, the model can achieve strong generalization over its given training tasks.

Each task $\mathcal{T}_i$ consists of its own initial state distribution, $\mu_i(s_0)$ and loss function $\mathcal{L}_{\mathcal{T}_i}$, which leads to the task-specific advantage function $A_{i,R}^\pi$. Moreover, $\mathcal{T}_i$ is represented as an MDP with horizon H, and its trajectory rollout is used for policy evaluation and updates. With the reward and cost functions associated with $\mathcal{T}_i$ and a parameterized policy π_θ, the corresponding return and cost gradients can be expressed similarly to the CPO formulation: $g(\theta,\mathcal{T}_i) = \mathbb{E}_{s_t,a_t\sim\pi_\theta,\mu_{\mathcal{T}_i}}\left[\nabla_\theta A_{i,R}^\pi(s,a|\theta)\right]$ and $a(\theta,\mathcal{T}_i) = \mathbb{E}_{s_t,a_t\sim\pi_\theta,\mu_{\mathcal{T}_i}}\left[\nabla_\theta A_{i,C}^\pi(s,a|\theta)\right]$.

In general, meta-learning trains meta-parameter θ in the outer level and task-specific parameters ϕ_i in the inner level for task $\mathcal{T}_i$. Meta parameters can produce task-specific parameters (e.g., as an initializer) for fast adaptation for both

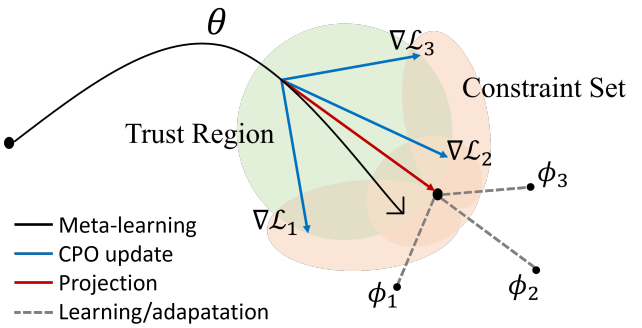


Figure 2: The black curve under meta parameter θ is the learning trajectory in meta-policy space. Each blue $\nabla L_i = \nabla_\theta A^R_{\phi_k}$ is the task-specific gradient, performing a tug-of-war of each task to fulfill generalization/ adaptation over multiple tasks. The projection finds a better intersection of the constraint set, preventing safety violations arising from average gradients.

training and testing in the same fashion, specified as follows

$$\texttt{outer-level:}$$
$$\theta \coloneqq \underset{\theta}{\operatorname{argmax}}\, F(\theta), \quad \text{s.t.} \quad G(\theta) \leq 0$$
$$\text{where} \;\; F(\theta) = \frac{1}{M}\sum_{i=1}^{M} A^{\pi}_{i,R}(\phi_i, \mathcal{D}^{tr}_i), \tag{8}$$
$$\texttt{inner-level:}$$
$$\phi_i = \text{Alg}(\theta, \mathcal{D}^{tr}_i) = \textbf{CPO}(\theta, \mathcal{D}^{tr}_i),$$

where $G(\theta)$ is defined in the same way as $F(\theta)$ with the defined $A^{\pi}_{i,C}(\phi_i, \mathcal{D}^{tr}_i)$. Moreover, $\mathcal{D}^{tr}$ and $\mathcal{D}^{test}$ are the collection of training and testing tasks respectively. In the unconstrained settings, the Alg($\bullet$) is often instantiated as gradient descent updates, which is however inapplicable in constrained settings. Moreover, in the inner level, $\text{Alg}(\theta, \mathcal{D}^{tr}_i)$ can be executed multiple times. Model-based or model-free, primal methods or primal-dual methods, provide many options for the instantiation of Alg($\bullet$). Here, constrained policy optimization (CPO, (Achiam et al. 2017)), a model-free primal method, is chosen as the optimizer.

Our algorithm can be broken down into two parts: local updates with nominal CPO steps and meta updates following the differentiation through the meta parameters. As previously mentioned, CPO serves as a primary update rule for both parameters. This sequential process ensures that the policy optimization is enhanced through the local learner's update and generalized effectively by the meta-learner, resulting in improved overall performance and adherence to the specified constraints. We elaborate on our approach in the subsequent section, and Figure 2 provides a visual representation of the MAML concept in *constrained* settings. In this context, multiple local learners engage in a tug-of-war to guide the policy to their respective task's optimal point. The projection then identifies the intersection of constraint sets from each task, helping to mitigate safety violations that average updates might cause.

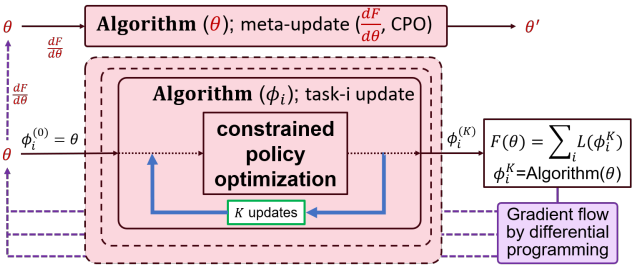


Figure 3: Diagram of our meta-learning approach with CPO as the base algorithm. It optimizes for meta-parameter θ that can quickly adapt to new tasks under constraints.

End-to-End Trainability via DCO with Adaptable Safety Guarantee

In meta-learning, the main computational complexity comes from computing gradients of the loss function with regard to the meta parameter, which has to go through local updates. However, such gradient computations may be intractable due to the complex optimization problem, *i.e.* CPO. To enable meta-learning and facilitate differentiation within the optimization, the Differentiable Convex Optimization (DCO) is used. This framework allows us to effectively and efficiently enable end-to-end differentiation within the optimization layers.

Local Update (Inner-level)

When adapting to a certain task $\mathcal{T}_i$, we update the model's meta parameters θ to its local copy ϕ_i with CPO. In our method, the updated parameter vector ϕ_i is computed using multiple CPO updates on task $\mathcal{T}_i$. With $\phi^0_i = \theta$, the local parameter ϕ_i is updated successively in the following form

$$\begin{aligned} \phi^{k+1}_i = \underset{\phi_i}{\operatorname{argmax}} \quad & g(\phi^k_i, \mathcal{D}^{tr}_i)^T(\phi_i - \phi^k_i) \\ \text{s.t.} \quad & \frac{1}{2}\|\phi_i - \phi^k_i\|^2_H \leq \delta \\ & b_{\phi_i} + a(\phi^k_i, \mathcal{D}^{tr}_i)^T(\phi_i - \phi^k_i) \leq 0 \end{aligned} \tag{9}$$

where the superscript of ϕ^k_i represents the iteration index of local updates.

The parameters are defined in a similar way to those in (3). To obtain the local-learner ϕ^K_i after K updates, we implement CPO for each task $\mathcal{T}_i$ to perform local updates. During the local-update, individual gradients and final updates are stored to link each update at the K local step to the meta-learner using respective gradients that find average updates capturing shared knowledge across tasks. However, policy updates may violate cost constraints $J_C(\pi) \leq h$ and KL-divergence $|\pi - \pi^k|^2_H$, requiring a *backtracking line search* for feasibility (Achiam et al. 2017)."

Algorithm 1: **Meta-CPO** for fast adaption under constraints

Require: $p(\mathcal{T})$: distribution over tasks
Require: δ, h: optimization constraints
Initialization Randomly initialize θ
while not done **do**
  `/*local updates*/`
  Sample tasks under $p(\mathcal{T})$
  for each sampled $\mathcal{T}_i$ **do**
    for $k = 0, \ldots, K-1$ **do**
      Sample multiple trajectories $\mathcal{D} = \{\tau\}$ using policy $\pi_{\phi_i^k}$ in $\mathcal{T}_i$ with $\phi_i^0 = \theta$
      Estimate parameters in (9)
      Update $\phi_i^{k+1} \leftarrow \phi_i^k$ with CPO in (9) and backtracking linesearch
    end for
  end for
  `/*meta updates*/`
  Estimate F, G and compute the gradients $dF/d\theta$ and $dG/d\theta$ with (11) using DCO
  Update θ with (12) and backtracking line search
end while
Output: θ for meta-testing (*not shown*) in new tasks.

Meta Update (Outer-level)

Recall the meta-learning framework in (8) with the following meta loss function with regard to meta parameter θ

$$\begin{aligned} \max_{\theta} \quad & F(\theta) = \frac{1}{M}\sum_{i=1}^{M} A_{i,R}^{\pi}(\phi_i, \mathcal{D}_i^{tr}), \\ \text{s.t.} \quad & G(\theta) \leq 0 \end{aligned} \tag{10}$$

To update the meta parameter θ by gradient descent algorithms, the gradient can be estimated by (chain rule) $\frac{dF}{d\theta} = \frac{1}{M}\sum_{i=1}^{M} \frac{d\text{Alg}_i(\theta)}{d\theta} g(\phi_i, \mathcal{D}_i^{tr})$. Note that the total derivative $\frac{d\text{Alg}_i(\theta)}{d\theta}$ passes derivatives through $\text{Alg}_i(\bullet)$ such that we need to differentiate through it. This highlights a major challenge in meta-learning: even when $\text{Alg}(\bullet)$ is in a fairly simple form of gradient descent in (8), the requirement for second-order derivative makes it computationally intense. As a result, (Finn, Abbeel, and Levine 2017) only keeps the first-order terms, and (Rajeswaran et al. 2019) proposes implicit differentiation. However, for safe learning, the safety constraint can not be reconciled, such as a soft constraint/penalty instead of a hard one. In this case, the complex issue of differentiation is made worse by the fact that $\text{Alg}(\bullet)$ solves a constrained learning problem. This can be part of the reason why constrained meta-learning has not been investigated as much as its unconstrained counterpart because the constraints might damage or complicate end-to-end differentiability. Here we propose to differentiate through the constrained policy update, via DCO, to enable end-to-end meta-training.

With DCO, we can obtain the derivative of the meta loss function with regard to the meta parameters as

$$\frac{dF}{d\theta} = \frac{1}{M}\sum_{i=1}^{M}\prod_{k=0}^{K-1} \frac{d\text{Alg}_i(\phi_i^{(k+1)})}{d\phi_i^{(k)}} g(\phi_i^K, \mathcal{D}_i^{tr}) \tag{11}$$

where $\frac{d\text{Alg}_i(\phi_i^{k+1})}{d\phi_i^k}$ is enabled by differentiating through the local CPO update in (9) by DCO, which allows computing the derivative of the loss function with respect to any parameters in the quadratic programming (9). As multiple parameters in (9), appearing in both objective and constraints, depend on ϕ_i^k, the total derivative $d\text{Alg}_i(\phi_i^{k+1})/d\phi_i^k$ will be the summation of all of the partial derivatives. In analogy, $\frac{dG}{d\theta}$ can be computed in the same way for the update in (10).

With the derivative of the meta objective and constraint functions evaluated, the meta updates can be readily performed in a similar way using CPO.

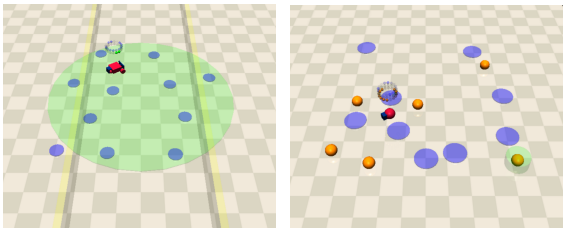
$$\begin{aligned} \theta' = \underset{\theta'}{\text{argmax}} \quad & \left(\frac{dF}{d\theta}\right)^T (\theta' - \theta) \\ \text{s.t.} \quad & \frac{1}{2}\|\theta' - \theta\|_H^2 \leq \delta_\theta \\ & b_\theta + \left(\frac{dG}{d\theta}\right)^T (\theta' - \theta) \leq 0 \end{aligned} \tag{12}$$

Once (12) is infeasible, a similar strategy to (4) is adopted. The evaluation of $F(\theta)$, $G(\theta)$, and their derivatives over multiple tasks enables the generalization to new tasks, including both return improvement and the satisfaction of the safety constraints. The whole architecture of meta-learning with CPO (Meta-CPO) is depicted in Figure 3. The pseudocode is provided in Algorithm 1, while the complete source code is available on GitHub[1].

Theoretical Analysis

In our approach, the rollout $[\theta_n \cdots \{\phi_i^k\}_{i=1}^M \cdots \theta_{n+1}]$ is made to optimize meta-learner θ_n with differentiation through local-learners ϕ_i^k, where k is the number of local iterations and i is index of local-learners. For bi-level meta-updates, we reformulate the work of TRPO/CPO for the theoretical analysis of meta-learner θ. We begin with defining the average performance of local-learner: $\Delta\bar{J}^{k+1} := \bar{J}(\phi^{k+1}) - \bar{J}(\phi^k) \geq \frac{1}{M}\sum_{i=1}^{M}\big[L_{\phi_i^{k+1}}(\phi_i^k) - C_i^k D_{KL}^{\max}(\phi_i^k, \phi_i^{k+1})\big]$, where $\bar{J}(\phi^{k+1})$ is a mean performance of ϕ_i at local step $k+1$ (i.e., $\frac{1}{M}\sum_{i=1}^{M} J(\phi_i^{k+1})$), and RHS is a mean surrogate of performance difference of ϕ_i^k and its next update ϕ_i^{k+1}. Since all local learners demonstrate improvement over their previous iterations within the trust region, carefully chosen step sizes ensure that the meta-learner's performance $J(\theta_{n+1})$ is non-decreasing compared to the previous meta-learner's performance $J(\theta_n)$. This property implies that as long as the local updates stay within the meta-learner's trust region, the meta-learner update is guaranteed to be superior or at least the same as a local

[1] https://github.com/Mgineer117/Meta-CPO



(a) Car-Circle-Hazard (b) Point-Button

Figure 4: Car-Circle-Hazard and Point-button environments. In Car-circle-Hazard environment, the agent avoids the walls and hazards, while the agent strives to only touch activated button by avoiding other dead buttons and hazards in Point-Button environment.

learner, $J(\theta_{n+1}) \geq \bar{J}(\phi_i^{k+1})$. Hence, this can yield the performance guarantees of meta parameters θ:

$$J(\theta_{n+1}) - J(\theta_n) \geq \Delta\bar{J}^{k+1} \geq \bar{L}_{\phi_i^k}(\phi_i^{k+1}) - \bar{C}_i^k \bar{D}_{KL}^{\max}$$

All above analysis upon the return maximization, $J(\theta)$, applies for cost satisfaction, $J_C(\theta)$, as CPO does. This also aligns with empirical experiments of Meta-CPO with a steady learning curve. Consequently, our approach can achieve adaptable safety guarantees while maintaining monotonic performance improvement.

Limitations

However, current DCO layers (cvxpylayers[2]) have limited ability to handle a large number of parameters in computations. This restricts our approach to a smaller parameter scale resulting in unstable performance for high-dimensional tasks and external disturbances. Additionally, the use of cvxpylayers prevents us from employing certain mathematical tricks for effective and memory-efficient matrix-vector product computations for solving CPO. Accordingly, we changed the KL-divergence metric to the Euclidean metric as $\|\theta - \theta^k\|_H^2 \longrightarrow ||\theta - \theta^k||_2^2$. Understanding that such conversion results in model-variant updates can compromise the guaranteed monotonic improvement within the trust region, we implement multiple local steps to alleviate this issue. This approach ensures that the local learner propagated from the meta-learner, is a better policy (under the mild assumption), aligning with the theoretical framework we propose.

Experiment

In our experiments, we aim to answer the following:

- Does Meta-CPO successfully achieve safety satisfaction in a fast-adapting manner?
- How much is Meta-CPO improving the test results?
- What major benefits are gained by Meta-CPO?

We designed three environments utilizing the Python Safety Gym library (Ray, Achiam, and Amodei 2019; Ji et al. 2023). Presented below are concise descriptions of tasks within the environments:

- **Point-Circle:** The agent is rewarded for running in a circle, but is constrained to stay within a safe region.
- **Car-Circle-Hazard:** The agent is rewarded for running in a circle, while staying within a safe region and avoiding hazards.
- **Point-Button:** The agent is rewarded for touching a goal button, but is constrained to touch any no-goal button and step on hazards.

The specifics of each environment, featuring non-physical Walls and Hazards, are visually illustrated in Figure 4. Our experimentation includes three distinct environmental configurations: point-circle ($S \subseteq \mathbb{R}^{28}, A \subseteq \mathbb{R}^2$), car-circle ($S \subseteq \mathbb{R}^{56}, A \subseteq \mathbb{R}^2$), and point-button ($S \subseteq \mathbb{R}^{60}, A \subseteq \mathbb{R}^2$). We used a policy network with two hidden layers of (32, 16). Larger networks like (64, 32) are also feasible following experimental settings demonstrated by the authors of CPO, albeit with some computational cost trade-off.

At each iteration of meta-learning, five tasks are sampled and five local updates are performed for each task (i.e., $K = 5$). Each task within these environments was generated with unique parameters, including factors like radius, distance between walls, number of hazards, and the range for spawning objects. These parameters were selected randomly from uniform distributions within predefined ranges. Table 1 provides a comprehensive overview of the specifics. Following meta-training, a meta-testing phase was conducted to evaluate the rapid learning capabilities of the meta-learner with unseen tasks.

Meta-CPO Evaluation and Comparative Analysis

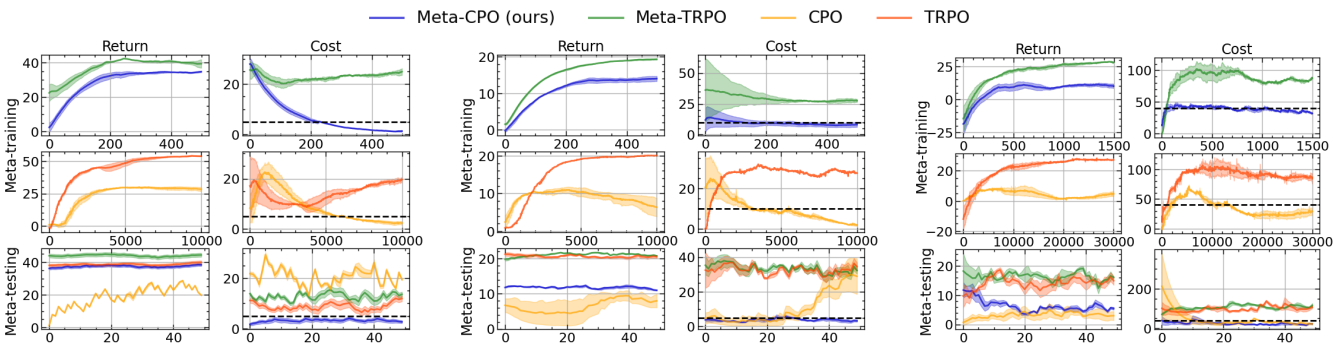
The learning curves depicting the progress of meta-training and meta-testing are presented in Figure 5. To establish a benchmark, we have included Meta-CPO, Meta-TRPO, CPO[3], and TRPO in our comprehensive analysis.

Our analysis indicates that the Trust Region Policy Optimization (TRPO) method consistently converges towards high returns but exhibits significant constraint violations, as expected. Conversely, CPO demonstrates notably unstable behavior during testing phases. The meta-algorithm consistently outperforms non-meta algorithms in both training and testing phases, showcasing rapid and robust adaptation to distinct tasks.

Our innovative Meta-CPO algorithm excels in rapidly learning new tasks while simultaneously satisfying safety constraints. Not only can it acquire new skills swiftly while ensuring safety, but it also demonstrates remarkable adaptability to varied cost constraints. As illustrated in Figure 5 (b), an agent trained with a cost limit of $h = 10$ seamlessly transfers its knowledge to operate effectively under a tighter limit of $h = 5$. This makes Meta-CPO a robust choice in non-stationary environments where safety is paramount.

[2]https://github.com/cvxgrp/cvxpylayers

[3]https://github.com/SapanaChaudhary/PyTorch-CPO



(a) Point-Circle (b) Car-Circle-Hazard (c) Point-Button

Figure 5: Two columns for each task: return (higher is better) and cost (lower is better). The black dashed line is the cost upper bound (below the line means satisfaction). The first two rows are training for meta and non-meta algorithms, respectively. The third row is meta-testing for *unseen* tasks, i.e., deploying meta-trained policies with few-shot adaptation and evaluating them in return and cost. All methods are trained with 3 random seeds, and the mean (solid curve) and standard error (error bar) are plotted with training iterations on x-axis. Meta-CPO (ours) works well in testing against baselines. Specifically for Car-Circle-Hazard, the cost upper bound was changed tighter in testing, i.e., $h : 10 \rightarrow 5$; see Figure 5 (b) (second column, black dashed line). Meta-CPO (ours) can adapt to satisfy the safety constraints (below limit) while all others behave unstably or fail.

Envs.	Meta-training	
P-Circle	$1.0 \leq r_c \leq 1.5$	$0.65 \leq s \leq 0.75$
C-Circle-H	$0.7 \leq r_c \leq 1.0$	$3 \leq n_h \leq 7$
P-Button	$3 \leq (n_b, n_h) \leq 6$	$1.75 \leq r_s \leq 2.0$
Envs.	Meta-testing	
P-Circle	$2.0 \leq r_c \leq 2.5$	$0.55 \leq s \leq 0.65$
C-Circle-H	$1.2 \leq r_c \leq 1.5$	$7 \leq n_h \leq 12$
P-Button	$6 \leq (n_b, n_h) \leq 10$	$2.5 \leq r_s \leq 3.0$

Table 1: P, C, and H represent Point, Car, and Hazard, respectively. Tasks are specified with different environmental and safety settings with the following parameters: r_c and r_s denote the circle radius and spawning range of objects, while s is the wall distance scale. Additionally, we have n_b for the number of buttons and n_h for the number of hazards. For P-Circle and C-Circle-H, $s \times r_c$ determines the wall distance, and the same s is applied for both environments. Task-specific parameters were sampled under a uniform distribution and no testing tasks were seen during training.

Related Works

There is a large volume of works on safe/robust learning, including (Zhang et al. 2020a,b), robotic learning (Brunke et al. 2021; Singh, Kumar, and Singh 2021), and comprehensive surveys (Garcıa and Fernández 2015; Moos et al. 2022). Specifically, the uncertainty variable can be treated as a context variable representing different tasks and can be subsequently solved as multi-task or meta-learning problems (Eghbal-zadeh, Henkel, and Widmer 2021; Rakelly et al. 2019). Moreover, given optimization theories, robust learning algorithms have also been developed based on interior point method (Jin, Mou, and Pappas 2021; Liu, Ding, and Liu 2020), successive convexification (Achiam et al. 2017) and (augmented) Lagrangian methods (Bertsekas and Tsitsiklis 2015; Geibel and Wysotzki 2005; Chow et al. 2017, 2018; Stooke, Achiam, and Abbeel 2020). In learning-based control, Lyapunov theory, model predictive control, and control barrier functions are also employed to develop robust learning algorithms (Choi et al. 2020; Zheng et al. 2021; Cheng et al. 2019; Ames et al. 2016; Berkenkamp et al. 2017; Sun, Kim, and How 2021; Chriat and Sun 2023b,c,a; Kanellopoulos et al. 2021). Additionally, with the worst-case criterion for safety, minimax policy optimization (Li et al. 2019; Zhang, Yang, and Basar 2019) or its generalization Stackelberg games (Yang et al. 2022; Zhou and Xu 2021; Lauffer et al. 2022; Bai et al. 2021) are often the frameworks to promote resilience. Other works include meta-adaptive nonlinear control integrating learning modules for fast adaptation in unpredictable settings (Shi et al. 2021; O'Connell et al. 2022).

Conclusions and Future Work

We proposed a novel constrained meta-Reinforcement Learning (RL) framework for adaptable safety guarantees in non-stationary environments. End-to-end differentiation is enabled via the differentiable convex programming, and the theoretical and empirical analysis demonstrated the advantages of our approach over benchmarks. We suggest future work that concentrates on enhancing the effectiveness and efficiency of generalizable AI, specifically by incorporating causality in scenarios with constraints. While meta-learning aims to leverage memory or training across multiple tasks for generalization, the incorporation of causality, which captures cause-and-effect relationships, has the potential to efficiently transfer knowledge from a particular task to different ones by revealing hidden environmental dynamics. Thus, fusing causality into the existing RL paradigm presents a promising avenue for more efficient learning and improved generalizability. Consequently, our future work will explore this direction to further enhance RL capabilities.

References

Achiam, J.; Held, D.; Tamar, A.; and Abbeel, P. 2017. Constrained policy optimization. In *International conference on machine learning*, 22–31. PMLR.

Agrawal, A.; Amos, B.; Barratt, S.; Boyd, S.; Diamond, S.; and Kolter, J. Z. 2019. Differentiable Convex Optimization Layers. In Wallach, H.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Ames, A. D.; Xu, X.; Grizzle, J. W.; and Tabuada, P. 2016. Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control*, 62(8): 3861–3876.

Bai, Y.; Jin, C.; Wang, H.; and Xiong, C. 2021. Sample-efficient learning of stackelberg equilibria in general-sum games. *Advances in Neural Information Processing Systems*, 34.

Berkenkamp, F.; Turchetta, M.; Schoellig, A.; and Krause, A. 2017. Safe model-based reinforcement learning with stability guarantees. In *Advances in neural information processing systems*, 908–918.

Bertsekas, D.; and Tsitsiklis, J. 2015. *Parallel and distributed computation: numerical methods*. Athena Scientific.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. arXiv 2020. *arXiv preprint arXiv:2005.14165*, 4.

Brunke, L.; Greeff, M.; Hall, A. W.; Yuan, c.; Zhou, S.; Panerati, J.; and Schoellig, A. P. 2021. Safe learning in robotics: From learning-based control to safe reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 5.

Cheng, R.; Orosz, G.; Murray, R. M.; and Burdick, J. W. 2019. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 3387–3395.

Choi, J.; Castañeda, F.; Tomlin, C. J.; and Sreenath, K. 2020. Reinforcement Learning for Safety-Critical Control under Model Uncertainty, using Control Lyapunov Functions and Control Barrier Functions. *arXiv preprint arXiv:2004.07584*.

Chow, Y.; Ghavamzadeh, M.; Janson, L.; and Pavone, M. 2017. Risk-constrained reinforcement learning with percentile risk criteria. *The Journal of Machine Learning Research*, 18(1): 6070–6120.

Chow, Y.; Nachum, O.; Duenez-Guzman, E.; and Ghavamzadeh, M. 2018. A lyapunov-based approach to safe reinforcement learning. *Advances in neural information processing systems*, 31.

Chriat, A. E.; and Sun, C. 2023a. Distributionally Safe Reinforcement Learning under Model Uncertainty: A Single-Level Approach by Differentiable Convex Programming. *arXiv preprint arXiv:2310.02459*.

Chriat, A. E.; and Sun, C. 2023b. On the Optimality, Stability, and Feasibility of Control Barrier Functions: An Adaptive Learning-Based Approach. *arXiv preprint arXiv:2305.03608*.

Chriat, A. E.; and Sun, C. 2023c. Wasserstein Distributionally Robust Control Barrier Function using Conditional Value-at-Risk with Differentiable Convex Programming. *arXiv preprint arXiv:2309.08700*.

Eghbal-zadeh, H.; Henkel, F.; and Widmer, G. 2021. Learning to infer unseen contexts in causal contextual reinforcement learning. In *Self-Supervision for Reinforcement Learning Workshop-ICLR 2021*.

Finn, C.; Abbeel, P.; and Levine, S. 2017. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In Precup, D.; and Teh, Y. W., eds., *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, 1126–1135. PMLR.

Garcıa, J.; and Fernández, F. 2015. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1): 1437–1480.

Geibel, P.; and Wysotzki, F. 2005. Risk-sensitive reinforcement learning applied to control under constraints. *Journal of Artificial Intelligence Research*, 24: 81–108.

Hospedales, T.; Antoniou, A.; Micaelli, P.; and Storkey, A. 2022. Meta-Learning in Neural Networks: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9): 5149–5169.

Ji, J.; Zhang, B.; Pan, X.; Zhou, J.; Dai, J.; and Yang, Y. 2023. Safety-Gymnasium. https://github.com/PKU-Alignment/safety-gymnasium. Accessed: 2023-03-16.

Jin, W.; Mou, S.; and Pappas, G. 2021. Safe pontryagin differentiable programming. *Advances in Neural Information Processing Systems*, 34.

Jumper, J.; Evans, R.; Pritzel, A.; Green, T.; Figurnov, M.; Ronneberger, O.; Tunyasuvunakool, K.; Bates, R.; Žídek, A.; Potapenko, A.; et al. 2021. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873): 583–589.

Kakade, S.; and Langford, J. 2002. Approximately optimal approximate reinforcement learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, 267–274.

Kanellopoulos, A.; Fotiadis, F.; Sun, C.; Xu, Z.; Vamvoudakis, K. G.; Topcu, U.; and Dixon, W. E. 2021. Temporal-logic-based intermittent, optimal, and safe continuous-time learning for trajectory tracking. In *2021 60th IEEE Conference on Decision and Control (CDC)*, 1263–1268. IEEE.

Lauffer, N.; Ghasemi, M.; Hashemi, A.; Savas, Y.; and Topcu, U. 2022. No-Regret Learning in Dynamic Stackelberg Games. *arXiv preprint arXiv:2202.04786*.

Li, S.; Wu, Y.; Cui, X.; Dong, H.; Fang, F.; and Russell, S. 2019. Robust multi-agent reinforcement learning via minimax deep deterministic policy gradient. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 4213–4220.

Liu, Y.; Ding, J.; and Liu, X. 2020. Ipo: Interior-point policy optimization under constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 4940–4947.

Marcus, G.; and Davis, E. 2019. *Rebooting AI: Building artificial intelligence we can trust*. Vintage.

Moos, J.; Hansel, K.; Abdulsamad, H.; Stark, S.; Clever, D.; and Peters, J. 2022. Robust Reinforcement Learning: A Review of Foundations and Recent Advances. *Machine Learning and Knowledge Extraction*, 4(1): 276–315.

O'Connell, M.; Shi, G.; Shi, X.; Azizzadenesheli, K.; Anandkumar, A.; Yue, Y.; and Chung, S.-J. 2022. Neural-fly enables rapid learning for agile flight in strong winds. *Science Robotics*, 7(66): eabm6597.

Rajeswaran, A.; Finn, C.; Kakade, S. M.; and Levine, S. 2019. Meta-learning with implicit gradients. *Advances in neural information processing systems*, 32.

Rakelly, K.; Zhou, A.; Finn, C.; Levine, S.; and Quillen, D. 2019. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International conference on machine learning*, 5331–5340. PMLR.

Ray, A.; Achiam, J.; and Amodei, D. 2019. Benchmarking Safe Exploration in Deep Reinforcement Learning.

Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; and Moritz, P. 2015. Trust Region Policy Optimization. In Bach, F.; and Blei, D., eds., *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, 1889–1897. Lille, France: PMLR.

Shi, G.; Azizzadenesheli, K.; O'Connell, M.; Chung, S.-J.; and Yue, Y. 2021. Meta-adaptive nonlinear control: Theory and algorithms. *Advances in Neural Information Processing Systems*, 34: 10013–10025.

Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; et al. 2016. Mastering the game of Go with deep neural networks and tree search. *nature*, 529(7587): 484–489.

Singh, B.; Kumar, R.; and Singh, V. P. 2021. Reinforcement learning in robotic applications: a comprehensive survey. *Artificial Intelligence Review*, 1–46.

Stooke, A.; Achiam, J.; and Abbeel, P. 2020. Responsive safety in reinforcement learning by pid lagrangian methods. *arXiv preprint arXiv:2007.03964*.

Sun, C.; Kim, D.-K.; and How, J. P. 2021. FISAR: Forward invariant safe reinforcement learning with a deep neural network-based optimizer. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 10617–10624. IEEE.

Yang, B.; Zheng, L.; Ratliff, L. J.; Boots, B.; and Smith, J. R. 2022. Stackelberg MADDPG: Learning Emergent Behaviors via Information Asymmetry in Competitive Games.

Yang, T.-Y.; Rosca, J.; Narasimhan, K.; and Ramadge, P. J. 2020. Projection-Based Constrained Policy Optimization. arXiv:2010.03152.

Zhang, H.; Chen, H.; Xiao, C.; Li, B.; Liu, M.; Boning, D.; and Hsieh, C.-J. 2020a. Robust deep reinforcement learning against adversarial perturbations on state observations. *Advances in Neural Information Processing Systems*, 33: 21024–21037.

Zhang, K.; Sun, T.; Tao, Y.; Genc, S.; Mallya, S.; and Basar, T. 2020b. Robust multi-agent reinforcement learning with model uncertainty. *Advances in Neural Information Processing Systems*, 33: 10571–10583.

Zhang, K.; Yang, Z.; and Basar, T. 2019. Policy optimization provably converges to Nash equilibria in zero-sum linear quadratic games. *Advances in Neural Information Processing Systems*, 32.

Zheng, L.; Shi, Y.; Ratliff, L. J.; and Zhang, B. 2021. Safe reinforcement learning of control-affine systems with vertex networks. In *Learning for Dynamics and Control*, 336–347. PMLR.

Zhou, Z.; and Xu, H. 2021. Decentralized Adaptive Optimal Tracking Control for Massive Autonomous Vehicle Systems With Heterogeneous Dynamics: A Stackelberg Game. *IEEE Transactions on Neural Networks and Learning Systems*, 32(12): 5654–5663.