Augmented Proximal Policy Optimization for Safe Reinforcement Learning

Juntao Dai[1,2*], **Jiaming Ji**[1*], **Long Yang**[3], **Qian Zheng**[1,2†], **Gang Pan**[1,2]

[1] The State Key Lab of Brain-Machine Intelligence, Zhejiang University, Hangzhou, China
[2]College of Computer Science and Technology, Zhejiang University, Hangzhou, China
[3]School of Artificial Intelligence, Peking University, Beijing, China
{juntaodai, jmji, qianzheng, gpan}@zju.edu.cn, yanglong001@pku.edu.cn

Abstract

Safe reinforcement learning considers practical scenarios that maximize the return while satisfying safety constraints. Current algorithms, which suffer from training oscillations or approximation errors, still struggle to update the policy efficiently with precise constraint satisfaction. In this article, we propose Augmented Proximal Policy Optimization (APPO), which augments the Lagrangian function of the primal constrained problem via attaching a quadratic deviation term. The constructed multiplier-penalty function dampens cost oscillation for stable convergence while being equivalent to the primal constrained problem to precisely control safety costs. APPO alternately updates the policy and the Lagrangian multiplier via solving the constructed augmented primal-dual problem, which can be easily implemented by any first-order optimizer. We apply our APPO methods in diverse safety-constrained tasks, setting a new state of the art compared with a comprehensive list of safe RL baselines. Extensive experiments verify the merits of our method in easy implementation, stable convergence, and precise cost control.

Introduction

Reinforcement Learning (RL) (Sutton and Barto 2018) has achieved significant successes in many challenging high-dimensional tasks such as sophisticated video games (Mnih et al. 2013, 2016), robotic locomotion (Peters and Schaal 2008; Schulman et al. 2017; Haarnoja et al. 2018; Vuong, Zhang, and Ross 2019), and Go (Silver et al. 2016, 2017). However, standard RL algorithms merely consider maximizing performance in the policy space, which is insufficient in real-world applications. Some policies are infeasible since they lead to undesirable behaviors, especially for safety violations. For instance, human-machine hybrid scenarios (e.g., autonomous driving, service robots) require machines to perform their tasks without harming humans. Thus, it is essential to explore safe reinforcement learning, which searches for maximizing return policy while satisfying safety constraints. The constrained Markov decision process (CMDP) (Altman 1995) is a well-known sequential framework to formalize safe RL. It attaches additional constraints on the traditional Markov decision process (MDP) to restrict feasible policy sets.

A common approach to solving a CMDP is approximating the non-convex constrained objective function by a quadratic optimization problem within some trust regions (Achiam et al. 2017; Yang et al. 2020). However, these works require expensive computation since their solution includes calculating the inverse Fisher information matrix. According to benchmark evaluation (Ray, Achiam, and Amodei 2019), simple Lagrangian-based algorithms perform as well or even better than the above quadratic optimization methods. However, convergence analysis (Tessler, Mankowitz, and Mannor 2018; Paternain et al. 2019) shows that Lagrangian dual parameters update more slowly than policy iteration. Many constraint-violating iterations arise when the multipliers are not fully effective, and the constraints are over-valued when the multipliers are over-increased. Thus, Lagrangian-based methods have inherent oscillation and cost overshoot in the learning dynamics (Platt and Barr 1987; Wah et al. 2000) leading to poor convergence. The penalty function is another technique used in safe RL, constructing different penalty terms to control cost. Penalty function-based algorithms generally require sufficiently large or infinite penalty factors; otherwise, the constructed penalty function fails to be equivalent to the primal problem and leading to probable suboptimal solutions (Freund 2004). It faces a tough trade-off since growing the penalty factor increases the approximation error (Zhang et al. 2022; Liu, Ding, and Liu 2020).

In this article, we propose the Augmented Proximal Policy Optimization (APPO) method to address the above three issues. Firstly, APPO augments the Lagrangian function of the primal constrained problem via attaching a quadratic deviation term. The additional term provides stable and continuous cost control to complement the lagged Lagrangian multipliers, which improves the convergence of our method. Secondly, we theoretically prove that this attached term, also regarded as a quadratic penalty, is an exact penalty from the primal problem in the constructed primal-dual problem. Thus, the constructed multiplier-penalty function is equivalent to the primal constrained problem to provide precise cost control. Thirdly, we apply the clipped technique inspired by PPO (Schulman et al. 2017) to both reward and cost objectives to update policy with data sampled by proximal policies, which makes APPO eliminate the trust region

[*]These authors contributed equally.
[†]Corresponding author.

constraint and avoid cumbersome Hessian matrix inversion.

Conclusively, the merits of the APPO algorithm are as follows. (1) **Stable convergence.** APPO eliminates the delay in the multiplier's effect, leading to stable and tight convergence to the constraint-satisfying optimal policies without inherent oscillation. (2) **Precise cost control.** Since the multiplier-penalty term is equivalent to the constraints, the solution of APPO is exactly the same as the solution to primal problem. Thus, APPO avoids suboptimal policies resulting from inequivalence and significant approximate errors. (3) **Easy implementation.** APPO alternately updates the Lagrangian multiplier and the policy via solving constructed augmented primal dual problem. It is easy to implement and solve by any first-order optimizer using clipped surrogate objectives for reward and cost.

We outline our contributions from the following aspects.

- We propose a novel method, Augmented Proximal Policy Optimization (APPO), to solve CMDPs. It is shown that APPO has advantages on implementation, convergence, and precise cost control.
- We prove the equivalence between the primal-dual problem and the primal constrained problem (Theorem 1) and illustrate the help of the quadratic deviation term for convergence (Proposition 1).
- We provide the implementation of the proposed APPO algorithm based on the deep Actor-Critic framework while using an analogous clipped surrogate approximation for both reward and cost. We further demonstrate that APPO outperforms the state-of-the-art methods.

Preliminaries

Constrained Markov Decision Process

A Markov Decision Process (Sutton and Barto 2018) is a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, r, P, \mu, \gamma)$, where $\mathcal{S}$ is the state space and $\mathcal{A}$ is the action space. The function $r(s, a, s') : \mathcal{S}\times\mathcal{A}\times\mathcal{S} \to \mathbb{R}$ denotes the reward observed upon a transition from from s to s' with action a. $P(s'|s, a) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition probability from state s to s' after playing action a, and $\mu(s) : \mathcal{S} \to [0, 1]$ is the initial state distribution. $\gamma \in [0, 1]$ is a discount factor. We define a stationary policy π as a probability distribution on $\mathcal{S} \times \mathcal{A}$, with $\pi(a|s)$ denoting the probability of playing action a in state s. Then, Π denotes the set of all stationary policies.

The goal of reinforcement learning is to maximize a performance measure, $J(\pi)$, which is typically defined as an infinite horizon discounted total return,

$$J(\pi) \doteq \mathop{\mathbb{E}}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t r_t\right]. \tag{1}$$

Here $\tau = \{s_t, a_t, r_t\}_{t=0}^{\infty} \sim \pi$ denotes the distribution over trajectories generated by π, where $s_0 \sim \mu$, $a_t \sim \pi(\cdot|s_t)$, $s_{t+1} \sim P(\cdot|s, a)$ and $r_t = r(s_t, a_t, s_{t+1})$.

We express the *state value function* of π as $V^\pi(s) \doteq \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t r_t|s_0 = s\right]$ and the *state-action value function* as $Q^\pi(s, a) \doteq \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t r_t|s_0 = s, a_0 = a\right]$. The advantage function is $A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$.

A Constrained Markov Decision Process (CMDP) (Altman 1995) is a standard MDP $\mathcal{M}$ augmented with an additional set $\mathcal{C}$ of auxiliary cost functions $\{c_i\}_{i=0}^m$ and cost limits $\{b_i\}_{i=0}^m$. Each $c_i(s, a, s') : \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ is a map from transistion tuples to costs.

Define the expected disounted cost return of policy π with respect to the cost function $c_i \in \mathcal{C}$ as $J_{c_i}(\pi) \doteq \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t c_i(s_t, a_t, s_{t+1})\right]$. The set of feasible policies for a CMDP is then $\Pi_\mathcal{C} \doteq \{\pi \in \Pi : J_{c_i}(\pi) \le b_i, \forall i\}$. The goal of safe reinforcement learning with respect to a CMDP is to search the optimal feasible policy π^* such that

$$\pi^* = \arg\max_{\pi\in\Pi_\mathcal{C}} J(\pi). \tag{2}$$

Replacing r with c_i, we define the cost value function $V_{c_i}^\pi$, cost action-value function $Q_{c_i}^\pi$, and cost advantage function $A_{c_i}^\pi$ for the auxiliary costs in an analogy to V^π, Q^π and A^π.

Policy Optimization with Constraints

With the definition of normalized discounted state visitation distribution as $d^\pi(s) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t P(s_t = s|\pi)$, we can express the difference in performance between policies (Kakade and Langford 2002) as:

$$J(\pi') - J(\pi) = \frac{1}{1-\gamma}\mathop{\mathbb{E}}_{\substack{s\sim d^{\pi'}\\ a\sim\pi'}}\left[A^\pi(s, a)\right]. \tag{3}$$

It allows us to rewrite the original goal of safe reinforcement learning (2) into an iterative search for the optimal policy (Achiam et al. 2017; Zhang, Vuong, and Ross 2020):

$$\begin{aligned}&\pi_{k+1} = \arg\max_{\pi} \mathop{\mathbb{E}}_{\substack{s\sim d^{\pi}\\ a\sim\pi}}\left[A^{\pi_k}(s, a)\right]\\ &\textit{s.t.}\quad J_{c_i}(\pi_k) + \frac{1}{1-\gamma}\mathop{\mathbb{E}}_{\substack{s\sim d^{\pi}\\ a\sim\pi}}\left[A_{c_i}^{\pi_k}(s, a)\right] \le b_i, \quad \forall i.\end{aligned} \tag{4}$$

Methodology

In this section, we give the theoretical foundation of APPO. For simplicity, let $\mathcal{L}_R^{\pi_k}(\pi) = \mathbb{E}_{s\sim d^\pi, a\sim\pi}\left[A^{\pi_k}(s, a)\right]$, and $\phi_{c_i}^{\pi_k}(\pi) = J_{c_i}(\pi_k) + \frac{1}{1-\gamma}\mathbb{E}_{\substack{s\sim d^{\pi}\\ a\sim\pi}}\left[A_{c_i}^{\pi_k}(s, a)\right] - b_i$.

Consider the equivalent form of the primal problem (4):

$$\begin{aligned}&\pi_{k+1} = \arg\min_{\pi\in\Pi} -\mathcal{L}_R^{\pi_k}(\pi)\\ &\textit{s.t.}\quad h_{k,i}(\pi) = \phi_{c_i}^{\pi_k}(\pi) + p_i = 0, \quad p_i \ge 0, \quad \forall i,\end{aligned} \tag{5}$$

where p_i is the slack variable to construct the equivalent equality constraint.

We use a quadratic deviation term to supplement the Lagrange function concerning Eq. (5),

$$\begin{aligned}\mathcal{P}_k(\pi, \boldsymbol{\lambda}, \boldsymbol{p}, \sigma) = &-\mathcal{L}_R^{\pi_k}(\pi)\\ &+ \sum_{i=1}^{m}\lambda_i h_{k,i}(\pi) + \frac{\sigma}{2}\sum_{i=1}^{m}h_{k,i}^2(\pi),\end{aligned} \tag{6}$$

where σ is the penalty factor. Then we construct the augmented Lagrangian primal-dual problem as

$$(\pi_{k+1}, \boldsymbol{\lambda}_{k+1}, \boldsymbol{p}_{k+1}) = \max_{\boldsymbol{\lambda}\ge 0}\min_{\pi,\boldsymbol{p}}\mathcal{P}_k(\pi, \boldsymbol{\lambda}, \boldsymbol{p}, \sigma). \tag{7}$$

Like the Lagrangian method, we can alternately update π, $\boldsymbol{\lambda}$, and $\boldsymbol{p}$ to find the optimal triplet.

Since the closed-form solution of $\boldsymbol{p}$ of each iteration can be calculated as $p_i = \max\left\{-\frac{\lambda_i}{\sigma} - \phi_{c_i}(\pi), 0\right\}$, we simplify the solution to the primal-dual problem in Eq. (6) as

$$(\pi_{k+1}, \boldsymbol{\lambda}_{k+1}) = \max_{\boldsymbol{\lambda}\geq 0} \min_{\pi} \mathcal{L}(\pi, \boldsymbol{\lambda}, \sigma), \tag{8}$$

where

$$\mathcal{L}(\pi, \boldsymbol{\lambda}, \sigma) = -\mathcal{L}_R^{\pi_k}(\pi) + \frac{\sigma}{2}\sum_{i=1}^{m}\left(\max\left\{\frac{\lambda_i}{\sigma} + \phi_{c_i}^{\pi_k}(\pi), 0\right\}^2 - \frac{\lambda_i^2}{\sigma^2}\right). \tag{9}$$

See Appendix A.1 for the detailed derivation.

The constructed function (9) contains a multiplier-penalty term to achieve complementary constraint control from the Lagrangian duality and the quadratic deviation penalty. It is shown that the multiplier-penalty term takes effect when the policy comes into a slack margin of constraint violations ($\phi_{c_i}(\pi) \geq -\frac{\lambda_i}{\sigma}$). This adaptive margin leaves a chance for the penalty term to warm up before constraint-violating updates, making constraint control more stable and effective than the penalty after the agent violation.

We also prove that the multiplier-penalty term is an exact penalty since the following Theorem 1 holds:

Theorem 1. *There exists a $\overline{\sigma} < \infty$ such that for $\forall \sigma > \overline{\sigma}$, the primal problem (4) and the augmented primal-dual problem (8) share the same feasible optimal solution set.*

Proof. See Appendix A.2. □

According to Theorem 1, we can search for the optimal feasible policies of a CMDP by iteratively solving the max-min problem (8). Notably, the finiteness of σ is vital since traditional penalty methods require an infinite factor to guarantee the exactness (Freund 2004), which would lead to practical numerical issues and significant approximate errors (Zhang et al. 2022).

The gradient of $\mathcal{L}(\pi, \boldsymbol{\lambda}, \sigma)$ takes the form

$$\nabla_\pi \mathcal{L}(\pi, \boldsymbol{\lambda}, \sigma) = -\nabla_\pi \mathcal{L}_R^{\pi_k}(\pi) + \sum_{\phi_{c_i}^{\pi_k}(\pi) \geq -\frac{\lambda_i}{\sigma}} \left(\lambda_i + \sigma\phi_{c_i}^{\pi_k}(\pi)\right) \nabla_\pi \phi_{c_i}^{\pi_k}(\pi). \tag{10}$$

Suppose $(\pi^*, \boldsymbol{\lambda}^*)$ are the optimal policy and its corresponding dual parameters, and consider the KKT condition to $(\pi^*, \boldsymbol{\lambda}^*)$,

$$\nabla_\pi \mathcal{L}(\pi^*, \boldsymbol{\lambda}^*) = -\nabla_\pi \mathcal{L}_R^{\pi_k}(\pi) + \sum_i \lambda_i^* \nabla_\pi \phi_{c_i}^{\pi_k}(\pi^*). \tag{11}$$

Since Eq. (10) and Eq. (11) is consistent at the optimal point $(\pi^*, \boldsymbol{\lambda}^*)$, we can obtain the following Proposition.

Proposition 1. *The constraint violation nearby the optimal point $(\pi^*, \boldsymbol{\lambda}^*)$ satisfies*

$$\max\left\{\phi_{c_i}(\pi_k), -\frac{\lambda_i^k}{\sigma_k}\right\} \approx \frac{1}{\sigma_k}(\lambda_i^* - \lambda_i^k). \tag{12}$$

According to Proposition 1, We can reduce the constraint violations of the policy iteration close to the optimal point by reasonably increasing the factor σ. Intuitively, a sufficiently large σ can effectively dampen the oscillations to achieve better convergence.

Based on Eq. (12), we give a modified update way for Lagrange multipliers.

$$\lambda_i^{k+1} = \max\left\{\lambda_i^k + \sigma\phi_{c_i}(\pi_k), 0\right\}. \tag{13}$$

So far, the Eq. (9) and Eq. (13) show the key idea of our Augmented Proximal Policy Optimization method.

Algorithm

In this section, we implement our APPO algorithm in a deep RL setting. Consider a set of parameterized policies (i.e., neural networks with a fixed architecture) $\Pi_\theta = \{\pi_\theta : \theta \in \Theta\}$ from which we can easily evaluate and sample. Our algorithm will iteratively update policies by drawing samples from the environment.

It is intractable to solve the minimization of the objective (9) for its inclusion of unknown future state segments and poor sampling efficiency. Inspired by Proximal Policy Optimization (Schulman et al. 2017), APPO uses an analogous clipped surrogate objective for the cost as for the reward. Thus, the practical optimization objective is derived from Eq. (9) as:

$$\mathcal{L}_R(\theta) = \mathbb{E}_t\left[\min\left(r_t(\theta)A_t^R, \text{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)A_t^R\right)\right] \tag{14}$$

$$\mathcal{L}_{c_i}(\theta) = \mathbb{E}_t\left[\max\left(r_t(\theta)A_t^{c_i}, \text{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)A_t^{c_i}\right)\right] \tag{15}$$

$$\mathcal{L}(\theta, \boldsymbol{\lambda}, \sigma) = -\mathcal{L}_R(\theta) + \frac{\sigma}{2}\sum_{i=1}^{m}\left(\max\left\{\frac{\lambda_i}{\sigma} + \mathcal{L}_{c_i}(\theta) - \hat{b}, 0\right\}^2 - \frac{\lambda_i^2}{\sigma^2}\right), \tag{16}$$

where we use the Generalized Advantage Estimator (GAE) (Schulman et al. 2015) to estimate advantages A_t^R, $A_t^{c_i}$ from trajectories $\{s_t, a_t, r_t, V_t, V_t^{c_i}\}_{t=0}^{\infty}$ collected from some old policies $\pi_{\theta_{\text{old}}}$. $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ is the importance sampling ratio and $\hat{b} = J_{c_i}(\pi_{\theta_{\text{old}}}) - b_i$. The surrogate objectives $\mathcal{L}_R$ (14) and $\mathcal{L}_{c_i}$ (15), which are pessimistic bounds on the unclipped objectives, allow us to update policy based on the samples collected from some proximal policies. Such local approximations are proved to often lead to more stable behavior and better sample efficiency (Schulman et al. 2017; Achiam et al. 2017; Metelli et al. 2021).

After each policy update by minimizing the $\mathcal{L}(\theta, \boldsymbol{\lambda}, \sigma)$ in Eq. (16), we use Eq. (13) to update the Lagrangian multiplier. The remaining problem is to pre-select an appropriate penalty factor involved in policy iteration and multiplier update. We provide an adaptively growing way to adjust the penalty factor, which significantly avoids the trouble of selecting the penalty factor.

Referring to the estimated relationship between constraints ϕ_{c_i}, multipliers λ_i, and penalty factor σ in Proposition 1, we define a new metric to measure the constraint

violation degree:

$$\nu(\pi_{\theta_k}) = \sqrt{\sum_{i=0}^{m} \max\left\{ J_{c_i}(\pi_{\theta_k}) - b_i, -\frac{\lambda_i^k}{\sigma_k} \right\}^2}, \quad (17)$$

and a threshold $\varepsilon_k = \frac{1}{\sigma_k}$. We adjust the penalty factor σ_k according to the magnitude of the constraint violation degree $\nu(\pi_{\theta_k})$. When the constraint violation degree $\nu(\pi_{\theta_k})$ exceeds the threshold ϵ_k, we appropriately increase the penalty factor σ_k (i.e., $\sigma_{k+1} = \rho\sigma_k$) to obtain more stable and constraint-satisfying iterations.

Notably, the loss function of APPO in Eq. (16) is differentiable almost everywhere, so we could easily solve it via a first-order optimizer (e.g., Adam optimizer (Kingma and Ba 2014)) to update the parametric policy. We present the pseudo-code of APPO in Algorithm 1.

Related Works

In this section, we review the existing safe RL algorithm that relates to the proposed APPO. Please refer to the survey (Gu et al. 2022) for more discussion.

Quadratic methods. Constrained Policy Optimization (CPO) (Achiam et al. 2017) search policy with near-constraint satisfaction guarantees in a trust region, based on a new bound on the expected return and constraint of two nearby policies. In practice, CPO takes a time-consuming backtracking line search to obtain the next policy. Yang et al. (2020) proposed Projection-based Constrained Policy Optimization (PCPO) utilizing a two-step approach to searches for the optimal policy and then projects the policy back into the feasible set. These quadratic methods involve a local policy search based on conjugate gradient (Süli and Mayers 2003) whose solution requires an inverse Fisher information matrix leading to expensive computation for each update.

Lagrangian-based methods. We find simple Lagrangian-based algorithms perform as well or even better than above higher-order algorithms in a recent empirical comparison in Safety Gym (Ray, Achiam, and Amodei 2019). Primal-Dual Optimization (PDO) (Chow et al. 2017) and its variants (Tessler, Mankowitz, and Mannor 2018; Paternain et al. 2019) leverage the Lagrangian Duality to learn constraint-satisfying policies in safe RL scenarios. We also view FOCOPS (Zhang, Vuong, and Ross 2020) as a primal-dual method, since it uses a simple Lagrangian argument to find its KKT points as the optimum of constrained local policy optimization. However, convergence analysis of the Lagrangian method proves that the multiplier update more slowly than the policy iteration, probably resulting in unsafe intermediate policies during training for no fully effective penalty (Tessler, Mankowitz, and Mannor 2018; Paternain et al. 2019). Thus, the Lagrangian methods are hard to converge, mainly appearing to oscillate when close to the cost threshold. CPPO-PID (Stooke, Achiam, and Abbeel 2020) combine PID control with Lagrangian methods to dampen cost oscillation but is hard to implement and apply in practice due to their sensitivity to three PID hyper-parameters.

Algorithm 1: Augmented Proximal Policy Optimization

Initialize Policy network π_{θ_0}, Value networks V_ψ and Cost Value networks $V_{\varphi_i}^{c_i}$ for all cost function c_i.
Initialize multiplier $\boldsymbol{\lambda}^0$, penalty factor σ_0, and ρ.
for each episodic iteration k **do**
  Generate trajectories $\tau \sim \pi_{\theta_k}$.
  for each mini-batch **do**
    Update critic networks V_ψ and $V_{\varphi_i}^{c_i}$.
    Compute $\mathcal{L}_R(\theta_k)$ in Eq. (14)
    Compute $\mathcal{L}_{c_i}(\theta_k)$ in Eq. (15) for all $c_i \in \mathcal{C}$
    Update Policy network using:
      $\theta_{k+1} \leftarrow \theta_k - \eta\nabla_\theta\mathcal{L}(\theta_k)$ in Eq. (16)
    Update $\boldsymbol{\lambda}$ using:
      $\lambda_i^{k+1} = \max\left\{\lambda_i^k + \sigma\phi_{c_i}(\pi_k), 0\right\}$
    if $\nu(\pi_{\theta_k}) > \varepsilon_k$ **then**
      $\sigma_{k+1} = \rho\sigma_k$
    end if
  end for
end for

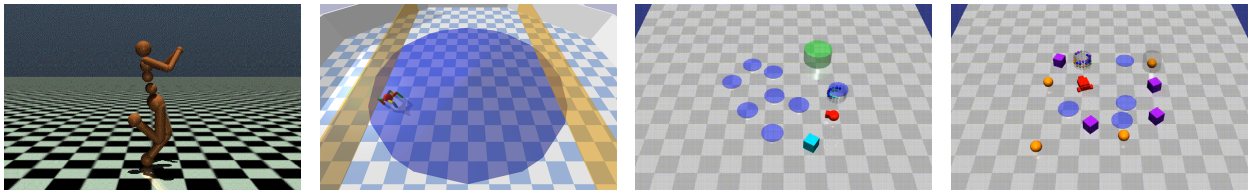
Penalty functions-based methods. Another class of successful first-order methods uses penalty functions to control constraints. IPO (Liu, Ding, and Liu 2020) augments the objective with logarithmic barrier functions to restrict policies into the feasible set. However, it is theoretically shown to provide suboptimal solutions since the surrogate penalty function with infinite penalty factors is not exactly the same as the primal problem(Freund 2004). IPO assumes feasible initialization and intermediate iterations, which cannot be fulfilled in practice and require further recovery. Zhang et al. (2022) propose P3O using exact ℓ_1-penalty method to derive an equivalent unconstrained objective. P3O requires sufficiently large penalty factors, theoretically larger than the optimal dual parameters of the primal problem, to guarantee exactness. It faces a difficult penalty factor tuning since larger factors will bring more significant estimation errors.

Compared with the prior work, the proposed APPO algorithm has advantages in at least three aspects. Firstly, APPO is easy to implement and solve by any first-order optimizer. Secondly, APPO has better convergence since the attached quadratic penalty term complement the lagged Lagrangian multipliers. Thirdly, APPO solve an exact multiplier-penalty function equivalent to the primal problem to have minor approximation errors and avoid converging into suboptimal solutions.

Experiments

In this section, extensive experiments empirically demonstrate the following properties of our APPO algorithm:

- APPO outperforms current state-of-the-art algorithms in safe RL tasks on maximizing cumulative return and satisfying constraints.
- APPO provides reliable safety satisfaction and stable convergence to the optimal policy. It overcomes the inherent oscillation issues due to lagged Lagrange multiplier updates.



(a) Run (b) Circle (c) Goal (d) Button

Figure 1: Four experimental task scenarios: Run, Circle, Goal, and Button.

Algorithm		Ours	PDO	FOCOPS	CPPO-PID	IPO	P3O	CPO	PCPO
Ant	R	$\mathbf{1062.9 \pm 4.4}$	997.5 ± 8.8	1058.2 ± 7.2	1038.6 ± 6.2	1040.8 ± 5.2	1054.0 ± 7.6	1024.1 ± 51.0	1048.7 ± 30.7
(103.12)	C	102.2 ± 6.6	80.9 ± 7.6	100.0 ± 7.4	100.8 ± 6.8	97.6 ± 14.0	102.7 ± 10.5	101.1 ± 73.2	99.1 ± 39.6
Humanoid	R	4989.2 ± 2.3	4953.5 ± 4.9	4925.9 ± 1.2	$\mathbf{4989.8 \pm 2.1}$	931.5 ± 331.3	4864.0 ± 2.74	4963.8 ± 1.4	511.1 ± 118.2
(20.14)	C	18.8 ± 2.7	13.7 ± 3.6	14.1 ± 1.8	16.6 ± 2.1	19.7 ± 8.3	18.5 ± 1.3	19.7 ± 1.4	20.1 ± 4.7
Ball Circle	R	$\mathbf{768.1 \pm 32.0}$	727.6 ± 43.1	498.0 ± 57.1	761.1 ± 23.9	756.7 ± 33.7	610.6 ± 50.7	380.6 ± 33.2	51.5 ± 35.4
(3.0)	C	2.3 ± 7.5	2.7 ± 2.1	2.5 ± 3.7	2.6 ± 4.1	3.0 ± 6.5	2.4 ± 3.3	2.4 ± 3.5	1.5 ± 3.1
Car Circle	R	$\mathbf{1262.9 \pm 52.6}$	1239.5 ± 45.2	898.9 ± 136.0	1204.8 ± 39.4	1223.2 ± 66.2	1055.3 ± 49.5	791.8 ± 92.5	368.0 ± 54.9
(3.0)	C	2.1 ± 2.3	1.8 ± 2.8	2.9 ± 5.8	2.1 ± 2.1	3.0 ± 9.3	1.6 ± 2.9	2.0 ± 4.0	48.2 ± 33.4
Point Goal	R	$\mathbf{23.6 \pm 4.1}$	16.2 ± 2.9	22.8 ± 1.3	11.3 ± 5.6	19.9 ± 1.3	18.4 ± 3.6	26.6 ± 1.4	3.1 ± 2.5
(25.0)	C	24.3 ± 19.7	22.2 ± 5.3	24.8 ± 3.5	24.8 ± 30.0	23.9 ± 21.0	21.8 ± 29.9	89.4 ± 65.4	24.4 ± 45.0
Car Goal	R	$\mathbf{28.1 \pm 4.3}$	28.1 ± 6.9	24.8 ± 21.3	15.5 ± 5.3	25.4 ± 8.0	20.0 ± 16.3	36.4 ± 0.3	34.5 ± 2.3
(25.0)	C	18.5 ± 25.0	20.2 ± 11.6	22.6 ± 29.2	24.2 ± 30.4	24.3 ± 28.6	15.3 ± 22.3	52.2 ± 6.0	60.2 ± 5.5
Point Button	R	$\mathbf{8.6 \pm 7.3}$	6.6 ± 8.0	2.8 ± 0.7	2.3 ± 2.4	5.3 ± 4.9	7.0 ± 4.3	24.2 ± 1.4	29.4 ± 0.6
(25.0)	C	20.4 ± 25.1	17.8 ± 35.8	22.5 ± 9.0	20.1 ± 28.9	24.3 ± 29.9	24.5 ± 26.6	56.6 ± 78.0	58.3 ± 69.7
Car Button	R	$\mathbf{2.2 \pm 2.4}$	1.8 ± 2.6	1.6 ± 3.0	1.0 ± 2.0	1.9 ± 1.2	1.0 ± 2.7	15.2 ± 2.2	20.4 ± 3.1
(25.0)	C	23.9 ± 52.3	23.9 ± 42.4	20.9 ± 34.1	20.7 ± 30.7	60.5 ± 15.3	17.9 ± 27.0	154.6 ± 37.1	277.3 ± 30.7

Table 1: Mean performance and normal 95% confidence interval on eight safe RL scenarios. Cost thresholds are in brackets under the environment names. We bold the best return values with constraint satisfaction.

- APPO provides precise cost control with the exact penalty. It effectively deals with different cost thresholds.
- APPO is robust to its hyper-parameters (the penalty factor and the learning rate of the Lagrangian multiplier).

Baselines. We organize baseline algorithms used for comparison in the following logic. We compare APPO with first-order Lagrangian-based classic or state-of-the-art algorithms (i.e., PDO (Chow et al. 2017; Ray, Achiam, and Amodei 2019), FOCOPS (Zhang, Vuong, and Ross 2020), and CPPO-PID (Stooke, Achiam, and Abbeel 2020)) and penalty function-based methods with similar quality (i.e., IPO (Liu, Ding, and Liu 2020) and P3O (Zhang et al. 2022)). In addition, we consider the existing typical or SOTA quadratic algorithms (i.e., CPO (Achiam et al. 2017), TRPO-L (Ray, Achiam, and Amodei 2019), and PCPO (Yang et al. 2020)).

Task scenarios. For a comprehensive evaluation, we select four representative tasks from three well-known safe RL benchmark environments (Safe MuJoCo (Zhang, Vuong, and Ross 2020), Safety Gym (Ray, Achiam, and Amodei 2019), and Bullet Safety Gym (Gronauer 2022)) as our experimental scenarios. In each task, we learn two different types of intelligent agents. We present the four task scenarios in rough order of difficulty:

- Run: We attempt to train robotic agents (i.e., Ant and Humanoid) like Figure 1(a) to run on a flat surface and meet a specific speed limit that cannot exceed 50% of the speed achieved by an unconstrained PPO agent.
- Circle: The goal of agents (i.e., Ball and Car) in the Circle task is to move as fast and close to the blue circle's boundary in Figure 1(b) as possible. When agents leave the safety zone sandwiched between two yellow boundaries, costs are received.
- Goal: Agents (i.e., Point and Car) move to a series of random goal positions colored green in Figure 1(c). The remaining layouts in the environment are traps and obstacles that the agent is penalized for approaching.
- Button: Agents (i.e., Point and Car) should observe, navigate to, and press the currently highlighted button (the orange one in Figure 1(d)). The Button task and the Goal task shared the same constraint settings.

Since the four tasks are from three different benchmark environments, the types of agents were not uniform. See Appendix B for more details.

Overall Performance

From the perspective of safe RL, we only compare the cumulative return under satisfying constraints and consider all constraint-violating results as poor performance.

Table 1 lists the numerical performance of all tested algorithms in eight safe scenarios. We find that our APPO outperforms other baseline algorithms in finding the optimal policy while satisfying the constraints, most significantly, with an

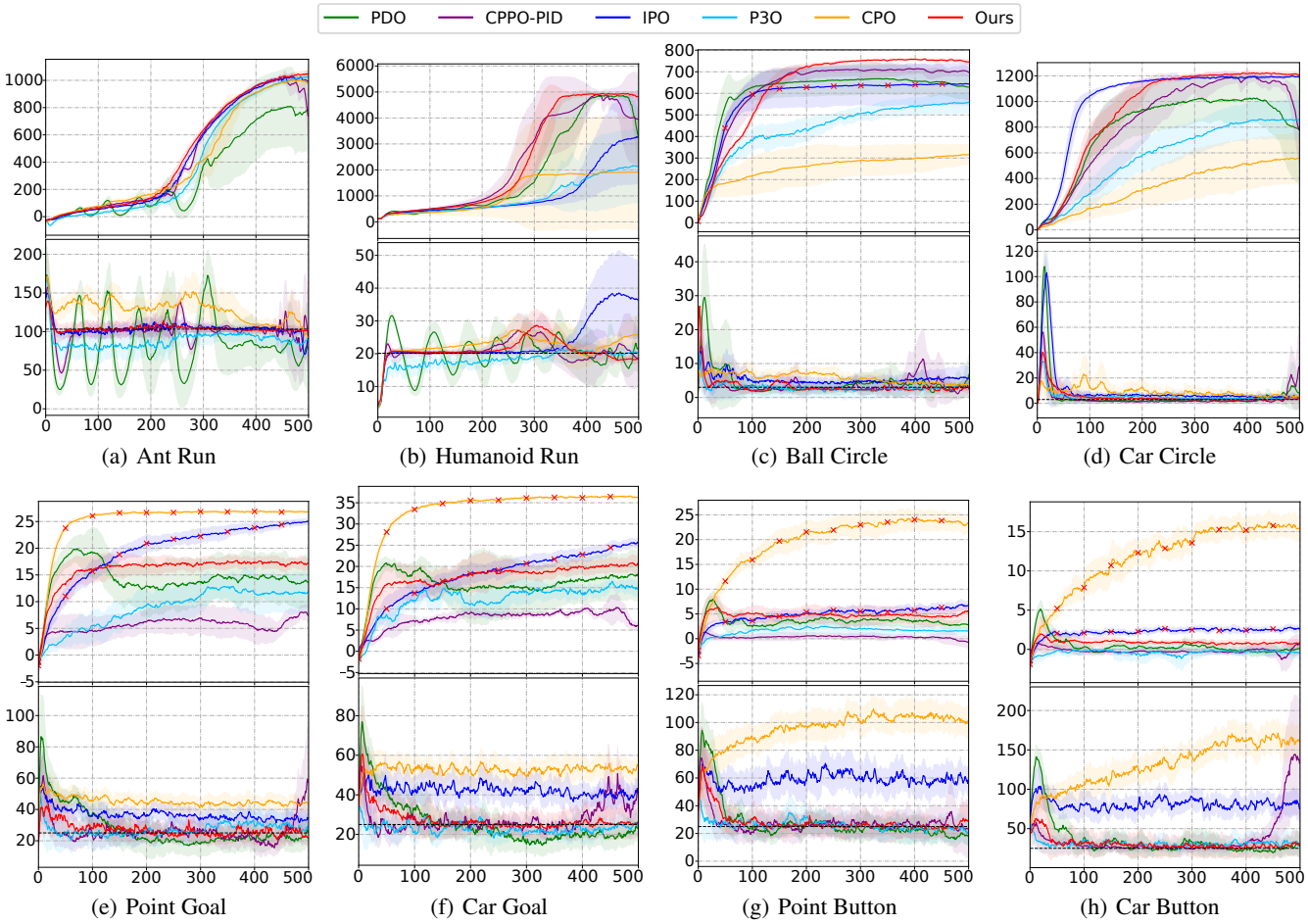


Figure 2: Average episode return (the first plot of each task) and cost (the second plot of each task) for the tested algorithms and tasks. The solid line is the mean, the shaded area is the standard deviation, the black dashed line in the constraint plot is the cost threshold, and the red crosses mark all constraint-violating return curves. See Appendix C for all curves.

average +41% higher reward improvement over PDO, FOCOPS, CPPO-PID and safety violations of IPO, CPO, PCPO on the Car Button task.

Figure 2 shows the corresponding learning curves. The dashed line in the constraint figures is the cost threshold b. We observe that APPO converges to a constraint-satisfying policy faster and smoother than other baseline algorithms, with few constraint-violating behaviors during training. In contrast, algorithms with Lagrangian duality (including PDO, FOCOPS, and CPP-PID) show varying degrees of oscillation, which we further discuss in the next section. We also observe that the penalty function-based methods (IPO and P3O) generally perform slightly worse than the Lagrangian-based methods or lose control of the constraints, such as IPO in Goal and Button tasks. Moreover, Ant Run is a toy task, so most algorithms find the optimal safe policy. However, with the increase in problem complexity and stochastic, some quadratic methods (CPO, TRPO-L, and PCPO) can not satisfy the constraint. Such results are also seen in the prior works (Ray, Achiam, and Amodei 2019).

Evaluation of Advantages

In this section, we empirically verify the merits of APPO. More experimental results are presented in Appendix C.

Stable convergence. APPO has good convergence to the optimal point while satisfying the constraints. It avoids the inherent oscillation of Lagrangian methods. Figure 3 shows the learning curve for one set of experiments with a fixed learning rate of the Lagrangian multiplier for different algorithms. Although all algorithms eventually find the optimal constraint-satisfying policy, APPO converges to the cost threshold most stable and tightly, while other baseline algorithms oscillate around the threshold leading to significant safety violations during training.

Moreover, we observe that the Lagrange multiplier of PDO has a significant lag compared to the constraint curve. It often loses or overdoes cost control due to too small or too large lagged multipliers. FOCOPS limits the maximum of the multipliers. Although this avoids the oscillation success-

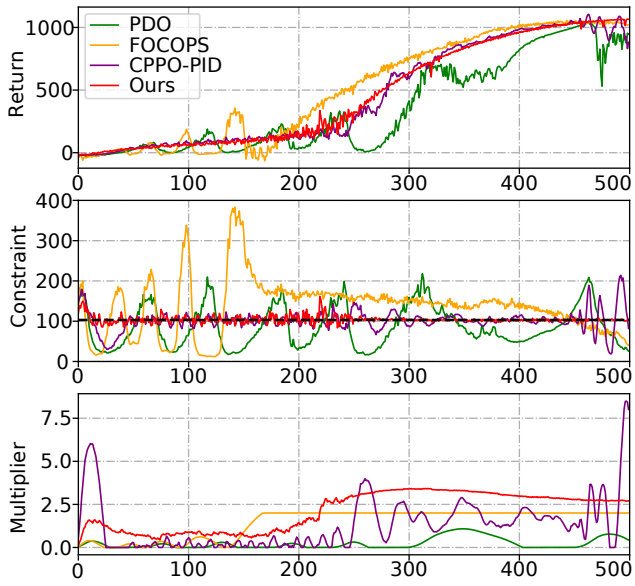


Figure 3: Oscillations in episodic returns (costs and Lagrange multipliers) from algorithms using Lagrange duality to control costs in Ant Run task.

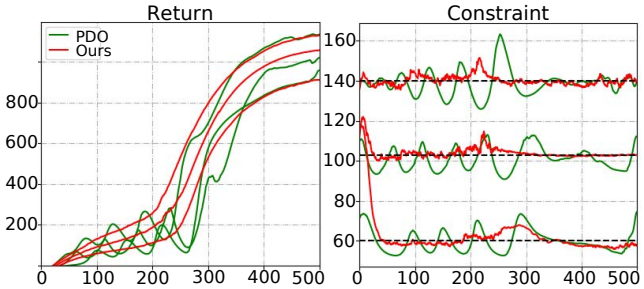


Figure 4: Episodic returns and costs from algorithms using Lagrange dual to control costs at cost limits 60, 103.12, 140 (black dashed lines) in Ant Run task.

fully (as the learning curve after Epoch 200), it is not conducive to satisfying the constraints. CPPO-PID shortens the lag between multipliers update and constraints, which eases the oscillation to some extent. However, the CPPO-PID algorithm includes three sensitive hyper-parameters that need plenty of time to tune.

Precise cost control. Figure 2 shows that APPO has precise safety satisfaction on all difficulty-level tasks. We further verify that APPO works effectively for different cost threshold levels. The experimental results in Figure 4 show that APPO satisfies the constraints in all cases and precisely converges to the cost upper limit to enlarge the policy searching space. Moreover, it illustrates that the oscillation of the baseline Lagrangian-based algorithms is not a case of a particular cost limit.

We design another set of experiments for precise cost control that train different algorithms with fixed hyper-parameters in different cost thresholds. Figure 5 shows that

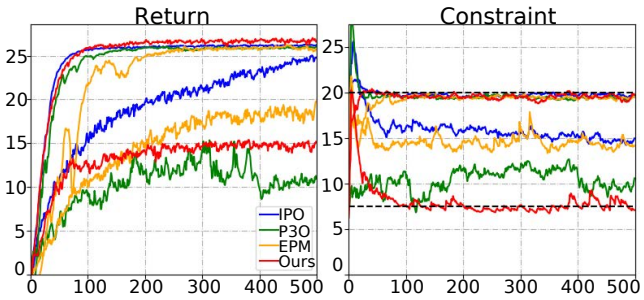


Figure 5: Learning curves from different algorithms in Point Goal task with cost limit 25 and 50. The hyper-parameters are tuned at cost limit 50.

APPO effectively adapts to different cost thresholds, leading to exact cost control with fixed hyper-parameters. We also observe that penalty function-based methods (IPO and P3O) may require further tuning to accommodate different safety settings. In Figure 5, APPO degenerates into the exterior-point method (EPM) when the Lagrangian part is removed. This ablation comparison illustrates the contribution of Lagrangian duality to safety satisfaction in APPO.

Robustness

Besides the learning rate of multipliers, another key hyper-parameter in APPO is the penalty factor σ. We believe that the penalty function term in APPO provides continuous control of the cost and complements each other with the lagged multiplier part. Additionally, Theorem 1 guarantees the exactness of the APPO algorithm under a finite penalty factor. Instead of using fixed σ, we provide an adaptive method in Algorithm 1 to achieve better robustness, namely, increasing σ appropriately whenever the policy exceeds a certain cost threshold. The experimental results are stable in a wide range of hyper-parameter settings. We take the PDO algorithm as a reference since APPO degenerates into PDO when σ is zero. By comparison, the quadratic penalty term does play a role in stabilizing the learning process and converging to a more optimal safe policy. See Appendix C for more robustness experimental results.

Conclusion

In this article, we proposed a novel multiplier-penalty function for solving the constrained Markov decision process and designed a new deep safe RL algorithm, Augmented Proximal Policy Optimization (APPO), to outperform current state-of-the-art safe RL baselines on a wide range of tasks. Moreover, we theoretically and empirically verify the merits of our algorithm in easy implementation, good convergence without oscillation, and exact penalties to avoid suboptimal policies. Since there is still a gap between CMDP and the safety of human-machine cooperation required in reality, we consider that our future work will focus on deep RL tasks with more stringent safety criteria.

Acknowledgments

This paper was supported by STI 2030 Major Projects (2021ZD0200400), Natural Science Foundation of China (61925603), and the Fundamental Research Funds for the Central Universities.

References

Achiam, J.; Held, D.; Tamar, A.; and Abbeel, P. 2017. Constrained policy optimization. In *International Conference on Machine Learning*, 22–31. PMLR. ISBN 2640-3498.

Altman, E. 1995. *Constrained Markov decision processes*. INRIA.

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.

Freund, R. M. 2004. Penalty and barrier methods for constrained optimization. *Lecture Notes, Massachusetts Institute of Technology*.

Gronauer, S. 2022. Bullet-Safety-Gym: A Framework for Constrained Reinforcement Learning. Technical report, mediaTUM.

Gu, S.; Yang, L.; Du, Y.; Chen, G.; Walter, F.; Wang, J.; Yang, Y.; and Knoll, A. 2022. A Review of Safe Reinforcement Learning: Methods, Theory and Applications. *arXiv preprint arXiv:2205.10330*.

Haarnoja, T.; Zhou, A.; Abbeel, P.; and Levine, S. 2018. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In Jennifer, D.; and Andreas, K., eds., *Proceedings of the 35th International Conference on Machine Learning*, volume 80, 1861–1870. PMLR.

Kakade, S.; and Langford, J. 2002. Approximately optimal approximate reinforcement learning. In *In Proc. 19th International Conference on Machine Learning*. Citeseer.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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. ISBN 2374-3468.

Metelli, A. M.; Pirotta, M.; Calandriello, D.; and Restelli, M. 2021. Safe Policy Iteration: A Monotonically Improving Approximate Policy Iteration Approach. *Journal of Machine Learning Research*, 22(97): 1–83.

Mnih, V.; Badia, A. P.; Mirza, M.; Graves, A.; Lillicrap, T.; Harley, T.; Silver, D.; and Kavukcuoglu, K. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*. PMLR.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Paternain, S.; Chamon, L.; Calvo-Fullana, M.; and Ribeiro, A. 2019. Constrained reinforcement learning has zero duality gap. *Advances in Neural Information Processing Systems*, 32.

Peters, J.; and Schaal, S. 2008. Reinforcement learning of motor skills with policy gradients. *Neural networks*, 21(4): 682–697.

Platt, J.; and Barr, A. 1987. Constrained differential optimization. In *Neural Information Processing Systems*.

Ray, A.; Achiam, J.; and Amodei, D. 2019. Benchmarking safe exploration in deep reinforcement learning. *arXiv preprint arXiv:1910.01708*, 7(1): 2.

Schulman, J.; Moritz, P.; Levine, S.; Jordan, M.; and Abbeel, P. 2015. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*.

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

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

Silver, D.; Hubert, T.; Schrittwieser, J.; Antonoglou, I.; Lai, M.; Guez, A.; Lanctot, M.; Sifre, L.; Kumaran, D.; and Graepel, T. 2017. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*.

Stooke, A.; Achiam, J.; and Abbeel, P. 2020. Responsive safety in reinforcement learning by pid lagrangian methods. In *International Conference on Machine Learning*, 9133–9143. PMLR. ISBN 2640-3498.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press. ISBN 0262352702.

Süli, E.; and Mayers, D. F. 2003. *An introduction to numerical analysis*. Cambridge university press. ISBN 0521007941.

Tessler, C.; Mankowitz, D. J.; and Mannor, S. 2018. Reward constrained policy optimization. *arXiv preprint arXiv:1805.11074*.

Vuong, Q.; Zhang, Y.; and Ross, K. W. 2019. Supervised policy update for deep reinforcement learning. In *International Conference on Learning Representations*.

Wah, B. W.; Wang, T.; Shang, Y.; and Wu, Z. 2000. Improving the performance of weighted Lagrange-multiplier methods for nonlinear constrained optimization. *Information Sciences*, 124(1-4): 241–272.

Yang, T.-Y.; Rosca, J.; Narasimhan, K.; and Ramadge, P. J. 2020. Projection-based constrained policy optimization. *arXiv preprint arXiv:2010.03152*.

Zhang, L.; Shen, L.; Yang, L.; Chen, S.; Yuan, B.; Wang, X.; and Tao, D. 2022. Penalized Proximal Policy Optimization for Safe Reinforcement Learning. *arXiv preprint arXiv:2205.11814*.

Zhang, Y.; Vuong, Q.; and Ross, K. W. 2020. First order constrained optimization in policy space. *arXiv preprint arXiv:2002.06506*.