# Efficient Constrained Policy Optimization

**Amirreza Kazemi**
aka208@sfu.ca

**Alireza Akbari Khanaposhtani**
aaa208@sfu.ca

## Abstract

We study the problem of policy optimization in an infinite-horizon $\gamma$-discounted Constrained Markov Decision Process (CMDP). Our goal is to learn a policy that returns a large expected total reward and also violates the constraints by a small amount. Specifically, we employ a primal-dual approach for the constrained optimization problem and update the primal (policy) and dual (lagrangian multiplier) variables alternatively. In order to update the policy parameters, we rely on a surrogate objective function instantiated from FMA-PG framework Vaswani et al. [2021]. The surrogate functions enable us to optimize our policy efficiently in terms of the agent interaction with the environment, and also enjoy monotonic improvement irrespective of the policy parameterization. We experimentally validated the performance of the proposed method in tabular and linear function approximation settings on synthetic and CartPole environments.

## 1 Motivation

Reinforcement Learning (RL) algorithms have seen significant progress on a variety of sequential decision-making tasks such as robot locomotion and manipulation Levine et al. [2015], Lillicrap et al. [2015], playing Go Silver et al. [2016], and Starcraft DeepMind [2019]. In these applications, the agent is allowed to explore any action or state during learning. However, in many safety-critical applications such as Abbeel et al. [2006] and autonomous driving, complete freedom could potentially cause damage to the RL agent or the environment around it. Hence, the agent is subject to constraints on its utilities. Constrained Markov Decision Process (CMDP) Altman [1999a] is a natural framework to model the long-term constraints that must be satisfied by an agent. The objective for CMDP is maximizing expected return (similar to MDPs) while (approximately) satisfying a set of long-term constraints.

Policy gradient (PG) methods Sutton et al. [1999] are an important class of model-free algorithms in RL and have been extensively used to solve CMDPs Spooner and Savani [2020], Tessler et al. [2018], Ding et al. [2020], Xu et al. [2020]. In particular, Ding et al. [2020], Xu et al. [2020] have employed the natural policy gradient method Kakade [2001] to learn a policy and have achieved global convergence in terms of optimality gap and constraint violation. Even though there is no lack of theory in the above-mentioned works, they need to interact with the environment for each policy update, i.e. collecting samples based on the current policy, which can be computationally expensive. On the other hand, and in order to develop efficiently implementable algorithms, methods such as TRPO Schulman et al. [2015], PPO Schulman et al. [2017]and MDPO Abdolmaleki et al. [2018], originally proposed for MDPs, have been extended to solve CMDPs Achiam et al. [2017]. These methods rely on constructing a surrogate function for the policy and then updating the policy parameters by maximizing the surrogate. Although surrogate-based methods have achieved great empirical performance, they have theoretical guarantees only in the tabular setting.

In this project, we aim to employ a surrogate function instantiated from FMA-PG framework and apply it on a constrained RL problem. The surrogate function supports off-policy updates and shares computational feasibility for high-dimensional state-action spaces with other surrogate-based methods. Furthermore, in contrast with existing surrogate functions, ours enjoys monotonic policy improvement for any policy parametrization.

## 2 Related Work

In general, the literature on the constrained Markov Decision Process can be categorized as:

**Primal-only approach**. The primal-only methods Achiam et al. [2017], Chow et al. [2018], Dalal et al. [2018], Liu et al. [2020], Xu et al. [2020] directly target finding an optimal feasible policy without relying on the Lagrangian formulation. In order to take the constraints into account, primal-only methods impose constraints through a special design of the objective function or the policy update rule. For instance, inspired by trust region policy optimization (TRPO) Schulman et al. [2015], the Constrained Policy Optimization (CPO) method Achiam et al. [2017] designs a surrogate objective for CMDPs. Interior-point Policy Optimization (IPO) Liu et al. [2020] method augments the objective with a logarithmic barrier function and proposes a first-order policy optimization algorithm. More recently, Xu et al. [2020] proposed Constrained-Rectified Policy Optimization (CRPO) by employing immediate switches between optimizing the objective and reducing the constraint only if the constraint is violated. Of primal-only methods, Xu et al. [2020] is the only method with convergence guarantee to the globally optimal policy in both tabular and function approximation settings.

**Primal-dual approach**. Primal-dual methods Tessler et al. [2018], Paternain et al. [2019], Yu et al. [2019], Ding et al. [2020] Stooke et al. [2020] convert the constrained optimization problem into unconstrained saddle-point problem. Consequently, they have to update the primal (policy) variable as well as the dual (Lagrangian multiplier) variable. Briefly, Tessler et al. [2018] guaranteed a local optimal convergence, and Paternain et al. [2019] demonstrated that their method converges to the neighborhood of optimal policy. Adopting natural policy gradient Kakade [2001] optimization for primal variables and gradient descent for dual variables, Ding et al. [2020] proposed the only primal-dual algorithm that converges to global optimal policy in the general policy parameterization setting.

## 3 Problem Formulation

Consider a discounted infinite horizon constrained Markov Decision Process (CMDP) Altman [1999b] defined by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, r, c, b, \gamma, \rho \rangle$, where $\mathcal{S}$ denotes a countable state set, $\mathcal{A}$ denotes a countable action set, $\mathcal{P}$ is a transition probability function $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \Delta_{\mathcal{S}}$ ($\Delta_{\mathcal{S}}$ is a simplex over states) with $P(s'|s,a)$ is the probability of transitioning into state $s'$ under action $a$ in the previous state $s$, $r$ is a reward function $r : \mathcal{S} \times \mathcal{A} \to [0,1]$, $c$ is a constrain reward function $c : \mathcal{S} \times \mathcal{A} \to [0,1]$, $b$ is a constrain offset, $\gamma \in [0,1)$ is a discount factor and $\rho$ is the initial distribution over states. Each policy $\pi : \mathcal{S} \to \Delta_{\mathcal{A}}$ induces a distribution $p^\pi(.|s)$ over actions for each state $s$. It also induces a measure $d^\pi$ over states such that $d^\pi(s) = \sum_{t=0}^{\infty} \gamma^t P(s_t = s | s_0 \sim \rho, a_t \sim p^\pi(a_t|s_t))$. Similary we define a measure over state-action $\mu^\pi$ such that $\mu(s,a)^\pi = d^\pi(s) p^\pi(a|s)$.

For a given policy $\pi : \mathcal{S} \to \Delta_{\mathcal{A}}$, we define value functions $V_r^\pi : \mathcal{S} \to \mathbb{R}$ and $V_c^\pi : \mathcal{S} \to \mathbb{R}$ associated with the reward $r$ and the constraint $c$ respectively as follows:

$$V_r^\pi(s) = \mathbb{E}_{\mathcal{P},\pi}[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | s_0 = s] \qquad V_c^\pi(s) = \mathbb{E}_{\mathcal{P},\pi}[\sum_{t=0}^{\infty} \gamma^t c(s_t, a_t) | s_0 = s]$$

where $a_t \sim \pi(.|s_t)$ and $s_{t+1} \sim \mathcal{P}(.|s_t, a_t)$. Similarly, for a given policy $\pi$ we define state-action value functions $Q_r^\pi(s,a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and $Q_c^\pi(s,a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ as follows:

$$Q_r^\pi(s,a) = \mathbb{E}_{\mathcal{P},\pi}[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | s_0 = s, a_0 = a] \qquad Q_c^\pi(s,a) = \mathbb{E}_{\mathcal{P},\pi}[\sum_{t=0}^{\infty} \gamma^t c(s_t, a_t) | s_0 = s, a_0 = a]$$

From Bellman equation $Q_\Diamond^\pi(s,a) = \Diamond(s,a) + \gamma \mathbb{E}_{s' \sim P(.|s,a)}[V_\Diamond^\pi(s')]$, where $\Diamond$ symbol is $r$ or $c$ function. Let $\Pi$ be a set of all possible policies, the agent's objective is to find a policy $\pi \in \Pi$ that maximizes the expected total reward defined as $J_r(\pi) = \mathbb{E}_{s \sim d^\pi}[V_r^\pi(s)]$ while ensuring the expected total constraint reward $J_c(\pi) = \mathbb{E}_{s \sim d^\pi}[V_c^\pi(s)]$ is greater than or equal to some constant $b$. Formally:

$$\max_{\pi \in \Pi} J_r(\pi) \quad \text{s.t.} \quad J_c(\pi) \geq b \tag{1}$$

# 4 Method

We follow the primal-dual approach in this work. Let $\pi^* \in \Pi$ be the solution for the problem in 1. From Lagrangian duality we have: $\pi^*$ is the solution for 1, if and only if for some $\lambda^* \geq 0$, ($\pi^*$, $\lambda^*$) is the solution for the following unconstrained optimization problem.

$$\max_{\pi \in \Pi} \min_{\lambda \geq 0} J_r(\pi) + \lambda(J_c(\pi) - b) = J(\pi, \lambda) \tag{2}$$

We aim to alternatively update $\pi$ and $\lambda$ for $T$ iterations. In other words, as shown in equation 3 at each iteration we first fix the $\pi$ and minimize the objective with respect to $\lambda$, and then fix $\lambda$ and maximize the objective with respect to $\pi$. In section 4.1 we explain the FMA-PG framework and our instantiated surrogate function (sMDPO). We will then use sMDPO and projected gradient descent to update $\pi$ and $\lambda$ for $T$ iterations respectively in section 4.2.

$$\lambda_{t+1} = \arg\min_{\lambda \geq 0} J(\pi_t, \lambda) \qquad \pi_{t+1} = \arg\max_{\pi \in \Pi} J(\pi, \lambda_{t+1}) \tag{3}$$

## 4.1 FMA-PG and sMDPO

We need to first explain a notion called policy functional representation which FMA-PG relies on to construct surrogates. Functional representation of policy $\pi$ refers to a set of simplex $p^\pi(.|s)$ for all states $s$ or set of logits $z^\pi(a, s)$ [1] for all state-action pairs $(a, s)$. Note that there are multiple functional representations for a single policy and the policy functional representation is independent from its parameterization. Defining functional representation, we now state functional mirror ascent (FMA), a form of mirror ascent [cite] that updates functional representation directly. Let us define a strictly convex diffrentiable function $\phi$ as mirror map and $D_\phi(\pi, \mu) = \phi(\pi) - \phi(\mu) - \langle \phi(\mu), \pi - \mu \rangle$ as bregman divergence between policies $\pi$ and $\mu$ associated with mirror map $\phi$. FMA update to maximize function $J(\pi, \lambda)$ with respect to $\pi$ is as follows:

$$\pi_{t+1} = \arg\max_{\pi \in \Pi} [\langle \pi, \nabla_\pi J(\pi_t, \lambda_{t+1}) \rangle - \frac{1}{\eta} D_\phi(\pi, \pi_t)] \tag{4}$$

Note that $\pi$ is a functional representation of policy and the gradient of $J(\pi, \lambda)$ is taken with respect to that. The FMA update returns a policy that is aligned with the gradient direction at the current policy and its divergence from the current policy is restricted by the factor $\frac{1}{\eta}$. Finding the optimal policy among all feasible policies $\Pi$ is however intractable. Consequently, the standard approach is to parameterize policies by $\theta \in \mathbb{R}^d$, and we assume $\Pi$ only consists policies that are realizable by a model parameterized with $\theta$. Therefore, defining $\pi(\theta)$ as parametric realization of functional representation $\pi$, the update can be rewritten as:

$$\theta_{t+1} = \arg\max_{\theta \in \mathbb{R}^d} [\langle \pi(\theta), \nabla_\pi J(\pi(\theta_t), \lambda_{t+1}) \rangle - \frac{1}{\eta} D_\phi(\pi(\theta), \pi(\theta_t))] \tag{5}$$

where $\pi_{t+1} = \pi(\theta_{t+1})$ and $\pi_t = \pi(\theta_t)$. Importantly, we can optimize the above equation using gradient ascent iteratively on $\theta$ without any need to run the updated policy and collect samples from the environment. This property known as off-policy updates enables policy parameters $\theta$ to be updated multiple times with only one sample collection, and thus makes our method computationally efficient. Adding some constant terms with respect to $\theta$ to the update rule in equation 5, we can obtain a surrogate function for $J(\pi(\theta), \lambda_t)$:

$$\theta_{t+1} = \arg\max_{\theta \in \mathbb{R}^d} l_t^{\pi,\phi,\eta}(\theta) = J(\pi(\theta_t), \lambda_{t+1}) + \langle \pi(\theta) - \pi(\theta_t), \nabla_\pi J(\pi(\theta_t), \lambda_{t+1}) \rangle - \frac{1}{\eta} D_\phi(\pi(\theta), \pi(\theta_t)) \tag{6}$$

Therefore, we aim to maximize $J(\pi(\theta), \lambda_t)$ through maximizing the above generic surrogate function $l_t^{\pi,\phi,\eta}(\theta)$. We need to choose the policy functional representation $\pi$ and the mirror map $\phi$ to specify our surrogate. In this project, we choose softmax $z^\pi(a, s)$ for functional representation and *logsumexp* for the mirror map. Therefore, we have $\frac{\partial J(\pi)}{\partial z^\pi(a,s)} = d^\pi(s) A^\pi(a, s) p^\pi(a|s)$ and

[1] $p^\pi(a|s) = \frac{\exp z^\pi(a,s)}{\sum_{a'} \exp z^\pi(a',s)}$

$\phi(z^\pi) = \sum_s d^\pi(s) \log \sum_a \exp z^\pi(a, s)$. Substituting the resulting terms in $l_t(\theta)$ formula we attain sMDPO surrogate function, and the update rule for $\theta$ will be:

$$\theta_{t+1} = \arg\max_{\theta \in \mathbb{R}^d} l_t^{sMDPO}(\theta) = E_{(s,a)\sim\mu^\pi(t)}[A^{\pi_t}(s, a, \lambda_{t+1}) + \frac{1}{\eta} \log \frac{p^\pi(a|s, \theta)}{p^\pi(s|a, \theta_t)}] \tag{7}$$

where $A^{\pi_t}(s, a, \lambda_{t+1}) = A_r^{\pi_t}(s, a) + \lambda_{t+1} A_c^{\pi_t}(s, a)$.[cite] proved that optimizing the above surrogate leads to monotonic improvement for $J(\pi, \lambda_t)$ for any $\eta \leq 1 - \gamma$, where $\gamma$ is the discount factor. In other words, for $\theta_{t+1}$ which and $\theta_t$ we have $J(\pi(\theta_{t+1}), \lambda_{t+1}) \geq J(\pi(\theta_t), \lambda_{t+1})$, where $\theta_t$ is the parameter used to construct surrogate $l_t^{sMDPO}(\theta)$, and $\theta_{t+1}$ is the parameter that maximizes $l_t^{sMDPO}(\theta)$. Importantly, for a $\beta$-smooth $l_t(\theta)$, any number of gradient ascent update on $l_t(\theta)$ with a step size $\alpha_\pi \leq \frac{1}{\beta}$ results in a $\theta_m$ that the monotonic improvement condition holds for it, i.e. $J(\pi(\theta_m), \lambda_t) \geq J(\pi(\theta_t, \lambda_t)$. Assuming $l_t$ is $\beta$-smooth, we will perform some iterations of gradient ascent on it and the updated $\theta$ improves our objective. Also, note that we have not assumed any specific parametrization for $\theta$ and the results hold for any parametrization including neural networks.

### 4.2 Primal and Dual Updates

In the previous section, we derived a surrogate objective that allows us to update policy parameters. Combining policy parameter update procedure with projected gradient descent for updating dual variable (equation 8) we have shown constrained FMA-PG algorithm in Algorithm 1.

$$\nabla_\lambda J(\pi_t, \lambda) = J_c(\pi_t) - b \qquad \lambda_{t+1} = \mathbb{P}_{[0,U]}[\lambda_t - \alpha_\lambda \nabla_\lambda J(\pi_t, \lambda)] \tag{8}$$

$U$ is the bounded on the optimal dual variable $\lambda^*$, and is computed as $\frac{1}{(1-\gamma) max_\pi J_c^\pi - b}$.

**Algorithm 1** Constrained FMA-PG

**Input:** $\pi$ = softmax, $\phi$ = *logsumexp*, $\theta_0$, $\lambda_0$, $T$, $\eta = 1 - \gamma$, $\alpha_\pi$ (inner loop step size for policy update), $\alpha_\lambda$ (step size for dual update), $m$ inner loops, $\gamma$ (discount factor)

Compute $U = \frac{1}{(1-\gamma) max_\pi J_c^\pi - b}$
**for** $t \leftarrow 0$ to $T - 1$ **do**
    Compute functional representation $\pi_t = \pi(\theta_t)$.
    Run policy associated with $\pi_t$ to estimate $Q_c^{\pi_t}(s, a)$ and $Q_r^{\pi_t}(s, a)$.
    Compute $\nabla_\lambda J(\pi_t, \lambda) = J_c(\pi_t) - b$ and update $\lambda$ using 8.
    Compute $A^{\pi_t}(s, a, \lambda_{t+1}) = A_r^{\pi_t}(s, a) + \lambda_{t+1} A_c^{\pi_t}(s, a)$
    Form sMDPO surrogate function $l_t^{sMDPO}(\theta)$ as shown in 7.
    Initialize inner-loop: $\omega_0 = \theta_t$.
    **for** $k \leftarrow 1$ to $m$ **do**
        $\omega_{k+1} \leftarrow \omega_k + \alpha_\pi \nabla_\omega l_t(\omega_k)$
    **end for**
    $\theta_{t+1} = \omega_m$.
**end for**

**Output:** $\lambda_T, \theta_T$.

Having monotonic improvement for $J(\pi, \lambda)$ in the above algorithm , we can ensure that $J_r(\pi(\theta_{t+1})) + \lambda J_c(\pi(\theta_{t+1})) \geq J_r(\pi(\theta_t)) + \lambda J_c(\pi(\theta_t))$ for any $\lambda$. However, the desired behavior is monotonically improving the expected return associated with reward $J_r^(\pi)$ while satisfying the constraint on $J_c(\pi)$. Our theoretical bound does not necessarily imply this and forming a theoretical guarantee on $J_r(\pi)$ is left for future works.

## 5 Experiment

In this section, we perform experiments in two environments to compare sMDPO with GDA Ding et al. [2020] and CBP Jain et al. [2022] methods. The first environment is a 5X5 synthetic gridworld environment. The second is the Cartpole environment from the OpenAI gym Brockman et al. [2016], and is modified to include two constraints. The agent is rewarded to keep the pole upright, whereas it receives a constraint if (1) the cart enters certain areas (x-axis position), or (2) the angle of the pole is smaller than a certain threshold.

Table 1: Algorithms hyperparameters for model-based gridworld environment ($\gamma = 0.9$).

| Algorithm | $\alpha_\pi$ | $\alpha_\lambda$ | $m$ | $\eta$ |
|---|---|---|---|---|
| sMDPO | 1 | 0.01 | 10 | 10 |
| GDA | 0.1 | 0.01 | - | - |
| CBP | - | 100 | - | - |

The performance of methods is reported in terms of optimality gap and constraint violation defined for a policy $\pi$ as below:

$$\text{optimality gap (OG)} = J_r(\pi^*) - J_r(\pi) \qquad \text{constraint violation (CV)} = b - J_c(\pi) \tag{9}$$

Also, for all methods, we used grid search to find the best hyperparameter combination. For the first environment, we considered tabular parameterization of policy and for the second environment, we considered linear parameterization. In the following sections, we asses the performance of methods in tabular and linear parameterization respectively, and finally provide some ablation studies to obtain a better insight into the source of our method's gain.

### 5.1 Tabular Setting

**Model-based tabular parameterization for gridworld**. We consider a model-based setting, where we have complete knowledge of the CMDP, and thus estimation errors do not affect performance of algorithms. For each algorithm, we report the performance with $\gamma = 0.9$ (original discount factor) and $\gamma = 0.8$ (included to measure the robustness with respect to environment misspecification) averaged across 5 runs with 95 % confidence interval in Figure 3. The best hyperparameter combination for each method is reported in Table 1 and refers to the one with CV $\in \{-0.25, 0\}$ and with the least OG.

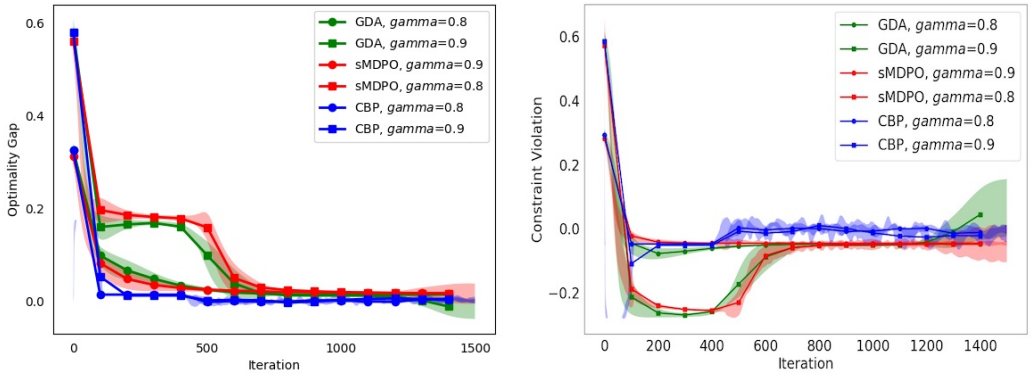



Figure 1: Performance of algorithms on model-based gridworld environment

.

**Model-free tabular parameterization for gridworld**. In the second experiment, we consider the model-free setting and approximate the Q functions with TD sampling. We report the performance of algorithms with [1000, 2000, 3000] number of samples used for approximation averaged across 5 runs with 95 % confidence interval. Similar to the previous section, the best hyperparameter combination for each method is reported in Table 2 and refers to the one with CV $\in \{-0.25, 0\}$ and with the least optimality gap.

In both experiments with the tabular parametrization, sMDPO converges later than the two other algorithms, however, they all converge to the similar optimality gap value, and in most cases satisfy the constraint.

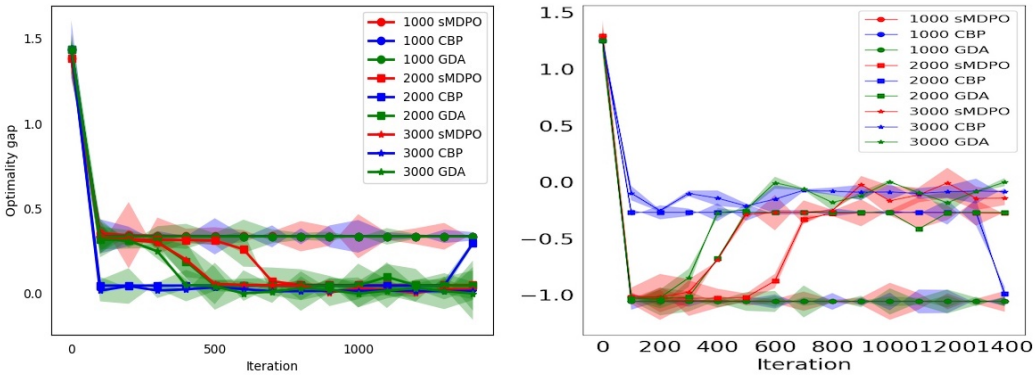



Figure 2: Performance of algorithms on model-free gridworld environment

.

Table 2: Algorithms hyperparameters for model-free gridworld environment (number of samples for Q estimation= 3000).

| Algorithm | $\alpha_\pi$ | $\alpha_\lambda$ | $m$ | $\eta$ |
|---|---|---|---|---|
| sMDPO | 1 | 0.1 | 100 | 0.1 |
| GDA | 0.1 | 0.1 | - | - |
| CBP | - | 5 | - | - |

### 5.2 Linear Setting

**Linear parameterization for CartPole**. We perform linear function approximation on CartPole environment, and compare three methods. We used tile coding Sutton and Barto [2018] to construct the feature space, and LSTDQ Lagoudakis and Parr [2003] to estimate the Q functions for both reward and constraint reward. The best hyperparameter combination is shown in Table 3 and refers to the one that achieves the maximum return, while satisfying CV $\in \{-6, 0\}$ for both constraints. All three algorithms satisfy the constraints, achieve comparable reward, but sMDPO converges faster than the other two.

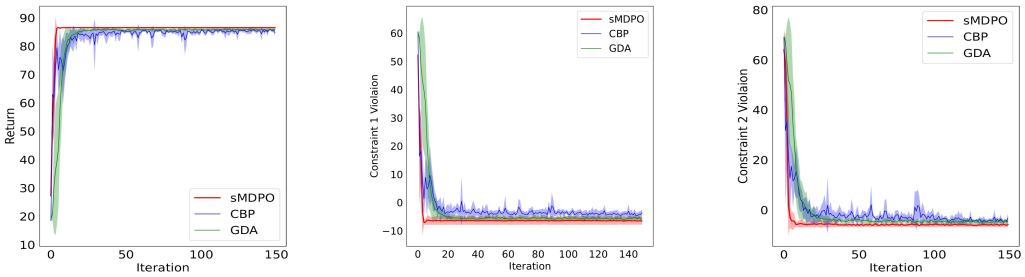



Figure 3: Performance of algorithms on CartPole environment with LFA

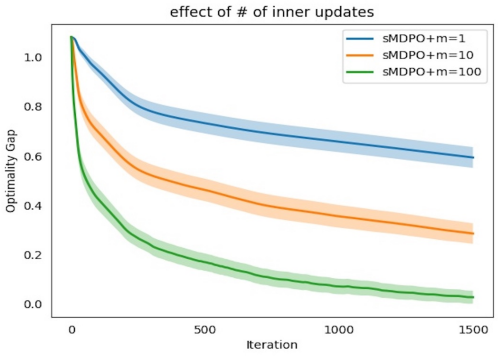
.

### 5.3 Ablation Study

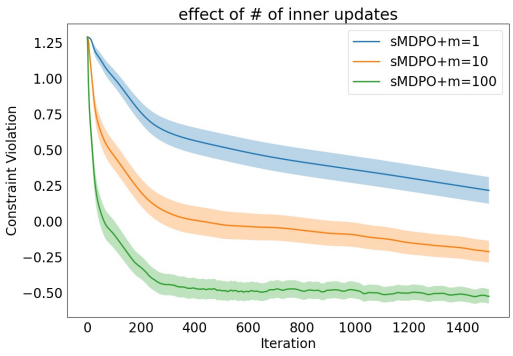
In this section, we analyze the effect of sMDPO inner loops to gain a better insight about its performance. It is shown that increasing the number of inner loops improves the performance of sMDPO in tabular parametrization.

Table 3: Algorithms hyperparameters for CartPole environment.

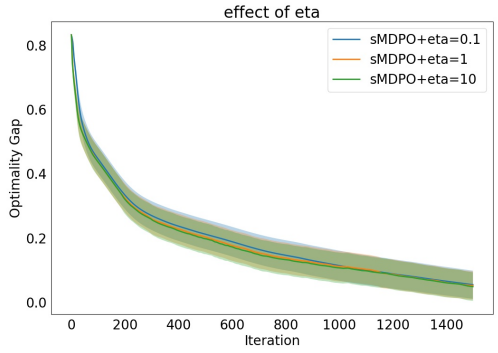
| Algorithm | $\alpha_\pi$ | $\alpha_\lambda$ | $m$ | $\eta$ |
|---|---|---|---|---|
| sMDPO | 1 | 10 | 10 | 1 |
| GDA | 0.001 | 0.1 | - | - |
| CBP | - | 50 | - | - |



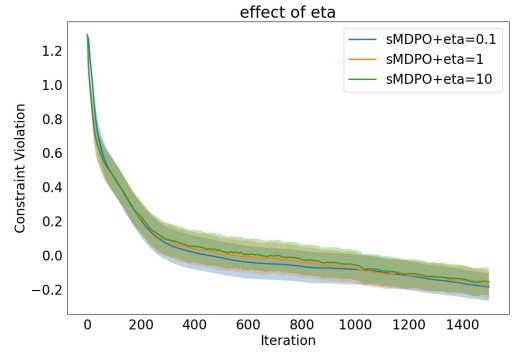

(a) OG - model-based Gridworld




(b) CV - model-based Gridworld




(c) OG - model-based Gridworld




(d) CV - model-based Gridworld

Figure 4: Ablation study on number of updates and divergence regularization

*Unfortunately, we have not had enough amount of time to perform ablation study in other settings.*

## 6 Conclusion

In this project, we used a surrogate function instantiated from FMA-PG frmework Vaswani et al. [2021] to solve the constrained reinforcement learning problem. The surrogate function supports off-policy updates and allows us to update the policy parameters efficiently with regards to agent interaction with the environment. We evaluated the algorithm in tabular and linear function approximation setting, and also compared it with two state-of-the-art algorithms in constrained RL. This project can be extended for future works in two perspectives: 1) The monotonic improvement is better to be on total expected return $J_r$. 2) Neural networks can be used as parameterization of policy and experiments on high-dimensional state-action spaces can be conducted to asses sMDPO in such settings. Also, sMDPO should be compared with a broader range of constrained policy optimization algorithms.

## 7 Author Contributions

AK was responsible for writing the final report, preparing a presentation about FMA-PG framework and experiments, and the required implementations on the CartPole environment. AA was responsible for the required implementations of the Gridworld environment, preparing a presentation on introduction of RL and CMDP problem, and performing the second round of experiments (comments on the day of presentation). Authors studied the related works, did the literature review, and also wrote the proposal together. Finally, we gratefully acknowledge support from Sharan Vaswani during this project.

## References


Pieter Abbeel, Adam Coates, Morgan Quigley, and Andrew Ng. An application of reinforcement learning to aerobatic helicopter flight. *Advances in neural information processing systems*, 19, 2006.

Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Remi Munos, Nicolas Heess, and Martin Riedmiller. Maximum a posteriori policy optimisation, 2018. URL `https://arxiv.org/abs/1806.06920`.

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization, 2017. URL `https://arxiv.org/abs/1705.10528`.

Eitan Altman. *Constrained Markov decision processes: stochastic modeling*. Routledge, 1999a.

Eitan Altman. *Constrained Markov decision processes: stochastic modeling*. Routledge, 1999b.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016. URL `https://arxiv.org/abs/1606.01540`.

Yinlam Chow, Ofir Nachum, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. A lyapunov-based approach to safe reinforcement learning, 2018. URL `https://arxiv.org/abs/1805.07708`.

Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. Safe exploration in continuous action spaces, 2018. URL `https://arxiv.org/abs/1801.08757`.

G Alphastar DeepMind. Mastering the real-time strategy game starcraft ii, 2019.

Dongsheng Ding, Kaiqing Zhang, Tamer Basar, and Mihailo Jovanovic. Natural policy gradient primal-dual method for constrained markov decision processes. *Advances in Neural Information Processing Systems*, 33:8378–8390, 2020.

Arushi Jain, Sharan Vaswani, Reza Babanezhad, Csaba Szepesvari, and Doina Precup. Towards Painless Policy Optimization for Constrained MDPs. *arXiv e-prints*, art. arXiv:2204.05176, April 2022.

Sham M Kakade. A natural policy gradient. *Advances in neural information processing systems*, 14, 2001.

Michail G Lagoudakis and Ronald Parr. Least-squares policy iteration. *The Journal of Machine Learning Research*, 4:1107–1149, 2003.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies, 2015. URL `https://arxiv.org/abs/1504.00702`.

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, 2015. URL `https://arxiv.org/abs/1509.02971`.

Yongshuai Liu, Jiaxin Ding, and Xin Liu. Ipo: Interior-point policy optimization under constraints. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34:4940–4947, 04 2020. doi: 10.1609/aaai.v34i04.5932.

Santiago Paternain, Luiz F. O. Chamon, Miguel Calvo-Fullana, and Alejandro Ribeiro. Constrained reinforcement learning has zero duality gap, 2019. URL https://arxiv.org/abs/1910.13393.

John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust region policy optimization, 2015. URL https://arxiv.org/abs/1502.05477.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Thomas Spooner and Rahul Savani. A natural actor-critic algorithm with downside risk constraints, 2020. URL https://arxiv.org/abs/2007.04203.

Adam Stooke, Joshua Achiam, and Pieter Abbeel. Responsive safety in reinforcement learning by pid lagrangian methods, 2020. URL https://arxiv.org/abs/2007.03964.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In S. Solla, T. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999. URL https://proceedings.neurips.cc/paper/1999/file/464d828b85b0bed98e80ade0a5c43b0f-Paper.pdf.

Chen Tessler, Daniel J. Mankowitz, and Shie Mannor. Reward constrained policy optimization, 2018. URL https://arxiv.org/abs/1805.11074.

Sharan Vaswani, Olivier Bachem, Simone Totaro, Robert Mueller, Shivam Garg, Matthieu Geist, Marlos C. Machado, Pablo Samuel Castro, and Nicolas Le Roux. A general class of surrogate functions for stable and efficient reinforcement learning, 2021. URL https://arxiv.org/abs/2108.05828.

Tengyu Xu, Yingbin Liang, and Guanghui Lan. Crpo: A new approach for safe reinforcement learning with convergence guarantee, 2020. URL https://arxiv.org/abs/2011.05869.

Ming Yu, Zhuoran Yang, Mladen Kolar, and Zhaoran Wang. Convergent policy optimization for safe reinforcement learning, 2019. URL https://arxiv.org/abs/1910.12156.