# Towards Painless Policy Optimization for Constrained MDPs


## Abstract

We study policy optimization in an infinite horizon, $\gamma$-discounted constrained Markov decision process (CMDP). Our objective is to return a policy that achieves large expected reward with a small constraint violation. We consider the online setting with linear function approximation and assume global access to the corresponding features. We propose a generic primal-dual framework that allows us to bound the reward sub-optimality and constraint violation for arbitrary algorithms in terms of their primal and dual regret on online linear optimization problems. We instantiate this framework to use coin-betting algorithms and propose the **Coin Betting Politex (CBP)** algorithm. Assuming that the action-value functions are $\varepsilon_{\mathrm{b}}$-close to the span of the $d$-dimensional state-action features and no sampling errors, we prove that $T$ iterations of CBP result in an $O\left(\frac{1}{(1-\gamma)^3\sqrt{T}} + \frac{\varepsilon_{\mathrm{b}}\sqrt{d}}{(1-\gamma)^2}\right)$ reward sub-optimality and an $O\left(\frac{1}{(1-\gamma)^2\sqrt{T}} + \frac{\varepsilon_{\mathrm{b}}\sqrt{d}}{1-\gamma}\right)$ constraint violation. Importantly, unlike gradient descent-ascent and other recent methods, CBP does not require extensive hyperparameter tuning. Via experiments on synthetic and Cartpole environments, we demonstrate the effectiveness and robustness of CBP.

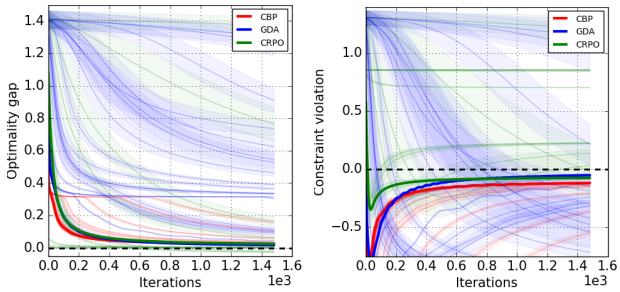

## 1 INTRODUCTION

Popular reinforcement learning (RL) algorithms focus on optimizing an unconstrained objective, and have found applications in games such as Atari (Mnih et al., 2015) or Go (Silver et al., 2016), robot manipulation tasks (Tan et al., 2018; Zeng et al., 2020) or clinical trials (Schaefer et al., 2005). However, many applications require the planning agent to satisfy constraints – for example, in wireless sensor networks (Buratti et al., 2009; Julian et al., 2002) there is a constraint on average power consumption of a deployed policy. Similarly, in safe RL, the policy is constrained to only visit certain states while exploring in physical systems (Moldovan and Abbeel, 2012; Ono et al., 2015; Fisac et al., 2018). The constrained Markov decision process (CMDP) (Altman, 1999) is a natural framework to model long-term constraints that need to be satisfied by a policy. The typical objective for CMDPs is to maximize the cumulative reward (similar to unconstrained MDPs), while (approximately) satisfying the constraint.

We focus on a well-studied problem in CMDPs – return an approximately feasible policy (that is allowed to violate the constraints by a small amount), while (approximately) maximizing the cumulative reward. The past literature on this topic considered two approaches. The first approach is *primal-only algorithms*, where constraints are (approximately) enforced without directly relying on introducing a Lagrangian formulation (Achiam et al., 2017; Chow et al., 2018; Dalal et al., 2018; Liu et al., 2020; Xu et al., 2021). Of these methods, only the recent work of Xu et al. (2021) guarantees global convergence to the optimal feasible policy in both the tabular and function approximation settings.

The second approach in CMDPs is to form the Lagrangian, and solve the resulting saddle-point problem using *primal-dual algorithms* (Altman, 1999; Borkar, 2005; Bhatnagar and Lakshmanan, 2012; Borkar and Jain, 2014; Tessler et al., 2018; Liang et al., 2018; Paternain et al., 2019; Yu et al., 2019; Ding et al., 2021, 2020; Stooke et al., 2020). Such approaches update both the policy parameters (primal variables), while updating the Lagrange multipliers (dual variables). Of these methods, Tessler et al. (2018) prove a local convergence guarantee, while Paternain et al. (2019) prove

*Submitted to the 38th Conference on Uncertainty in Artificial Intelligence* (UAI 2022). **To be used for reviewing only**.




(a) Optimality gap (OG) (b) Constraint violation (CV)

Figure 1: **Hyperparameter sensitivity:** Optimality gap and constraint violation (averaged across 5 runs) for different hyperparameters for **GDA** (Ding et al., 2020), **CRPO** (Xu et al., 2021) and the proposed algorithm **CBP** on a gridworld environment with access to the true CMDP. The dark lines show the performance of the best hyperparameter, while the lighter-shade lines represent results while using other hyperparameters. Both GDA and CRPO exhibit large variations in their performance, while CBP is more robust. See Section 6 for details.

that their proposed algorithm will converge to a neighbourhood of the optimal policy. More recently, Ding et al. (2020) proposed to use natural policy gradient updates (Kakade, 2001) for changing the policy parameters while using gradient descent to update the dual variables. They prove that this primal-dual algorithm converges to the optimal policy in both the tabular and the function approximation settings.

Although there is no lack in algorithms designed for CMDPs, *these algorithms are often highly sensitive to the choice of their hyperparameters*. For example, Fig. 1 demonstrates the effect of varying the hyperparameters for two provably efficient algorithms, the primal-dual natural-policy ascent, gradient descent method (in short, GDA) of Ding et al. (2020) and the primal-only CRPO method of Xu et al. (2021) on a synthetic tabular environment. While one can find hyperparameters that control the worst-case performance of either GDA or CRPO, such choices result in a poor empirical performance on individual instances, a feature that GDA and CRPO share with unconstrained MDP policy optimization algorithms, such as Politex (Abbasi-Yadkori et al., 2019), or natural policy gradient (Kakade, 2001).

**CONTRIBUTIONS:** *Designing robust policy optimization algorithms that require minimal hyperparameter tuning is our main motivation, and towards this, we make the following contributions.*

**Generic Primal-Dual Framework**: In Section 3, we cast the problem of planning in discounted infinite horizon CMDPs to a generic primal-dual framework. In particular, we prove that any algorithm that can control (i) the primal and dual regret for specific online linear optimization problems and (ii) the errors due to function approximation and sampling, will (approximately) maximize the cumulative discounted reward while (approximately) minimizing the constraint violation (Theorem 3.1). Importantly, this result holds for any CMDP and is independent of how the policies or value functions are represented.

**Instantiating the Framework**: In Section 4, we instantiate the framework using two algorithms from the online linear optimization literature – Gradient Descent Ascent (GDA) (Section 4.2.1) and Coin-Betting (CB) (Section 4.2.2). While GDA requires setting the hyperparameters to specific problem-dependent constants, CB is more robust to hyperparameter tuning (see Fig. 1). In the simpler tabular setting, the approximation errors can be easily controlled and we use Theorem 3.1 in conjunction with existing regret bounds to prove that the average optimality gap (difference in the cumulative reward of achieved policy and the optimal policy) and the average constraint violation decrease at an $O\left(1/\sqrt{T}\right)$ rate (Corollaries A.1 and A.2).

**Handling Linear Function Approximation**: In Section 5, we assume global access to a $d$-dimensional feature map $\Phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$, and that the action-value functions for any policy are $\varepsilon_{\text{b}}$-close to the span of these features. With this assumption, we prove that it is possible to control the approximation errors for each state-action pair. Subsequently, in Section 5.1, we use the robust coin-betting algorithms to instantiate the primal-dual framework in the linear function approximation setting and propose the *Coin-Betting Politex* (CBP) algorithm. Ignoring sampling errors, in Section 5.1.1, we prove that the average optimality gap for CBP scales as $O\left(\frac{1}{(1-\gamma)^3\sqrt{T}} + \frac{\varepsilon_{\text{b}}\sqrt{d}}{(1-\gamma)^2}\right)$, while the average constraint violation is $O\left(\frac{1}{(1-\gamma)^2\sqrt{T}} + \frac{\varepsilon_{\text{b}}\sqrt{d}}{(1-\gamma)}\right)$. With linear function approximation, the average constraint violation for the algorithm of Ding et al. (2020) decreases at a worse $O\left(1/T^{1/4}\right)$ rate. On the other hand, the CRPO algorithm of Xu et al. (2021) results in an $O\left(1/\sqrt{T}\right)$ bound for both the average suboptimality and constraint violation. However, both algorithms can amplify the function approximation errors to large, potentially unbounded values. Importantly, both algorithms require typically unknown quantities which impedes their practical use.

**Experimental Evaluation**: In Section 6, we first describe some practical considerations when implementing CBP. We then evaluate CBP and compare its empirical performance to the algorithms of Ding et al. (2020); Xu et al. (2021). Our experiments on synthetic tabular environment and the Cartpole environment

with linear function approximation demonstrate the consistent effectiveness and robustness of CBP.

## 2 PROBLEM FORMULATION

We consider an infinite-horizon discounted constrained Markov decision process (CMDP) (Altman, 1999) defined by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, r, c, b, \rho, \gamma \rangle$ where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the action set, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \Delta_{\mathcal{S}}$ is the transition probability function, $\rho \in \Delta_{\mathcal{S}}$ is the initial distribution of states and $\gamma \in [0, 1)$ is the discount factor. The primary reward to be maximized is denoted by $r : \mathcal{S} \times \mathcal{A} \to [0, 1]$. For each state $s$, we define the reward value function w.r.t. the policy $\pi : \mathcal{S} \to \Delta_{\mathcal{A}}$ as $V_r^\pi(s) = \mathbb{E}_{\pi,\mathcal{P}}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | s_0 = s\right]$ where $a_t \sim \pi(\cdot|s_t)$, and $s_{t+1} \sim \mathcal{P}(\cdot|s_t, a_t)$. The expected discounted return or *reward value* of a policy $\pi$ is defined as $V_r^\pi(\rho) = \mathbb{E}_{s_0\sim\rho}\left[V_r^\pi(s_0)\right]$. Similarly, the constraint reward is denoted by $c : \mathcal{S} \times \mathcal{A} \to [0, 1]$ and the *constraint reward value* for $\pi$ by $V_c^\pi(\rho)$. For each $(s, a)$ under policy $\pi$, the reward action-value function is defined as $Q_r^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ s.t. $Q_r^\pi(s, a) = r(s, a) + \gamma \mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}[V_r^\pi(s')]$ and satisfies the relation: $V_r^\pi(s) = \langle \pi(\cdot|s), Q_r^\pi(s, \cdot)\rangle (:= \mathbb{E}_{a\sim\pi(\cdot|s)}[Q_r^\pi(s, a)])$. We define $Q_c^\pi$ analogously. The agent's objective is to return a policy $\pi$ that maximizes $V_r^\pi(\rho)$, while ensuring that $V_c^\pi(\rho) \geq b$. Formally,

$$\max_\pi V_r^\pi(\rho) \quad \text{s.t.} \quad V_c^\pi(\rho) \geq b. \tag{1}$$

Throughout, we will assume the existence of a feasible policy (i.e., one with $V_c^\pi(\rho) \geq b$), and denote the optimal feasible policy by $\pi^*$. Due to sampling and other errors, we will aim for finding a policy $\pi$ such that,

$$V_r^\pi(\rho) \geq V_r^{\pi^*}(\rho) - \varepsilon \quad \text{s.t.} \quad V_c^\pi(\rho) \geq b - \varepsilon \tag{2}$$

with some $\varepsilon > 0$. In the next section, we specify a generic primal-dual framework solving the problem in Eq. (2).

## 3 PRIMAL-DUAL FRAMEWORK

By Lagrangian duality, $\pi^*$ is a solution to Eq. (1) if and only if for some $\lambda^* \geq 0$, $(\pi^*, \lambda^*)$ solves the saddle-point problem

$$\max_\pi \min_{\lambda\geq 0} V_r^\pi(\rho) + \lambda[V_c^\pi(\rho) - b]\,. \tag{3}$$

Here, $\lambda \in \mathbb{R}$ is the Lagrange multiplier for the constraint.

We will solve the above primal-dual saddle-point problem iteratively, by alternatively updating the policy (primal variable) and the Lagrange multiplier (dual variable). If $T$ is the total number of iterations, we define $\pi_t$ and $\lambda_t$ to be the primal and dual iterates for $t \in [T] := \{1, \ldots, T\}$. Updating the $(\pi_t, \lambda_t)$ variables will require estimating the action-value functions. We define $\hat{Q}_r^t := \hat{Q}_r^{\pi_t}$ and $\hat{Q}_c^t := \hat{Q}_c^{\pi_t}$ as the *estimated* action-value functions corresponding to the policy $\pi_t$. We also define $\hat{V}_c^\pi(s) := \langle \pi(\cdot|s), \hat{Q}_c^t(s, \cdot)\rangle$, $V_r^t(\rho) := V_r^{\pi_t}(\rho)$ and $V_c^t(\rho) := V_c^{\pi_t}(\rho)$. In this section, we assume that $\left\|Q_r^t - \hat{Q}_r^t\right\|_\infty \leq \tilde{\varepsilon}$ and $\left\|Q_c^t - \hat{Q}_c^t\right\|_\infty \leq \tilde{\varepsilon}$.

Given a generic primal-dual algorithm, our task is to characterize its performance in terms of its cumulative reward and constraint violation. Specifically, for a sequence of policies $\{\pi_0, \pi_1, \ldots, \pi_{T-1}\}$ and Lagrange multipliers $\{\lambda_0, \lambda_1, \ldots, \lambda_{T-1}\}$ generated by an algorithm, we define the *average optimality gap* (`OG`) and the *average constraint violation* (`CV`) as,

$$\text{Average optimality gap } (\texttt{OG}) := \frac{1}{T}\sum_{t=0}^{T-1}[V_r^{\pi^*}(\rho) - V_r^t(\rho)],$$

$$\text{Average constraint violation } (\texttt{CV}) := \frac{1}{T}\left[\sum_{t=0}^{T-1} b - V_c^t(\rho)\right]_+,$$

where $[x]_+ = \max\{x, 0\}$. For this algorithm, we define the *primal regret* and *dual regret* as follows:

$$\mathcal{R}^p(\pi^*, T) := \sum_{t=0}^{T-1} \langle \pi^*(\cdot|s) - \pi_t(\cdot|s), \hat{Q}_r^t + \lambda_t \hat{Q}_c^t]\rangle_{s\sim\nu_{\rho,\pi^*}},$$

$$\mathcal{R}^d(\lambda, T) := \sum_{t=0}^{T-1} (\lambda_t - \lambda)\left(\hat{V}_c^t(\rho) - b\right). \tag{4}$$

Here, $\langle f, g\rangle_{s\sim\nu_{\rho,\pi^*}} = \mathbb{E}_{s\sim\nu_{\rho,\pi^*}}[f(s)g(s)]$ and $\nu_{\rho,\pi^*}$ is the discounted occupation measure induced by following $\pi^*$ from $\rho$ normalized so that it becomes a probability measure. Observe that the above quantities correspond to the regret for online linear optimization algorithms that can independently update the primal and dual variables. Our main result (proved in Appendix B) in this section characterizes the performance of a generic algorithm in terms of its primal and dual regret.

**Theorem 3.1.** *Assuming that* $\left\|Q_r^t - \hat{Q}_r^t\right\|_\infty \leq \tilde{\varepsilon}$ *and* $\left\|Q_c^t - \hat{Q}_c^t\right\|_\infty \leq \tilde{\varepsilon}$, *for a generic algorithm producing a sequence of polices* $\{\pi_0, \pi_1, \ldots, \pi_{T-1}\}$ *and dual variables* $\{\lambda_0, \lambda_1, \ldots, \lambda_{T-1}\}$ *such that for all* $t$, $\lambda_t$ *is constrained to lie in the* $[0, U]$ *where* $U > \lambda^*$, `OG` *and* `CV` *can be bounded as:*

$$\texttt{OG} \leq \frac{\mathcal{R}^p(\pi^*, T) + (1-\gamma)\mathcal{R}^d(0, T)}{(1-\gamma)T} + \tilde{\varepsilon}\, g(U),$$

$$\texttt{CV} \leq \frac{\mathcal{R}^p(\pi^*, T) + (1-\gamma)\mathcal{R}^d(U, T)}{(U-\lambda^*)(1-\gamma)T} + \frac{\tilde{\varepsilon}\, g(U)}{(U-\lambda^*)},$$

*where* $g(U) := \left[\frac{1+U}{1-\gamma} + U\right]$.

We note that such a general primal-dual regret decomposition for convex MDPs (including CMDPs) was recently done by Zahavy et al. (2021). However, they handle the tabular setting where the primal variables correspond to state-action occupancy measures, whereas, the above result defines the primal variables to be the policy parameters. More importantly, our result does it require any assumption about the underlying CMDP. In the unconstrained setting, reducing the policy optimization problem to that of online linear optimization has been previously explored in the *Politex* algorithm (Abbasi-Yadkori et al., 2019), and we build upon this work.

In order to bound the average reward optimality gap and the average constraint violation, we need to (i) project the dual variables onto the $[0, U]$ interval and ensure that $U > \lambda^*$, (ii) update the primal and dual variables to control the respective regret in Eq. (4), and (iii) control the approximation error $\tilde{\varepsilon}$. Next, we use this recipe to design algorithms with provable guarantees.

# 4 INSTANTIATING THE FRAMEWORK

In this section, we will instantiate the primal-dual framework by using the above technique – specifying the value of $U$ in Section 4.1 and describing algorithms that control the primal and dual regret in Section 4.2.

## 4.1 UPPER-BOUND FOR DUAL VARIABLES

In Appendix B, we prove the following upper-bound on the optimal dual variable

**Lemma 4.1.** *The objective Eq. (1) satisfies strong duality, and the optimal dual variables are bounded as* $\lambda^* \leq \frac{1}{\zeta(1-\gamma)}$, *where* $\zeta := \max_\pi V_c^\pi(\rho) - b > 0$.

Unlike Ding et al. (2020, 2021) who bound the dual variables in terms of the unknown Slater constant, the upper-bound from Lemma 4.1 can be computed by maximizing the constraint value function as an unconstrained problem. Throughout, we will set $U = \frac{2}{\zeta\,(1-\gamma)}$ that satisfies the requirement $U > \lambda^*$ and projects the dual variables onto $[0, U]$ range (in Section 3).

## 4.2 CONTROLLING THE PRIMAL AND DUAL REGRET

In this section, we specify two algorithms to update the primal and dual variables, and control the primal and dual regret respectively. In particular, in Section 4.2.1, we will use mirror ascent to update the primal variables, and gradient descent to update the dual variables. Inspired by the literature on online linear optimization (Orabona and Pal, 2016), we will use robust, parameter-free algorithms to update the primal and dual variables in Section 4.2.2.

### 4.2.1 Gradient descent ascent

At iteration $t \in [T]$, if the primal and dual iterates are $\pi_t$ and $\lambda_t$ respectively, given $\hat{Q}_r^t$ and $\hat{Q}_c^t$, the gradient descent ascent (GDA) update can be written as follows: if $\hat{Q}_l^t(s,a) = \hat{Q}_r^t(s,a) + \lambda_t \hat{Q}_c^t(s,a)$ and $\hat{V}_c^t(\rho) = \sum_{s\in\mathcal{S}} \rho(s) \sum_{a\in\mathcal{A}} \pi_t(a|s) \hat{Q}_c^t(s,a)$, then,

$$\pi_{t+1}(a|s) = \frac{\pi_t(a|s)\exp\left(\eta_1 \hat{Q}_l^t(s,a)\right)}{\sum_{a'} \pi_t(a'|s)\exp\left(\eta_1 \hat{Q}_l^t(s,a')\right)} \tag{5}$$

$$\lambda_{t+1} = \mathbb{P}_{[0,U]}[\lambda_t - \eta_2\,(\hat{V}_c^t(\rho) - b)]. \tag{6}$$

Here $\mathbb{P}_{[a,b]}$ is a projection onto the $[a, b]$ interval, and $\eta_1$ and $\eta_2$ are the step-size parameters for the primal and dual updates respectively. In the tabular setting, the resulting algorithm is the same as that analyzed by Ding et al. (2020).

Analyzing the primal and dual regret for the above updates is fairly standard in online linear optimization. Using results from the paper of Orabona (2019), by setting $\eta_1 = \sqrt{\frac{2\log|\mathcal{A}|}{t}}\,\frac{1-\gamma}{1+U}$, $\eta_2 = \frac{U(1-\gamma)}{\sqrt{t}}$ and $U = \frac{2}{\zeta\,(1-\gamma)}$, we get $\mathcal{R}^p(\pi^*, T) \leq \frac{1+U}{1-\gamma}\sqrt{2\log|\mathcal{A}|}\sqrt{T}$ and $\mathcal{R}^d(\lambda, T) \leq \frac{U}{1-\gamma}\sqrt{T}$. Observe that both the primal and dual regret scale as $O(\sqrt{T})$, and using Theorem 3.1, both the average optimality gap and constraint violation will decrease at an $O(1/\sqrt{T})$ rate.

We also note that obtaining the above bounds requires setting the two step-sizes ($\eta_1$ and $\eta_2$) to specific values that depend on problem-dependent parameters. In Fig. 1, we have seen that GDA is quite sensitive to the values of $\eta_1$ and $\eta_2$, even in the simple tabular setting. In order to alleviate this, we use the recent progress in online linear optimization, and propose robust algorithms in the next section.

### 4.2.2 Coin-betting

Orabona and Pal (2016) and Orabona and Tommasi (2017) propose *coin-betting* algorithms that reduce the online linear optimization problems in Eq. (4) to online betting. Unlike adaptive gradient methods like AdaGrad (Duchi et al., 2011) or Adam (Kingma and Ba, 2014) that require setting the initial step-size, coin-betting algorithms are completely parameter-free. We will directly instantiate the regret-minimization algorithms from these works. First, we instantiate the algorithm of Orabona and Pal (2016) for updating the

policy (primal variables) in the CMDP setting. In order to do this, we define additional variables $w_t$ for each $(s,a)$ pair and iteration $t$. These variables will be computed recursively, and used to compute the policy $\pi_{t+1}$ at iteration $t$. In particular, for $t \geq 1$,

$$w_{t+1}(s,a) = \frac{\sum_{i=0}^{t} \tilde{A}_l^i(s,a)}{(t+1)+T/2}\left(1+\sum_{i=0}^{t}\tilde{A}_l^i(s,a)\, w_i(s,a)\right)$$
$$\pi_{t+1}(a|s) = \begin{cases} \pi_0(a|s), & \text{if } \sum_a \pi_0(a|s)\,[w_{t+1}(s,a)]_+ = 0 \\ \frac{\pi_0(a|s)\,[w_{t+1}(s,a)]_+}{\sum_{a'}\pi_0(a'|s)\,[w_{t+1}(s,a')]_+}, & \text{otherwise} \end{cases} \tag{7}$$

where, given $\pi_t$, $\tilde{A}_l^t(s,a)$ is equal to

$$\hat{A}_l^t(s,a)\,\mathcal{I}\{w_t(s,a)>0\} + [\hat{A}_l^t(s,a)]_+\,\mathcal{I}\{w_t(s,a)\leq 0\}$$

and $\hat{A}_l^t(s,a) = \frac{1-\gamma}{1+U}\left[\hat{Q}_l^t(s,a) - \left\langle \hat{Q}_l^t(s,\cdot), \pi_t(\cdot|s)\right\rangle\right]$. For the above calculation, we use the normalized (by $\frac{1+U}{1-\gamma}$) action-value functions that are ensured to lie in the $[0,1]$ range. The quantity $\hat{A}_l^t(s,a)$ can be interpreted as the (normalized) advantage function for policy $\pi_t$ in the unconstrained MDP with rewards equal to $r(s,a)+\lambda_t\, c(s,a)$. Observe that the above update does not have any tunable hyperparameters.

Similarly, we use the coin-betting algorithm of Orabona and Tommasi (2017) to update the Lagrange multipliers, instantiating it in the CMDP setting: for $t \geq 1$, if $\sigma(x) := \frac{1}{1+\exp(-x)}$, then,

$$\lambda_{t+1} = \lambda_0 - \beta_t\left[\frac{1}{1-\gamma} - \sum_{i=0}^{t}(\lambda_i-\lambda_0)\,(\hat{V}_c^{\pi_i}(\rho)-b)\right],$$
$$\beta_t = (1-\gamma)\left(2\sigma\left(\frac{2\sum_{i=0}^{t}(\hat{V}_c^{\pi_i}(\rho)-b)}{\frac{1}{1-\gamma}+\sum_{i=0}^{t}|\hat{V}_c^{\pi_i}(\rho)-b|}-1\right)\right). \tag{8}$$

Similar to the primal update, the dual update uses normalized (by $1/1-\gamma$) value functions that lie in the $[-1,1]$ range, and does not have any tunable parameter. Importantly, these updates result in *no-regret* algorithms meaning that both the primal and dual regret scale as $o(T)$. Specifically, for the primal updates in Eq. (7) and the dual updates in Eq. (8), Orabona and Tommasi (2017) prove that,

$$\mathcal{R}^p(\pi^*,T) \leq \frac{3(1+U)}{1-\gamma}\sqrt{T}\sqrt{1+\mathrm{KL}(\pi_0||\pi^*)},$$

$$\mathcal{R}^d(\lambda,T) \leq \frac{1}{1-\gamma} + \|\lambda-\lambda^0\|\sqrt{\left(\frac{1}{(1-\gamma)^2}+\frac{G_T}{1-\gamma}\right)\Gamma_T},$$

where $\Gamma_T = \log\left(1+(G_T\,(1-\gamma)+1)^2\,\|\lambda-\lambda^0\|^2\right)$ and $G_T = \sum_{i=0}^{T}|\hat{V}_c^{\pi_i}(\rho)-b| = O(T)$. Since both regrets scale as $O(\sqrt{T})$ in the worst case, using the coin-betting updates will also result in an $O(1/\sqrt{T})$ decrease in both the average optimality gap and constraint violation. Unlike the updates in Section 4.2.1, the coin-betting updates do not require tuning a hyperparameter.

If we can control the approximation errors, we can use the above algorithms to completely instantiate the primal-dual framework. In Appendix A, we do this for the simpler tabular setting, and consider the linear function approximation setting in the next section.

# 5 PUTTING EVERYTHING TOGETHER

In this section, we will bound the approximation errors in the linear function approximation setting and instantiate the above framework.

In order to scale to large state-action spaces, we consider the special case of linear function approximation and assume *global* access to a $d$-dimensional feature map $\Phi : \mathcal{S}\times\mathcal{A} \to \mathbb{R}^d$. Given $\Phi$, we make the following (approximate) realizability assumption on action-value functions (Abbasi-Yadkori et al., 2019).

**Assumption 5.1** (Linear function approximation)**.** With global access to the feature map $\Phi$, the action-value functions for each memoryless policy $\pi$ are $\varepsilon_{\text{b}}$-close to the span of the state-action features.

$$\inf_{\theta\in\mathbb{R}^d}\max_{(s,a)}|Q_r^\pi(s,a)-\langle\theta,\phi(s,a)\rangle| \leq \varepsilon_{\text{b}}$$
$$\inf_{\theta\in\mathbb{R}^d}\max_{(s,a)}|Q_c^\pi(s,a)-\langle\theta,\phi(s,a)\rangle| \leq \varepsilon_{\text{b}}$$

This setting subsumes the tabular case which can be recovered (with $\varepsilon_{\text{b}}=0$) when $d=|\mathcal{S}|\,|\mathcal{A}|$, and the feature-map consisting of one-hot vectors for each state-action pair. Given a good estimate of $\theta_r^\pi := \arg\min\left[\max_{(s,a)}|Q_r^\pi(s,a)-\langle\theta,\phi(s,a)\rangle|\right]$, we can easily estimate the action-value functions for every $(s,a)$ pair as $Q_r^\pi(s,a)\approx\langle\theta_r^\pi,\phi(s,a)\rangle$. A naive way to estimate $\theta_r^\pi$ is to form a subset $\mathcal{C}\subseteq\mathcal{S}\times\mathcal{A}$ of $(s,a)$ pairs, rollout $m$ independent trajectories using policy $\pi$ and starting from each $(s,a)\in\mathcal{C}$. The average (across trajectories) cumulative discounted return is an unbiased estimate $q_r(s,a)$ of the action-value function. If $q_r$ is defined to be the $|\mathcal{C}|$-dimensional vector of estimated action-value functions, and for a fixed set of weights $\omega$ s.t. $\omega(s,a)\geq 0$ and $\sum_{(s,a)\in\mathcal{C}}\omega(s,a)=1$, we use the weighted-least squares estimate with $z:=(s,a)$,

$$\hat{\theta}_r^\pi = \arg\min_\theta \sum_{z\in\mathcal{C}}\omega(z)\left[\langle\theta,\phi(z)\rangle - q_r(z)\right]^2. \tag{9}$$

For the $(s,a)\in\mathcal{C}$, the sampling error is $O(1/\sqrt{m})$ by using Hoeffding's inequality. For the $(s,a)\notin\mathcal{C}$, we

can then use the resulting $\hat{\theta}_r^\pi$ to estimate $\hat{Q}_r^\pi$ as $\hat{Q}_r^\pi = \langle \hat{\theta}^\pi, \phi(s,a) \rangle$. In Appendix B, we prove the following result to bound the extrapolation errors for all $(s,a)$.

**Lemma 5.2.** *For policy $\pi$, any distribution $\omega$ and subset $\mathcal{C}$, if we use $m$ trajectories to estimate the action-value function for each $(s,a) \in \mathcal{C}$, and solve Eq. (9) to compute $\hat{\theta}_r^\pi$, then for any $(s,a) \in (\mathcal{S} \times \mathcal{A})$ pair, the error $|\langle \phi(s,a), \hat{\theta}_r^\pi \rangle - Q_r^\pi|$ can be upper-bounded by*

$$\varepsilon_b (1 + \|\phi(s,a)\|_{G_\omega^{-1}}) + \frac{\|\phi(s,a)\|_{G_\omega^{-1}}}{1-\gamma} \sqrt{\frac{\log(2|\mathcal{C}|/\delta)}{2m}}.$$

*where, $G_\omega = \sum_{(s,a)\in\mathcal{C}} \omega(s,a) \phi(s,a) \phi(s,a)^\intercal$.*

Hence, the extrapolation errors can be upper-bounded by choosing $\mathcal{C}$ and $\omega$ to control the $\|\phi(s,a)\|_{G_\omega^{-1}}$ term for each $(s,a)$ pair. Moreover, to ensure scalability, we want that size of $\mathcal{C}$ to be independent of $|\mathcal{S}||\mathcal{A}|$. Fortunately, the Kiefer-Wolfowitz theorem (Kiefer and Wolfowitz, 1960) guarantees the existence of a *coreset* $\mathcal{C}$ s.t. $|\mathcal{C}| \leq \frac{d(d+1)}{2}$ and distribution $\omega$ that ensure $\sup_{(s,a)} \|\phi(s,a)\|_{G_\omega^{-1}} \leq \sqrt{d}$. If we can find such a $\mathcal{C}$ and distribution $\omega$, then function approximation error, $\tilde{\varepsilon} \leq \varepsilon_b(1+\sqrt{d}) + \frac{\sqrt{d}}{1-\gamma}\sqrt{\frac{\log(2d(d+1)/\delta)}{2m}}$. For our theoretical results, we will assume that such a coreset $\mathcal{C}$ and distribution $\omega$ is provided, and in Appendix C, we describe the G-experimental design procedure to compute it.

Now that we have control over $\tilde{\varepsilon}$, we instantiate the primal-dual framework with coin-betting algorithms.

## 5.1 CBP ALGORITHM

In this section, we use the coin-betting algorithms (Section 4.2.2) with linear function approximation to completely specify the Coin-Betting Politex (CBP) algorithm (Algorithm 1). In Algorithm 1, Line 2 computes the coreset $C$ and distribution $\omega$ offline (see Appendix C.2 for details). In order to set $U$, the upper-bound on the dual variables, we need to estimate $\zeta$ and this is achieved by solving the unconstrained problem maximizing $\hat{V}_c^\pi(\rho)$ in Line 3. While this can be done by any algorithm that can solve MDPs with linear function approximation (for example, NPG (Kakade, 2001) or Politex (Abbasi-Yadkori et al., 2019)), we will use Eq. (7) (see Section 6) in this work. After Monte-Carlo sampling $\forall (s,a) \in \mathcal{C}$ (Line 5) and estimating $\hat{\theta}_r^{\pi_t}$ and $\hat{\theta}_c^{\pi_t}$ according to Eq. (9) (Line 6), these vectors are used to calculate $\hat{Q}_r^{\pi_t}$ and $\hat{Q}_c^{\pi_t}$ for states encountered in a trajectory generated by policy $\pi_t$ (Line 8). These action-value functions are then used to update the policy at these states. While this can be achieved by any algorithm controlling the primal regret, CBP uses the parameter-free coin-betting updates (Line 9). At the end of iteration $t$, in Line 11, the dual variables are updated using the coin-betting algorithm.

**Algorithm 1:** Coin-Betting Politex

1 **Input**: $\pi_0$ (policy initialization), $\lambda_0$ (dual variable initialization), $m$ (Number of trajectories), $T$ (Number of iterations), Feature map $\Phi$.
2 Compute coreset $\mathcal{C}$ and distribution $\omega$
3 Solve the unconstrained problem $\max_\pi \hat{V}_c^\pi(\rho)$ to estimate $\zeta$ in Lemma 4.1 and set $U = \frac{2}{\zeta(1-\gamma)}$.
4 **for** $t \leftarrow 0$ **to** $T-1$ **do**
5 For every $(s,a) \in \mathcal{C}$, use $m$ trajectories starting from $(s,a)$ using policy $\pi_t$ and estimate the action-value functions $q_r(s,a)$ and $q_c(s,a)$.
6 Compute and store $\hat{\theta}_r^{\pi_t}$ and $\hat{\theta}_c^{\pi_t}$ using Eq. (9).
7 **for** *every $s$ encountered in the trajectory generated by $\pi_t$, and for every $a$* **do**
8 Compute $\hat{Q}_r^t(s,a) = \langle \hat{\theta}_r^{\pi_t}, \phi(s,a) \rangle$; $\hat{Q}_c^t(s,a) = \langle \hat{\theta}_c^{\pi_t}, \phi(s,a) \rangle$ and $\hat{Q}_l^t(s,a) = \hat{Q}_r^t(s,a) + \lambda_t \hat{Q}_c^t(s,a)$.
9 Update $\pi_{t+1}(a|s)$ using Eq. (7).
10 **end**
11 Compute $\hat{V}_c^{\pi_t}(\rho)$, update $\lambda_{t+1}$ using Eq. (8).
12 **end**

In the next section, we bound the average optimality gap and constraint violation for CBP.

### 5.1.1 Theoretical Guarantee

We now use Theorem 3.1 to bound the average optimality gap and constraint violation for Algorithm 1. We note that recent work (Liu and Orabona, 2021) uses parameter-free coin-betting algorithms for convex-concave min-max optimization. Since the function to be maximized in Eq. (3) is non-concave in $\pi$, this work is not directly applicable to our setting.

**Corollary 5.3.** *Under Assumption 5.1, OG and CV of CBP can be bounded as:*

$$\texttt{OG} \leq \frac{\left(\frac{3(1+U)\sqrt{1+KL(\pi_0||\pi^*)}}{1-\gamma} + \Psi\right)}{(1-\gamma)\sqrt{T}} + \frac{\tilde{\varepsilon}(1+2U)}{1-\gamma},$$

$$\texttt{CV} \leq \frac{\zeta\left(\frac{3(1+U)\sqrt{1+KL(\pi_0||\pi^*)}}{1-\gamma} + \Psi\right)}{\sqrt{T}} + \zeta\,\tilde{\varepsilon}(1+2U),$$

*where $U = \frac{2}{\zeta(1-\gamma)}$, $\tilde{\varepsilon} = \varepsilon_b(1+\sqrt{d}) + \frac{\sqrt{d}}{1-\gamma}\sqrt{\frac{\log(2d(d+1)/\delta)}{2m}}$ and $\Psi = 4U\sqrt{\log((T+1)U)} + 1$.*

Since $U = O(1/1-\gamma)$, the average optimality gap for CBP is $O\left(\frac{1}{(1-\gamma)^3\sqrt{T}} + \frac{\tilde{\varepsilon}}{(1-\gamma)^2}\right)$, while the aver-

age constraint violation scales as $O\left(\frac{1}{(1-\gamma)^2\sqrt{T}} + \frac{\tilde{\varepsilon}}{1-\gamma}\right)$. In the function approximation case, ignoring sampling errors, Ding et al. (2020) obtain an $O\left(\frac{1}{(1-\gamma)^3\sqrt{T}} + \left[\frac{\varepsilon_{\mathrm{b}}}{(1-\gamma)^3}\left\|\frac{d^{\pi^*}}{\rho}\right\|_\infty\right]^{1/2}\right)$ average optimality gap, and an $O\left(\frac{1}{(1-\gamma)^2 T^{1/4}} + \left[\frac{\varepsilon_{\mathrm{b}}}{(1-\gamma)^3}\left\|\frac{d^{\pi^*}}{\rho}\right\|_\infty\right]^{1/4}\right)$ average constraint violation. Here, $d^{\pi^*}$ is the distribution over states induced by the optimal policy, and $\rho$ is the initial state distribution. Compared to Corollary 5.3, the CV decreases at a slower $O\left(1/T^{1/4}\right)$ rate. Comparing the error terms, the bound for Ding et al. (2020) depends on the potentially large (even infinite) $\left\|\frac{d^{\pi^*}}{\rho}\right\|_\infty$ factor, while forming the coreset ensures that the errors are well controlled for Corollary 5.3. Furthermore, Ding et al. (2020) require knowledge of the typically unknown Slater constant for the CMDP.

On the other hand, Xu et al. (2021) use a neural function approximation (with 1 hidden layer) where only the first layer is trained. In order to compare to Corollary 5.3, we set the width of the second layer to 1 in Theorem 2 of Xu et al. (2021), making the function approximation equal to a linear mapping with a ReLU non-linearity. In this setting, Xu et al. (2021, Theorem 4) prove that both the average optimality gap and constraint violation scale as $O\left(\frac{1}{(1-\gamma)\sqrt{T}} + \frac{\varepsilon_{\mathrm{b}}}{(1-\gamma)^{2.5}}\left\|\frac{d^{\pi^*}}{\rho}\right\|_\infty\right)$. Observe that although both OG and CV decrease at an $O\left(1/\sqrt{T}\right)$ rate, the error amplification also depends on $\left\|\frac{d^{\pi^*}}{\rho}\right\|_\infty$. Furthermore, this result requires setting the hyperparameters according to the typically unknown $\mathrm{KL}(\pi^*||\pi_0)$ quantity. These problems make the theoretical results of Ding et al. (2020) and Xu et al. (2021) potentially vacuous, and the algorithms difficult to use.

# 6 EXPERIMENTS

In this section, we first describe some practical considerations for implementing CBP and compare with baselines GDA and CRPO on a synthetic tabular environment and the Cartpole environment with linear function approximation.

## 6.1 PRACTICAL CONSIDERATIONS

**Checking feasibility and Estimating $\zeta$:** We use the updates in Eq. (7) to solve the unconstrained problem maximizing $\hat{V}_c^{\pi}(\rho)$, and return policy $\tilde{\pi}$. If $\hat{V}_c^{\tilde{\pi}}(\rho) < b$, we declare the problem infeasible, whereas, if $\hat{V}_c^{\tilde{\pi}}(\rho) > b$, we estimate $\zeta = \hat{V}_c^{\tilde{\pi}}(\rho) - b$.[1] It is important to note that Lemma 4.1 does not require the exact maximization of $\hat{V}_c^{\pi}(\rho)$ to upper-bound $\lambda^*$. Any feasible policy for which $V_c^{\pi}(\rho) > b$ can be used to estimate $\zeta$ and upper-bound $\lambda^*$, though the tightest upper-bound is obtained for $\max_\pi V_c^{\pi}(\rho)$ (see the proof of Lemma 4.1 in Appendix B).

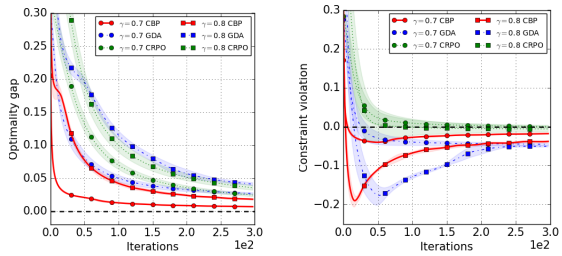
**Gradient normalization and practical coin-betting:** Recall that the coin-betting algorithms in Section 4.2.2 require normalizing the gradients by $1+U/1-\gamma$. Unfortunately, this upper-bound on the gradient norms is quite loose in practice, and directly using the updates Eqs. (7) and (8) results in poor empirical performance. Since coin-betting algorithms do not have a step-size that can be scaled to counteract the normalization, this issue needs to be handled differently. In particular, we continue to directly use the updates in Eq. (7) with the normalization, but use a heuristic, Algorithm 2 of Orabona and Tommasi 2017, for updating the dual variable. This heuristic is a way to adaptively normalize the dual gradients (depending on the previously observed values). For the details, see Algorithm 2 in Appendix C.1. While this heuristic introduces a hyperparameter in the dual updates, our empirical results suggest that the resulting coin-betting algorithm is quite robust to the choice of this parameter and so we use this method in our subsequent experiments.

## 6.2 TABULAR SETTING

We consider a synthetic gridworld environment similar to Sutton and Barto (2018, Example 3.5) (see Appendix C.3 for details) and set the discount factor $\gamma = 0.9$. We first consider a **model-based setting**, where we have complete knowledge of the CMDP. In Fig. 1 (in Section 1), we compared the performance of the three algorithms. For each algorithm, the hyperparameter range is described in Appendix D and the *best hyperparameter* corresponds to the least OG while satisfying CV $\in [-0.25, 0]$. The key observation is that *CBP is robust to its hyperparameter values*, while GDA and CRPO are sensitive to their hyperparameter values. In Fig. 6 (Appendix D.1), we show best performing variants for all methods. In addition, we demonstrate the poor performance of GDA when used with the theoretical step-sizes suggested in Corollary A.1. Next, we measure the robustness of the algorithms with respect to *environment misspecification* where we vary $\gamma$. In Fig. 2, we observe that CBP has consistently faster convergence with a lower variation in the performance.

In Appendix D.1, we also consider the **model-free setting** and approximate the $Q$ functions with sampling. In this case, we demonstrate the consistent superiority of CBP (Fig. 7) and its robustness to hyperparameters (Fig. 8) and environment misspecification (Fig. 9).

[1] If $\hat{V}_c^{\tilde{\pi}}(\rho) = b$, then we return policy $\tilde{\pi}$ as the optimal feasible policy in the CMDP.

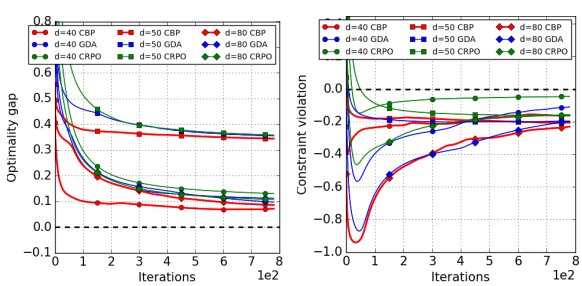



(a) OG (b) CV

Figure 2: **Environment Misspecification – Model-based tabular setting with varying $\gamma$:** Assuming access to the true CMDP, we vary discount factor $\gamma = \{0.7, 0.8\}$. We use the hyperparameters for the original CMDP with $\gamma = 0.9$. CBP converges faster with a smaller variance as compared to GDA and CRPO.

### 6.3 LINEAR SETTING

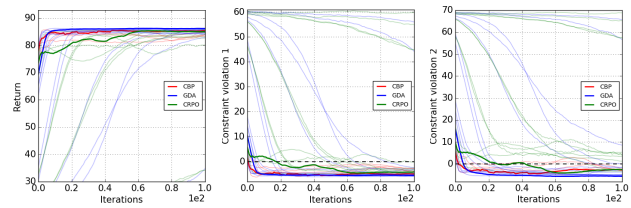
**Gridworld environment:** We start with linear function approximation (LFA) on the gridworld environment. We use tile coding (Sutton and Barto, 2018) to construct $d$-dimensional feature space (see Appendix D.2 for details). We used LSTDQ (Lagoudakis and Parr, 2003) to estimate $Q$ functions with 300 samples for all $(s, a)$ pairs. In Fig. 3, we show the performance of the best hyperparameter (see Table 2 for specific values) for each algorithm. We observe that the `OG` of CBP converges consistently faster across different feature dimensions. Again, we observe a good hyperparameter robustness of CBP in Fig. 10 (Appendix D). Fig. 11 in Appendix D.2 shows that we can obtain similar performance by using G-experimental design, but at a much lower computational cost.




(a) OG (b) CV

Figure 3: **LFA in gridworld environment:** For varying feature dimension $d$, `OG` for CBP consistently converges faster than the baselines GDA and CRPO.

**Cartpole environment with exploration:** We use the Cartpole environment from the OpenAI gym (Brockman et al., 2016), and modify it to include multiple constraints. The agent is rewarded to keep the pole upright, whereas it receives a constraint reward if (1) the cart enters certain areas (x-axis position), or




(a) Return (b) CV 1 (c) CV 2

Figure 4: **Cartpole environment:** Performance of CBP, GDA and CRPO with two constraints. The dark lines depict the performance with the best hyperparameters. The light lines correspond to performance with other setting of hyperparameters. CBP exhibit robustness to the choice of hyperparameters.

(2) the angle of pole is smaller than a certain threshold (see Appendix D.2 for details). We used tile coding to construct the feature space, and LSTDQ to estimate the $Q$ functions for both reward and constraint reward. Similar to Xu et al. (2021), we observed better performance by using entropy regularization (Geist et al., 2019; Haarnoja et al., 2018), which we used with all methods. We used the same regularization strength ($\nu = 0.001$) with all the methods, which made all of them reasonably stable.

In Fig. 4 we show the cumulative discounted reward and the constraint violation (`CV 1`, `CV 2`) for the two constraints as mentioned above. The dark lines correspond to the best hyperparameter that achieves the maximum return, while satisfying `CV` $\in [-6, 0]$ for both constraints, with the lighter shade-lines correspond to the other hyperparameters. All the three algorithms satisfy the constraints, achieve comparable reward, but CBP has considerably less variance in performance for different values of the hyperparameters. Finally, in Fig. 12 (Appendix D.2), we demonstrate that this trend holds across a range of regularization coefficients.

## 7 CONCLUSION

In this paper, we proposed a general primal-dual framework to solve CMDPs with function approximation. We instantiated this framework using coin-betting algorithms from online linear optimization, and proposed the CBP algorithm. We proved that CBP is not only theoretically sound and has good empirical performance, but also robust to hyper-parameter tuning and environment misspecification. In the future, we hope to use the recent advances in online linear optimization to design "painless" parameter-free policy optimization algorithms. We believe that this ambition is important for reproducibility in RL, and hope that our work will encourage future research in this area.

## References


Yasin Abbasi-Yadkori, Peter Bartlett, Kush Bhatia, Nevena Lazic, Csaba Szepesvari, and Gellért Weisz. Politex: Regret bounds for policy iteration using expert prediction. In *International Conference on Machine Learning*, 2019.

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International conference on machine learning*, pages 22–31. PMLR, 2017.

Eitan Altman. *Constrained Markov decision processes*, volume 7. CRC Press, 1999.

Shalabh Bhatnagar and K Lakshmanan. An online actor–critic algorithm with function approximation for constrained markov decision processes. *Journal of Optimization Theory and Applications*, 153(3):688–708, 2012.

Vivek Borkar and Rahul Jain. Risk-constrained Markov decision processes. *IEEE Transactions on Automatic Control*, 59(9):2574–2579, 2014.

Vivek S Borkar. An actor-critic algorithm for constrained Markov decision processes. *Systems & control letters*, 54(3):207–213, 2005.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Chiara Buratti, Andrea Conti, Davide Dardari, and Roberto Verdone. An overview on wireless sensor networks technology and evolution. *Sensors*, 9(9): 6869–6896, 2009.

Shicong Cen, Chen Cheng, Yuxin Chen, Yuting Wei, and Yuejie Chi. Fast global convergence of natural policy gradient methods with entropy regularization. *Operations Research*, 2021.

Yinlam Chow, Ofir Nachum, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. A lyapunov-based approach to safe reinforcement learning. *Advances in neural information processing systems*, 31, 2018.

Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. Safe exploration in continuous action spaces. *arXiv preprint arXiv:1801.08757*, 2018.

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, 2020.

Dongsheng Ding, Xiaohan Wei, Zhuoran Yang, Zhaoran Wang, and Mihailo Jovanovic. Provably efficient safe exploration via primal-dual policy optimization. In *International Conference on Artificial Intelligence and Statistics*, pages 3304–3312. PMLR, 2021.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

Jaime F Fisac, Anayo K Akametalu, Melanie N Zeilinger, Shahab Kaynama, Jeremy Gillula, and Claire J Tomlin. A general safety framework for learning-based control in uncertain robotic systems. *IEEE Transactions on Automatic Control*, 64(7): 2737–2752, 2018.

Matthieu Geist, Bruno Scherrer, and Olivier Pietquin. A theory of regularized markov decision processes. In *International Conference on Machine Learning*, pages 2160–2169. PMLR, 2019.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

David Julian, Mung Chiang, Daniel O'Neill, and Stephen Boyd. Qos and fairness constrained convex optimization of resource allocation for wireless cellular and ad hoc networks. In *Proceedings. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 2, pages 477–486. IEEE, 2002.

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

Jack Kiefer and Jacob Wolfowitz. The equivalence of two extremum problems. *Canadian Journal of Mathematics*, 12:363–366, 1960.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

Gen Li, Yuxin Chen, Yuejie Chi, Yuantao Gu, and Yuting Wei. Sample-efficient reinforcement learning is feasible for linearly realizable MDPs with limited revisiting. *Advances in Neural Information Processing Systems*, 34, 2021.

Qingkai Liang, Fanyu Que, and Eytan Modiano. Accelerated primal-dual policy optimization for safe reinforcement learning. *arXiv preprint arXiv:1802.06480*, 2018.

Mingrui Liu and Francesco Orabona. A parameter-free algorithm for convex-concave min-max problems. *arXiv preprint arXiv:2103.00284*, 2021.

Yongshuai Liu, Jiaxin Ding, and Xin Liu. Ipo: Interior-point policy optimization under constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 4940–4947, 2020.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Teodor Mihai Moldovan and Pieter Abbeel. Safe exploration in markov decision processes. *arXiv preprint arXiv:1205.4810*, 2012.

Masahiro Ono, Marco Pavone, Yoshiaki Kuwata, and J Balaram. Chance-constrained dynamic programming with application to risk-aware robotic space exploration. *Autonomous Robots*, 39(4):555–571, 2015.

Francesco Orabona. A modern introduction to online learning. *arXiv preprint arXiv:1912.13213*, 2019.

Francesco Orabona and David Pal. Coin betting and parameter-free online learning. *Advances in Neural Information Processing Systems*, 29:577–585, 2016.

Francesco Orabona and Tatiana Tommasi. Training deep networks without learning rates through coin betting. *Advances in Neural Information Processing Systems*, 30, 2017.

Santiago Paternain, Luiz FO Chamon, Miguel Calvo-Fullana, and Alejandro Ribeiro. Constrained reinforcement learning has zero duality gap. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 7555–7565, 2019.

Andrew J Schaefer, Matthew D Bailey, Steven M Shechter, and Mark S Roberts. Modeling medical treatment using Markov decision processes. In *Operations research and health care*, pages 593–612. Springer, 2005.

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.

Adam Stooke, Joshua Achiam, and Pieter Abbeel. Responsive safety in reinforcement learning by pid lagrangian methods. In *International Conference on Machine Learning*, pages 9133–9143. PMLR, 2020.

Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988.

Richard S Sutton and Andrew G Barto. Reinforcement learning: An introduction. second. *A Bradford Book*, 2018.

Jie Tan, Tingnan Zhang, Erwin Coumans, Atil Iscen, Yunfei Bai, Danijar Hafner, Steven Bohez, and Vincent Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. *arXiv preprint arXiv:1804.10332*, 2018.

Chen Tessler, Daniel J Mankowitz, and Shie Mannor. Reward constrained policy optimization. *arXiv preprint arXiv:1805.11074*, 2018.

Tengyu Xu, Yingbin Liang, and Guanghui Lan. CRPO: A new approach for safe reinforcement learning with convergence guarantee. In *International Conference on Machine Learning*, pages 11480–11491. PMLR, 2021.

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

Tom Zahavy, Brendan O'Donoghue, Guillaume Desjardins, and Satinder Singh. Reward is enough for convex MDPs. *arXiv preprint arXiv:2106.00661*, 2021.

Andy Zeng, Shuran Song, Johnny Lee, Alberto Rodriguez, and Thomas Funkhouser. Tossingbot: Learning to throw arbitrary objects with residual physics. *IEEE Transactions on Robotics*, 36(4):1307–1319, 2020.

# Supplementary material

## ORGANIZATION OF THE APPENDIX


A Theoretical Guarantees in the Tabular Setting

B Main Proofs

C Additional Implementation Details

D Additional Experimental Results


## A THEORETICAL GUARANTEES IN THE TABULAR SETTING

In the tabular setting, we use $m$ independent trajectories for *each* $(s, a)$ pair. By Hoeffding's inequality and union bound across all states and actions, the sampling error can be bounded by $\frac{1}{1-\gamma}\sqrt{\frac{\log(2SA/\delta)}{2m}}$ (similar to the proof of Lemma 5.2). Since all action-value functions can be represented in the tabular setting, the bias error term $\varepsilon_{\mathrm{b}} = 0$, and hence $\tilde{\varepsilon} = \frac{1}{1-\gamma}\sqrt{\frac{\log(2SA/\delta)}{2m}}$. Compared to the linear function approximation setting in Section 5 that has a computational complexity proportional to $O(d^2)$, the computational cost in the tabular setting is $O(SA)$. However, the approximation error is smaller than that in Lemma 5.2.

Now that we have bounded the approximation errors in the tabular setting, we instantiate Theorem 3.1 for GDA in Section 4.2.1. Plugging in the value of $U$, the primal and dual regret from Eq. (4) and $\tilde{\varepsilon}$, we obtain the following corollary.

**Corollary A.1.** *For the gradient descent ascent updates in Eqs. (5) and (6) with the specified step-sizes, $U = \frac{2}{\zeta(1-\gamma)}$, using $m$ trajectories, the average optimality gap (OG) and constraint violation (CV) can be bounded as:*

$$\texttt{OG} \leq \frac{\left(\frac{(1+U)\sqrt{2\log|A|}}{1-\gamma} + U\right)}{(1-\gamma)\sqrt{T}} + \frac{\varepsilon_s(1+2U)}{1-\gamma},$$

$$\texttt{CV} \leq \frac{\zeta\left(\frac{(1+U)\sqrt{2\log|A|}}{1-\gamma} + U\right)}{\sqrt{T}} + \varepsilon_s(1+2U),$$

*where* $\varepsilon_s = \frac{1}{1-\gamma}\sqrt{\frac{\log(2SA/\delta)}{2m}}$.

*Proof.* To get the result we replace the regrets for primal and dual of GDA (Orabona, 2019, Theorem 6.8) in Theorem 3.1 and get the required results. Specifically we set

$$\mathcal{R}^p(\pi^*, T) \leq \frac{1+U}{1-\gamma}\sqrt{2\log|A|}\sqrt{T},$$

and

$$\mathcal{R}^d(0, T), \mathcal{R}^d(U, T) \leq \frac{U}{1-\gamma}\sqrt{T}.$$

□

Hence, the average optimality gap for GDA is $O\left(\frac{1}{(1-\gamma)^3\sqrt{T}} + \frac{\varepsilon_{\mathrm{s}}}{(1-\gamma)^2}\right)$, while the average constraint violation scales as $O\left(\frac{1}{(1-\gamma)^2\sqrt{T}} + \frac{\varepsilon_{\mathrm{s}}}{1-\gamma}\right)$. Compared to the tabular result in Ding et al. (2020), the above bound on the

optimality gap is worse by a factor of $O(1/1-\gamma)$ and matches their bound on the constraint violation. On the other hand, in the tabular setting without sampling error (when $\varepsilon_s = 0$), Xu et al. (2021, Theorem 3) obtain an $O\left(\frac{1}{(1-\gamma)^{1.5}\sqrt{T}}\right)$ bound on both the optimality gap and constraint violation. However, in order to set this bound, they require the knowledge of $\mathrm{KL}(\pi^*||\pi_0)$ to set the algorithm hyper-parameters. This information is not available, making it difficult to implement their algorithm.

Now, we instantiate Theorem 3.1 for the coin-betting algorithms in Section 4.2.2. Plugging in the value of $U$, the primal and dual regret and $\tilde{\varepsilon}$, we obtain the following corollary.

**Corollary A.2.** *Using the primal updates in Eq. (7), and the dual updates in Eq. (8), with $U = \frac{2}{\zeta\,(1-\gamma)}$, using $m$ trajectories, the average optimality gap (OG) and constraint violation (CV) can be bounded as:*

$$\texttt{OG} \leq \frac{\left(\frac{3(1+U)\sqrt{1+KL(\pi_0||\pi^*)}}{1-\gamma} + \Psi\right)}{(1-\gamma)\sqrt{T}} + \frac{\varepsilon_s(1+2U)}{1-\gamma},$$

$$\texttt{CV} \leq \frac{\zeta\left(\frac{3(1+U)\sqrt{1+KL(\pi_0||\pi^*)}}{1-\gamma} + \Psi\right)}{\sqrt{T}} + \zeta\,\varepsilon_s(1+2U),$$

*where $\varepsilon_s = \frac{1}{1-\gamma}\sqrt{\frac{\log(2SA/\delta)}{2m}}$ and $\Psi = 4U\sqrt{\log((T+1)U)} + 1$.*

*Proof.* To get the result we replace the regrets for primal and dual of CB in Theorem 3.1 and get the required results. Specifically from (Orabona and Pal, 2016, Corollary 6) and (Orabona and Tommasi, 2017, Theorem 8), we get the upper-bound for primal regret and the dual regret:

$$\mathcal{R}^p(\pi^*, T) \leq \frac{3(1+U)}{1-\gamma}\sqrt{T}\sqrt{1+\mathrm{KL}(\pi_0||\pi^*)},$$

and

$$\mathcal{R}^d(\lambda, T) \leq \frac{1}{1-\gamma} + \left\|\lambda - \lambda^0\right\| \sqrt{\left(\frac{1}{(1-\gamma)^2} + \frac{G_T}{1-\gamma}\right)\Gamma_T}$$

where $\Gamma_T = \log\left(1 + (G_T\,(1-\gamma)+1)^2\left\|\lambda-\lambda^0\right\|^2\right)$ and $G_T = \sum_{i=0}^{T}|\hat{V}_c^{\pi_i}(\rho) - b|$. Since $|\hat{V}_c^{\pi_i}(\rho) - b| \leq \frac{1}{1-\gamma}$ we have $G_T \leq T/1-\gamma$ and $\left\|\lambda-\lambda^0\right\| \leq 2U$ for all $\lambda$. Using these upperbound and replace in $\mathcal{R}^d(\lambda, T)$ we get:

$$\mathcal{R}^d(\lambda, T) \leq \frac{4U\sqrt{(T+1)\log((T+1)U)} + 1}{1-\gamma}$$

□

## B MAIN PROOFS

The following well known result will be useful:

**Lemma B.1** (Value difference lemma)**.** *For any value function $V^\pi$ (reward or cost), and any two memoryless policies $\pi$ and $\pi'$,*

$$V^{\pi'} - V^\pi = (I - \gamma P_{\pi'})^{-1} \left[T_{\pi'} V^\pi - V^\pi\right]$$

*where $T_{\pi'} V^\pi = [r_{\pi'} + \gamma P_{\pi'} V^\pi]$ is the Bellman operator for policy $\pi'$.*

*Proof.* As is well known, $V^{\pi'} = (I - \gamma P_{\pi'})^{-1} r_{\pi'}$. Hence,

$$\begin{aligned} V^{\pi'} - V^\pi &= (I - \gamma P_{\pi'})^{-1} r_{\pi'} - V^\pi \\ &= (I - \gamma P_{\pi'})^{-1} \left(r_{\pi'} - (I - \gamma P_{\pi'}) V^\pi\right) \\ &= (I - \gamma P_{\pi'})^{-1} \left(r_{\pi'} + \gamma P_{\pi'} V^\pi - V^\pi\right) \\ &= (I - \gamma P_{\pi'})^{-1} \left[T_{\pi'} V^\pi - V^\pi\right] \end{aligned}$$

□

Let us now turn to the proof of Theorem 3.1:

**Theorem 3.1.** *Assuming that $\left\|Q_r^t - \hat{Q}_r^t\right\|_\infty \leq \tilde{\varepsilon}$ and $\left\|Q_c^t - \hat{Q}_c^t\right\|_\infty \leq \tilde{\varepsilon}$, for a generic algorithm producing a sequence of polices $\{\pi_0, \pi_1, \ldots, \pi_{T-1}\}$ and dual variables $\{\lambda_0, \lambda_1, \ldots, \lambda_{T-1}\}$ such that for all $t$, $\lambda_t$ is constrained to lie in the $[0, U]$ where $U > \lambda^*$, `OG` and `CV` can be bounded as:*

$$\begin{aligned} \texttt{OG} &\leq \frac{\mathcal{R}^p(\pi^*, T) + (1-\gamma)\mathcal{R}^d(0, T)}{(1-\gamma)T} + \tilde{\varepsilon}\, g(U), \\ \texttt{CV} &\leq \frac{\mathcal{R}^p(\pi^*, T) + (1-\gamma)\mathcal{R}^d(U, T)}{(U-\lambda^*)(1-\gamma)T} + \frac{\tilde{\varepsilon}\, g(U)}{(U-\lambda^*)}, \end{aligned}$$

*where $g(U) := \left[\frac{1+U}{1-\gamma} + U\right]$.*

*Proof.* We will begin with bounding the value differences in the Lagrangian using Lemma B.1. Let $T^r_{\pi^*}$ and $T^c_{\pi^*}$ be the Bellman operators of the optimal policy for the reward and cost respectively. Then,

$$[V_r^{\pi^*} - V_r^{\pi_t}] + \lambda_t \left[V_c^{\pi^*} - V_c^{\pi_t}\right] = (I - \gamma P_{\pi^*})^{-1} \left[ [T^r_{\pi^*} V_r^{\pi_t} - V_r^{\pi_t}] + \lambda_t \left[T^c_{\pi^*} V_c^{\pi_t} - V_c^{\pi_t}\right] \right]$$

Let $M_\pi$ be the state-action operator applied $Q$ functions such that $M_\pi(Q)(s) = \sum_a \pi(a|s) Q(s,a)$. Observe that $T^r_{\pi^*} V_r^{\pi_t} = M_{\pi^*} Q_r^{\pi_t}$ and $V_r^{\pi_t} = M_{\pi_t} Q_r^{\pi_t}$. The expressions for the constraint rewards are analogous. Rewriting the

above expression,

$$
\begin{aligned}
[V_r^{\pi^*} - V_r^{\pi_t}] + \lambda_t [V_c^{\pi^*} - V_c^{\pi_t}] &= (I - \gamma P_{\pi^*})^{-1} \left[ [M_{\pi^*} Q_r^{\pi_t} - M_{\pi_t} Q_r^{\pi_t}] + \lambda_t [M_{\pi^*} Q_c^{\pi_t} - M_{\pi_t} Q_c^{\pi_t}] \right] \\
&= (I - \gamma P_{\pi^*})^{-1} \left[ [M_{\pi^*} - M_{\pi_t}] [Q_r^{\pi_t} + \lambda_t Q_c^{\pi_t}] \right] \\
&= (I - \gamma P_{\pi^*})^{-1} \left[ [M_{\pi^*} - M_{\pi_t}] [\hat{Q}_r^{\pi_t} + \lambda_t \hat{Q}_c^{\pi_t}] \right] \\
&+ \underbrace{(I - \gamma P_{\pi^*})^{-1} \left[ [M_{\pi^*} - M_{\pi_t}] [Q_r^{\pi_t} - \hat{Q}_r^{\pi_t} + \lambda_t (Q_c^{\pi_t} - \hat{Q}_c^{\pi_t})] \right]}_{\text{Error}}
\end{aligned}
$$

Let us first bound the maximum norm of the "Error" term,

$$
\begin{aligned}
\|\text{Error}\|_\infty &= \left\| (I - \gamma P_{\pi^*})^{-1} \left[ [M_{\pi^*} - M_{\pi_t}] [Q_r^{\pi_t} - \hat{Q}_r^{\pi_t} + \lambda_t (Q_c^{\pi_t} - \hat{Q}_c^{\pi_t})] \right] \right\|_\infty \\
&\leq \frac{1}{1-\gamma} \left\| [Q_r^{\pi_t} - \hat{Q}_r^{\pi_t} + \lambda_t (Q_c^{\pi_t} - \hat{Q}_c^{\pi_t})] \right\|_\infty \\
&\leq \frac{1}{1-\gamma} \left\| Q_r^{\pi_t} - \hat{Q}_r^{\pi_t} \right\|_\infty + \lambda_t \left\| Q_c^{\pi_t} - \hat{Q}_c^{\pi_t}] \right\|_\infty
\end{aligned}
$$

By assumption, $\left\| Q_r^{\pi_t} - \hat{Q}_r^{\pi_t} \right\|_\infty, \left\| Q_r^{\pi_t} - \hat{Q}_r^{\pi_t} \right\|_\infty \leq \varepsilon$.

$$
\implies \|\text{Error}\|_\infty \leq \frac{\varepsilon}{1-\gamma}(1 + \lambda_t)
$$

Since the dual variables are projected onto the $[0, U]$ interval, $\lambda_t \leq U$, implying that

$$
\|\text{Error}\|_\infty \leq \frac{\varepsilon}{1-\gamma} (1 + U)
$$

Substituting in this bound on the error, using the convention that left-multiplication by a measure means integration with respect to it,

$$
\begin{aligned}
[V_r^{\pi^*}(\rho) - V_r^{\pi_t}(\rho)] + \lambda_t [V_c^{\pi^*}(\rho) - V_c^{\pi_t}(\rho)] &\leq \rho (I - \gamma P_{\pi^*})^{-1} \left[ [M_{\pi^*} - M_{\pi_t}] [\hat{Q}_r^{\pi_t} + \lambda_t \hat{Q}_c^{\pi_t}] \right] + \frac{\varepsilon}{1-\gamma} (1 + U) \\
&\leq \frac{1}{1-\gamma} \nu_{\rho,\pi^*} \left[ [M_{\pi^*} - M_{\pi_t}] [\hat{Q}_r^{\pi_t} + \lambda_t \hat{Q}_c^{\pi_t}] \right] + \frac{\varepsilon}{1-\gamma} (1 + U) \ ,
\end{aligned}
$$

where $\nu_{\rho,\pi^*} = (1-\gamma)\rho (I - \gamma P_{\pi^*})^{-1}$ is the discounted probability measure over the states obtained when starting from $\rho$ and following $\pi^*$. Summing from $t = 0$ to $T - 1$ and dividing by $T$.

$$
\frac{1}{T} \nu_{\rho,\pi^*} \sum_{t=0}^{T-1} \left[ [V_r^{\pi^*}(\rho) - V_r^{\pi_t}(\rho)] + \lambda_t [V_c^{\pi^*}(\rho) - V_c^{\pi_t}(\rho)] \right] \leq \frac{\nu_{\rho,\pi^*}}{(1-\gamma)T} \sum_{t=0}^{T-1} \left[ [\mathcal{M}_{\pi^*} - \mathcal{M}_{\pi_t}] [\hat{Q}_r^{\pi_t} + \lambda_t \hat{Q}_c^{\pi_t}] \right] + \frac{\varepsilon}{1-\gamma} (1 + U)
$$

Now, observe that

$$
\nu_{\rho,\pi^*} \sum_{t=0}^{T-1} \left[ [\mathcal{M}_{\pi^*} - \mathcal{M}_{\pi_t}] [\hat{Q}_r^{\pi_t} + \lambda_t \hat{Q}_c^{\pi_t}] \right] = \sum_{t=0}^{T-1} \langle \pi^*(\cdot|s) - \pi_t(\cdot|s), \hat{Q}_r^{\pi_t}(s, \cdot) + \lambda_t \hat{Q}_c^{\pi_t}(s, \cdot) ] \rangle_{s \sim \nu_{\rho,\pi^*}} = \mathcal{R}^p(\pi^*, T)
$$

Putting everything together,

$$
\frac{1}{T} \sum_{t=0}^{T-1} [V_r^{\pi^*}(\rho) - V_r^{\pi_t}(\rho)] + \frac{1}{T} \sum_{t=0}^{T-1} \lambda_t [V_c^{\pi^*}(\rho) - V_c^{\pi_t}(\rho)] \leq \frac{\mathcal{R}^p(\pi^*, T)}{(1-\gamma)T} + \frac{\varepsilon}{1-\gamma} (1 + U) \, . \tag{10}
$$

The above result bounds the sub-optimality in the Lagrangian. Next, we will see how this result implies a bound on the sub-optimality in the objective and the constraint violation. To bound the reward sub-optimality, we will upper bound the negative of the second term on the left-hand side in the above equation, i.e., we upper bound $\frac{1}{T}\sum_{t=0}^{T-1}\lambda_t\left[V_c^{\pi_t}(\rho)-V_c^{\pi^*}(\rho)\right]$. We have,

$$\begin{aligned}
\frac{1}{T}\sum_{t=0}^{T-1}\lambda_t\left[V_c^{\pi_t}(\rho)-V_c^{\pi^*}(\rho)\right] &\leq \frac{1}{T}\sum_{t=0}^{T-1}\lambda_t\left[V_c^{\pi_t}(\rho)-b\right] && \text{(since } V_c^{\pi^*}(\rho)\geq b\text{)}\\
&= \frac{1}{T}\sum_{t=0}^{T-1}\lambda_t\left[V_c^{\pi_t}(\rho)-\hat{V}_c^{\pi_t}(\rho)\right]+\frac{1}{T}\sum_{t=0}^{T-1}\lambda_t[\hat{V}_c^{\pi_t}(\rho)-b]\\
&= \frac{1}{T}\sum_{t=0}^{T-1}\lambda_t\left[V_c^{\pi_t}(\rho)-\hat{V}_c^{\pi_t}(\rho)\right]+\frac{\mathcal{R}^d(0,T)}{T}\\
&\leq U\varepsilon+\frac{\mathcal{R}^d(0,T)}{T}\,. && (11)
\end{aligned}$$

Using Eqs. (10) and (11),

$$\texttt{OG}=\frac{1}{T}\sum_{t=0}^{T-1}[V_r^{\pi^*}(\rho)-V_r^{\pi_t}(\rho)]\leq\frac{\mathcal{R}^p(\pi^*,T)+(1-\gamma)\mathcal{R}^d(0,T)}{(1-\gamma)T}+\frac{\varepsilon}{1-\gamma}(1+U)+U\varepsilon \tag{12}$$

This proves the first part of the theorem. We now bound the constraint violation. For an arbitrary $\lambda$,

$$\begin{aligned}
\frac{1}{T}\sum_{t=0}^{T-1}[(\lambda_t-\lambda)(V_c^{\pi_t}(\rho)-b)] &= \frac{1}{T}\sum_{t=0}^{T-1}\left[(\lambda_t-\lambda)(V_c^{\pi_t}(\rho)-\hat{V}_c^{\pi_t}(\rho))\right]+\frac{1}{T}\sum_{t=0}^{T-1}\left[(\lambda_t-\lambda)(\hat{V}_c^{\pi_t}(\rho)-b)\right]\\
&= \frac{1}{T}\sum_{t=0}^{T-1}\left[(\lambda_t-\lambda)(V_c^{\pi_t}(\rho)-\hat{V}_c^{\pi_t}(\rho))\right]+\frac{\mathcal{R}^d(\lambda,T)}{T}\,,
\end{aligned}$$

implying

$$\frac{1}{T}\sum_{t=0}^{T-1}[(\lambda_t-\lambda)(V_c^{\pi_t}(\rho)-b)]\leq U\varepsilon+\frac{\mathcal{R}^d(\lambda,T)}{T}\,. \tag{13}$$

Adding Eq. (13) and Eq. (10) and reordering the terms gives

$$\begin{aligned}
&\frac{1}{T}\sum_{t=0}^{T-1}(V_r^{\pi^*}(\rho)-V_r^{\pi_t}(\rho))+\frac{\lambda}{T}\sum_{t=0}^{T-1}(b-V_c^{\pi_t}(\rho))\\
&\leq\frac{1}{T}\sum_{t=0}^{T-1}\underbrace{\lambda_t(b-V_c^{\pi^*}(\rho))}_{\leq 0 \text{ since } V_c^{\pi^*}(\rho)\geq b.}+\underbrace{\frac{\mathcal{R}^p(\pi^*,T)+(1-\gamma)\mathcal{R}^d(\lambda,T)}{(1-\gamma)T}+\frac{\varepsilon}{1-\gamma}(1+U)+U\varepsilon}_{h(\lambda)}\\
\implies&\frac{1}{T}\sum_{t=0}^{T-1}(V_r^{\pi^*}(\rho)-V_r^{\pi_t}(\rho))+\frac{\lambda}{T}\sum_{t=0}^{T-1}(b-V_c^{\pi_t}(\rho))\leq h(\lambda)
\end{aligned}$$

We consider two cases: (i) if $\sum_{t=0}^{T-1}(b-V_c^{\pi_t}(\rho))\geq 0$, we set $\lambda=U$, else, if (ii) $\sum_{t=0}^{T-1}(b-V_c^{\pi_t}(\rho))<0$, we set $\lambda=0$. Using these choices, and since $\mathcal{R}^d(\lambda,T)$ is linearly increasing in $\lambda$,

$$\frac{1}{T}\sum_{t=0}^{T-1}(V_r^{\pi^*}(\rho)-V_r^{\pi_t}(\rho))+\frac{U}{T}\left[\sum_{t=0}^{T-1}(b-V_c^{\pi_t}(\rho))\right]_+\leq h(U)$$

Now take the policy $\pi'$ such that $V_r^{\pi^*}(\rho)-V_r^{\pi'}(\rho)=\frac{1}{T}\sum_{t=0}^{T-1}(V_r^{\pi^*}(\rho)-V_r^{\pi_t}(\rho))$ and $V_c^{\pi^*}(\rho)-V_c^{\pi'}(\rho)=\frac{1}{T}\sum_{t=0}^{T-1}(b-V_c^{\pi_t}(\rho))$. Then,

$$[V_r^{\pi^*}(\rho)-V_r^{\pi'}(\rho)]+U\left[b-V_c^{\pi'}(\rho)\right]_+\leq h(U)\,.$$

Using Lemma B.2 with $C = U > \lambda^*$ and $\beta = h(U)$, we get

$$\begin{aligned}
\texttt{CV} &= \frac{1}{T}\left[\sum_{t=0}^{T-1} b - V_c^{\pi_t}(\rho)\right]_+ = \left[b - V_c^{\pi'}(\rho)\right]_+ \\
&\leq \frac{h(U)}{U-\lambda^*} = \frac{\mathcal{R}^p(\pi^*, T) + (1-\gamma)\mathcal{R}^d(U,T)}{(U-\lambda^*)(1-\gamma)T} + \frac{1}{(U-\lambda^*)}\left[\frac{\varepsilon}{(1-\gamma)}(1+U) + U\varepsilon\right],
\end{aligned}$$

which completes the proof subject to proving Lemma B.2. □

**Lemma B.2** (Constraint violation bound)**.** *For any $C > \lambda^*$ and any $\tilde{\pi}$ s.t. $V_r^{\pi^*}(\rho) - V_r^{\tilde{\pi}}(\rho) + C[b - V_c^{\tilde{\pi}}(\rho)]_+ \leq \beta$, we have $[b - V_c^{\tilde{\pi}}(\rho)]_+ \leq \frac{\beta}{C-\lambda^*}$.*

*Proof.* Define $\nu(\tau) = \max_\pi\{V_r^\pi(\rho) \mid V_c^\pi(\rho) \geq b + \tau\}$ and note that by definition, $\nu(0) = V_r^{\pi^*}(\rho)$ and that $\nu$ is a decreasing function for its argument.

Let $V_l^{\pi,\lambda}(\rho) = V_r^\pi(\rho) + \lambda(V_c^\pi(\rho) - b)$. Then, for any policy $\pi$ s.t. $V_c^\pi(\rho) \geq b + \tau$, we have

$$\begin{aligned}
V_l^{\pi,\lambda^*}(\rho) &\leq \max_{\pi'} V_l^{\pi',\lambda^*}(\rho) \\
&= V_r^{\pi^*}(\rho) && \text{(by strong duality)} \\
&= \nu(0) && \text{(from above relation)} \\
\implies \nu(0) - \tau\lambda^* &\geq V_l^{\pi,\lambda^*}(\rho) - \tau\lambda^* = V_r^\pi(\rho) + \lambda^* \underbrace{(V_c^\pi(\rho) - b - \tau)}_{\text{Positive}} \\
\implies \nu(0) - \tau\lambda^* &\geq \max_\pi\{V_r^\pi(\rho) \mid V_c^\pi(\rho) \geq b + \tau\} = \nu(\tau)\,. \\
\implies \tau\lambda^* &\leq \nu(0) - \nu(\tau)\,. && (14)
\end{aligned}$$

Now we choose $\tilde{\tau} = -(b - V_c^{\tilde{\pi}}(\rho))_+$.

$$\begin{aligned}
(C-\lambda^*)|\tilde{\tau}| &= \lambda^*\tilde{\tau} + C|\tilde{\tau}| && \text{(since } \tilde{\tau} \leq 0\text{)} \\
&\leq \nu(0) - \nu(\tilde{\tau}) + C|\tilde{\tau}| && \text{(Eq. (14))} \\
&= V_r^{\pi^*}(\rho) - V_r^{\tilde{\pi}}(\rho) + C|\tilde{\tau}| + V_r^{\tilde{\pi}}(\rho) - \nu(\tilde{\tau}) && \text{(definition of } \nu(0)\text{)} \\
&= V_r^{\pi^*}(\rho) - V_r^{\tilde{\pi}}(\rho) + C(b - V_c^{\tilde{\pi}}(\rho))_+ + V_r^{\tilde{\pi}}(\rho) - \nu(\tilde{\tau}) \\
&\leq \beta + V_r^{\tilde{\pi}}(\rho) - \nu(\tilde{\tau})\,.
\end{aligned}$$

Now let us bound $\nu(\tilde{\tau})$:

$$\begin{aligned}
\nu(\tilde{\tau}) &= \max_\pi\{V_r^\pi(\rho) \mid V_c^\pi(\rho) \geq b - (b - V_c^{\tilde{\pi}}(\rho))_+\} \\
&\geq \max_\pi\{V_r^\pi(\rho) \mid V_c^\pi(\rho) \geq V_c^{\tilde{\pi}}(\rho)\} && \text{(tightening the constraint)} \\
\nu(\tilde{\tau}) \geq V_r^{\tilde{\pi}}(\rho) &\implies (C-\lambda^*)|\tilde{\tau}| \leq \beta \implies (b - V_c^{\tilde{\pi}}(\rho))_+ \leq \frac{\beta}{C-\lambda^*}
\end{aligned}$$

□

### B.1 PROOF OF LEMMA 4.1

**Lemma 4.1.** *The objective Eq. (1) satisfies strong duality, and the optimal dual variables are bounded as $\lambda^* \leq \frac{1}{\zeta(1-\gamma)}$, where $\zeta := \max_\pi V_c^\pi(\rho) - b > 0$.*

*Proof.* Starting from the Lagrangian form in Eq. (3),

$$V_r^*(\rho) := \max_{\pi} \min_{\lambda \geq 0} V_r^{\pi}(\rho) + \lambda[V_c^{\pi}(\rho) - b]$$

Using the linear programming formulation of CMDPs in terms of the state-occupancy measures $\mu$, we know that both the objective and the constraint are linear functions of $\mu$, and strong duality holds w.r.t $\mu$. Since $\mu$ and $\pi$ have a one-one mapping, we can switch the min and the max (Paternain et al., 2019), implying,

$$\begin{aligned} V_r^*(\rho) &= \min_{\lambda \geq 0} \max_{\pi} V_r^{\pi}(\rho) + \lambda[V_c^{\pi}(\rho) - b] \\ &= \max_{\pi} V_r^{\pi}(\rho) + \lambda^*[V_c^{\pi}(\rho) - b]\,. \end{aligned}$$

Define $\tilde{\pi} := \arg\max_{\pi} V_c^{\pi}(\rho)$. Then,

$$\begin{aligned} & V_r^*(\rho) \geq V_r^{\tilde{\pi}}(\rho) + \lambda^*[V_c^{\tilde{\pi}}(\rho) - b] \\ \implies & \lambda^* \leq \frac{V_r^*(\rho) - V_r^{\tilde{\pi}}(\rho)}{[V_c^{\tilde{\pi}}(\rho) - b]} \leq \frac{1}{(1-\gamma)\zeta}\,. \end{aligned}$$

□

### B.2 PROOFS FOR SECTION 5

**Lemma 5.2.** *For policy $\pi$, any distribution $\omega$ and subset $\mathcal{C}$, if we use $m$ trajectories to estimate the action-value function for each $(s,a) \in \mathcal{C}$, and solve Eq. (9) to compute $\hat{\theta}_r^{\pi}$, then for any $(s,a) \in (\mathcal{S} \times \mathcal{A})$ pair, the error $|\langle \phi(s,a), \hat{\theta}_r^{\pi} \rangle - Q_r^{\pi}|$ can be upper-bounded by*

$$\varepsilon_b(1 + \|\phi(s,a)\|_{G_\omega^{-1}}) + \frac{\|\phi(s,a)\|_{G_\omega^{-1}}}{1-\gamma} \sqrt{\frac{\log(2|\mathcal{C}|/\delta)}{2m}}.$$

*where, $G_\omega = \sum_{(s,a)\in\mathcal{C}} \omega(s,a)\phi(s,a)\phi(s,a)^{\intercal}$.*

*Proof.* By solving $\hat{\theta}_r^{\pi} = \arg\min_{\theta} \sum_{z\in\mathcal{C}} \omega(z) \left[\langle \theta, \phi(z) \rangle - q_r(z)\right]^2$ ( Eq. (9)), we get that

$$\theta_r^{\pi} = G_\omega^{-1} \sum_{(s,a)\in\mathcal{C}} \omega(s,a)\phi(s,a)\, q_r^{\pi}(s,a)$$

and lets denote $z = (s,a)$, $\phi(z) = \phi(s,a)$ and $\varepsilon(z) = q_r(z) - Q_r^{\pi}(z) + Q_r^{\pi}(z) - \langle \phi(z), \theta_r^* \rangle$ and $\theta_r^* := \arg\min_{\theta} \max_{(s,a)} \|Q_r^{\pi}(s,a) - \langle \theta, \phi(s,a) \rangle\|$ is the optimal parameter for the given policy $\pi$.

Therefore we can write $q_r(z) = \varepsilon(z) + \langle \phi(z), \theta_r^* \rangle$

$$\begin{aligned} |\langle \phi(z), \theta_r^{\pi} \rangle - Q_r^{\pi}| &= |\langle \phi(z), \theta_r^{\pi} \rangle - \langle \phi(z), \theta_r^* \rangle + \langle \phi(z), \theta_r^* \rangle - Q_r^{\pi}| \\ &\leq |\langle \phi(z), \theta_r^{\pi} \rangle - \langle \phi(z), \theta_r^* \rangle| + \varepsilon_b \quad [\varepsilon_b \text{ from } \textit{Assumption } 5.1] \end{aligned}$$

Now we need to bound the first term of above inequality. Based on the definition of $\varepsilon(z)$, we can write $\theta_r^{\pi} = G_\omega^{-1} \sum_{z'\in\mathcal{C}} (\langle \phi(z'), \theta_r^* \rangle + \varepsilon(z'))\, \omega(z')\phi(z')$. Using this equality we can get easily that:

$$\begin{aligned} |\langle \phi(z), \theta_r^{\pi} \rangle - \langle \phi(z), \theta_r^* \rangle| &= |\sum_{z'\in\mathcal{C}} \varepsilon(z')\omega(z')\phi(z)^T G_\omega^{-1} \phi(z')| \\ &\leq \sum_{z'\in\mathcal{C}} |\varepsilon(z')||\omega(z')\phi(z)^T G_\omega^{-1} \phi(z')| \\ &\leq |\max_{z'\in\mathcal{C}} \varepsilon(z')| \sum_{z'\in\mathcal{C}} |\omega(z')\phi(z)^T G_\omega^{-1} \phi(z')| \end{aligned}$$

To bound the sum term we can

$$\begin{aligned}\left(\sum_{z'\in\mathcal{C}}\omega(z')|\phi(z)^T G_\omega^{-1}\phi(z')|\right)^2 &\le \sum_{z'\in\mathcal{C}}\omega(z')\left(|\phi(z)^T G_\omega^{-1}\phi(z')|\right)^2 && \text{(Jensen's inequality)}\\ &= \phi(z)^T G_\omega^{-1}\left(\sum_{z'\in\mathcal{C}}\left[\omega(z')\phi(z')\phi(z')^T\right]\right)G_\omega^{-1}\phi(z)\\ &= \|\phi(z)\|^2_{G_\omega^{-1}}\end{aligned}$$

To finish the proof we need to bound $|\max_{z'\in\mathcal{C}}\varepsilon(z')|$. Based on the definition of $\varepsilon(z')$ we have

$$\begin{aligned}|\varepsilon(z')| &\le |q_r(z') - Q_r^\pi(z')| + |Q_r^\pi(z') - \langle\phi(z'),\theta_r^*\rangle|\\ &\le |q_r(z) - Q_r^\pi(z)| + \varepsilon_b\\ &\le \frac{1}{1-\gamma}\sqrt{\frac{\log(2/\delta)}{2m}} + \varepsilon_b\end{aligned}$$

where the second inequality is due to function approximation error (Assumption 5.1) and the last inequality comes from Hoeffding's inequality. Specifically, since the $m$ trajectories are independent, and the action-value functions lie in the $[0, 1/1-\gamma]$ range, we use Hoeffding's inequality to conclude that the sampling error for each $z\in\mathcal{C}$ can be upper-bounded by $\frac{1}{1-\gamma}\sqrt{\frac{\log(2/\delta)}{2m}}$. Since we desire uniform control over all states and actions in $\mathcal{C}$, by union bound, with probability $1-\delta$, $|q_r(z) - Q_r^\pi(z)| \le \frac{1}{1-\gamma}\sqrt{\frac{\log(2|\mathcal{C}|/\delta)}{2m}}$. Putting everything together we get the result. □

**Corollary 5.3.** *Under Assumption 5.1,* `OG` *and* `CV` *of CBP can be bounded as:*

$$\texttt{OG} \le \frac{\left(\frac{3(1+U)\sqrt{1+KL(\pi_0||\pi^*)}}{1-\gamma} + \Psi\right)}{(1-\gamma)\sqrt{T}} + \frac{\tilde{\varepsilon}(1+2U)}{1-\gamma},$$

$$\texttt{CV} \le \frac{\zeta\left(\frac{3(1+U)\sqrt{1+KL(\pi_0||\pi^*)}}{1-\gamma} + \Psi\right)}{\sqrt{T}} + \zeta\,\tilde{\varepsilon}(1+2U),$$

*where* $U = \frac{2}{\zeta(1-\gamma)}$, $\tilde{\varepsilon} = \varepsilon_b(1+\sqrt{d}) + \frac{\sqrt{d}}{1-\gamma}\sqrt{\frac{\log(2d(d+1)/\delta)}{2m}}$ *and* $\Psi = 4U\sqrt{\log((T+1)U)} + 1$.

*Proof.* The proof is similar to the proof of Corollary A.2 but with a different $\tilde{\varepsilon}$. □

# C ADDITIONAL IMPLEMENTATION DETAILS

In Appendix C.1, we describe a more practical variant of CBP and in Appendix C.2, we describe the offline G-experimental design procedure required to form the coreset $\mathcal{C}$ for CBP. Details about the synthetic tabular environment are presented in Appendix C.3 whereas Appendix C.4 details the hyperparameters used across the different experiments.

## C.1 PRACTICAL COIN-BETTING POLITEX ALGORITHM

We present the practical version of CBP which uses a parameter $\alpha_\lambda$ in Algorithm 2.

**Algorithm 2:** Practical Coin-Betting Politex

**Input**: $\alpha_\lambda > 0$ (parameter), $\pi_0$ (policy initialization), $\lambda_0$ (dual variable initialization), $m$ (Number of trajectories), $T$ (Number of iterations), Feature map $\Phi$.

**Initialize**: $L_0 = 0$

Compute coreset $\mathcal{C}$ and distribution $\omega$

Solve the unconstrained problem $\max_\pi \hat{V}_c^\pi(\rho)$ to estimate $\zeta$ in Lemma 4.1 and set $U = \frac{2}{\zeta(1-\gamma)}$.

**for** $t \leftarrow 0$ **to** $T-1$ **do**

  For every $(s,a) \in \mathcal{C}$, use $m$ trajectories starting from $(s,a)$ using policy $\pi_t$ and estimate the action-value functions $q_r(s,a)$ and $q_c(s,a)$.

  Compute and store $\hat{\theta}_r^{\pi_t}$ and $\hat{\theta}_c^{\pi_t}$ using Eq. (9).

  **for** *every $s$ encountered in the trajectory generated by $\pi_t$, and for every $a$* **do**

    Compute $\hat{Q}_r^t(s,a) = \langle \hat{\theta}_r^{\pi_t}, \phi(s,a)\rangle$; $\hat{Q}_c^t(s,a) = \langle \hat{\theta}_c^{\pi_t}, \phi(s,a)\rangle$ and $\hat{Q}_l^t(s,a) = \hat{Q}_r^t(s,a) + \lambda_t \hat{Q}_c^t(s,a)$.

    Update policy,

$$\hat{A}_l^t(s,a) = \frac{1-\gamma}{1+U}\left[\hat{Q}_l^t(s,a) - \left\langle \hat{Q}_l^t(s,\cdot), \pi_t(\cdot|s)\right\rangle\right]$$

$$\tilde{A}_l^t(s,a) = \hat{A}_l^t(s,a)\,\mathcal{I}\{w_t(s,a) > 0\} + [\hat{A}_l^t(s,a)]_+\,\mathcal{I}\{w_t(s,a) \le 0\}$$

$$w_{t+1}(s,a) = \frac{\sum_{i=0}^t \tilde{A}_l^i(s,a)}{(t+1)+T/2}\left(1 + \sum_{i=0}^t \tilde{A}_l^i(s,a)\, w_i(s,a)\right)$$

$$\pi_{t+1}(a|s) = \begin{cases} \pi_0(a|s), & \text{if } \sum_a \pi_0(a|s)\,[w_{t+1}(s,a)]_+ = 0 \\ \frac{\pi_0(a|s)\,[w_{t+1}(s,a)]_+}{\sum_a \pi_0(a|s)\,[w_{t+1}(s,a)]_+}, & \text{otherwise.} \end{cases}$$

  **end**

  Update dual variable,

$$\hat{V}_c^t(\rho) = \langle \rho(\cdot), \langle \hat{Q}_c^t(s,\cdot), \pi_t(\cdot|s)\rangle\rangle$$

$$g_t = b - \hat{V}_c^t(\rho)$$

$$L_t = \max(L_{t-1}, |g_t|)$$

$$\lambda_{t+1} = \lambda_0 + \frac{\sum_{i=0}^t g_i}{L_t \max(\sum_{i=0}^t |g_i| + L_t, \alpha_\lambda L_t)}\left(L_t + \sum_{i=0}^t [(\lambda_i - \lambda_0) g_i]_+\right)$$

**end**

### C.2 OFFLINE G-EXPERIMENTAL DESIGN TO BUILD CORESET $\mathcal{C}$

We use offline G-experimental design to form the coreset in Line 2 of Algorithm 1. In particular, we use the greedy iterative algorithm in Algorithm 3 to build $\mathcal{C}$: in iteration $\tau$, go through all the states and actions adding the $(s, a)$ pair (to $\mathcal{C}$) with the highest marginal gain computed as $g_\tau(s, a) := \|\phi(s, a)\|_{G_\tau^{-1}}$. Here $G_\tau$ is the Gram matrix formed by the features of the $(s, a)$ pairs present in $\mathcal{C}$ at iteration $\tau$. For a specified input $\varepsilon' > 0$, the algorithm terminates at iteration T when $\max_{(s,a)} g_{\mathrm{T}}(s, a) \leq \varepsilon'$. Hence, the algorithm directly controls $\sup_{(s,a)} \|\phi(s, a)\|_{G_\omega^{-1}} \leq \varepsilon'$ in Lemma 5.2, and hence controls $\varepsilon_{\mathrm{s}}$ in practice. However, this procedure does not have a guarantee on how large $|\mathcal{C}|$ can be. In practice, we set $\varepsilon'$ such that $|\mathcal{C}| = O(d)$. Although we only consider forming the coreset in an offline manner that involves iterating through all $SA$ state-action pairs, efficient online variants forming the coreset while running the algorithm have been developed recently (Li et al., 2021). Such techniques are beyond the scope of this paper and we plan to explore them in future work.

**Algorithm 3:** Coreset $\mathcal{C}$ formation using G-experimental design

**Input**: $\Phi$ (Feature map), $\varepsilon' > 0$ (tolerance parameter), $\nu = 1$ (default value).

**Initialize**: $G^{-1} = \frac{1}{\nu}\mathcal{I}_d$, $\mathcal{C} = \emptyset$, $g_{max} = \infty$ (maximum marginal gain).

**while** $g_{max} \leq \varepsilon'$ **do**

  $g_{max} = 0$

  **for** $\forall (s, a) \in (\mathcal{S} \times \mathcal{A})$ **do**

    Compute $g(s, a) = \sqrt{\phi(s, a)^\top G^{-1} \phi(s, a)}$ [marginal gain]

    **if** $g_{max} < g(s, a)$ **then**

      $(s^*, a^*) = (s, a)$

      $g_{max} = g(s, a)$

    **end**

  **end**

  $\mathcal{C} = \mathcal{C} \cup \{(s^*, a^*)\}$

  $G^{-1} = G^{-1} - \frac{G^{-1}\phi(s^*,a^*)\phi(s^*,a^*)^\top G^{-1}}{1+\phi(s^*,a^*)^\top G^{-1}\phi(s^*,a^*)}$ [Sherman-Morrison to compute $\left(G + \phi(s^*, a^*)\,\phi(s^*, a^*)^\top\right)^{-1}$]

**end**

### C.3 SYNTHETIC TABULAR ENVIRONMENT

In Fig. 5, we show the synthetic tabular environment which is modified from Example 3.5 (Sutton and Barto, 2018) to add the constraint rewards.

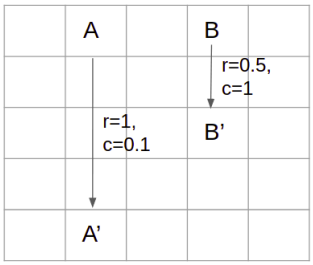



Figure 5: **Tabular environment:** A $5X5$ gridworld environment where all actions results in reward(r) and constraint reward(c) as 0, except special states denoted by $A$ and $B$. All four actions in states $(A, B)$ transitions the agent to states $(A', B')$ and results in reward as $(1, 0.5)$ and constraint rewards as $(0.1, 1)$ respectively. The remaining transitions incur zero reward and zero constrain reward.

### C.4 HYPER-PARAMETERS

Table 1: **Hyperparameters for tabular setting:** Shows the hyperparameters for different algorithms CBP, GDA and CRPO for experiments in Appendix D.1.

| Experiments | **CBP** | **GDA** | **CRPO** |
|---|---|---|---|
| *Model-based* | $\alpha_\lambda = 8$ | $\alpha_\pi = 1.0$, $\alpha_\lambda = 0.1$ | $\alpha_\pi = 0.75, \eta = 0.0$ |
| *Model-free* | $\alpha_\lambda = 8$ | $\alpha_\pi = 1.0$, $\alpha_\lambda = 0.1$ | $\alpha_\pi = 0.75, \eta = 0.0$ |

Table 2: **Hyperparameters for LFA setting with sampling :** Shows the hyperparamters used for different $d$ dimension features for CBP, GDA, CRPO with fixed number of samples for $\hat{Q}$ approximations (for gridworld experiments in Appendix D.2).

| **Algorithms** | $d = 40$ | $d = 56$ | $d = 80$ |
|---|---|---|---|
| **CBP** | $\alpha_\lambda = 0.25$ | $\alpha_\lambda = 0.25$ | $\alpha_\lambda = 0.1$ |
| **GDA** | $\alpha_\pi = 1.0$, $\alpha_\lambda = 0.1$ | $\alpha_\pi = 1.0$, $\alpha_\lambda = 1.0$ | $\alpha_\pi = 1.0$, $\alpha_\lambda = 0.1$ |
| **CRPO** | $\alpha_\pi = 0.75$ | $\alpha_\pi = 0.75$ | $\alpha_\pi = 0.75$ |

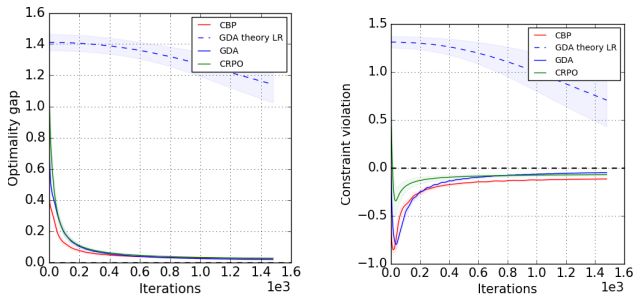
Table 3: **Hyperparameters for Cartpole environment**: Shows the best hyperparameter for different values of entropy regularization coefficient $\nu$ and different algorithms namely CBP, GDA and CRPO (for experiments in Appendix D.2).

| *Algorithms* | $\boldsymbol{\nu = 0}$ | $\boldsymbol{\nu = 0.1}$ | $\boldsymbol{\nu = 0.01}$ | $\boldsymbol{\nu = 0.001}$ |
|---|---|---|---|---|
| CBP | $\alpha_\lambda = 100$ | $\alpha_\lambda = 0.5$ | $\alpha_\lambda = 5.0$ | $\alpha_\lambda = 0.5$ |
| GDA | $\alpha_\pi = 0.01$, $\alpha_\lambda = 0.001$ | $\alpha_\pi = 0.1$, $\alpha_\lambda = 0.001$ | $\alpha_\pi = 0.01$, $\alpha_\lambda = 0.001$ | $\alpha_\pi = 0.1$, $\alpha_\lambda = 0.001$ |
| *CRPO* | $\alpha_\pi = 0.5$, $\eta = 0$ | $\alpha_\pi = 0.5$, $\eta = 0$ | $\alpha_\pi = 0.5$, $\eta = 0$ | $\alpha_\pi = 0.1$, $\eta = 10$ |

# D ADDITIONAL EXPERIMENTAL RESULTS

## D.1 TABULAR SETTING

**Model-based setting:** In Fig. 6, we demonstrate performance – optimality gap (OG) and constraint violation (CV) – with best hyperparameters for three algorithms namely, CBP, GDA and CRPO. In addition, we show the performance of GDA with theoretical learning rates of $\pi$ and $\lambda$ to focus on the importance of tuning GDA's hyperparameter for practical purpose. We observe OG converges to zero quickly for our CBP as compared to GDA and CRPO with constraint satisfaction (when $CV \leq 0$). The ideal performance metric is when both OG and CV converges to 0 value. Refer Table 1 for best values of hyperparameter. We used the following ranges of hyperparameters. For CBP, $\alpha_\lambda = \{1, 2, 5, 8, 15, 50, 100, 300, 500\}$. The hyperparameter of GDA varied as $\alpha_\pi = \{0.001, 0.01, 0.1, 1.0\}$ (learning rate policy) and $\alpha_\lambda = \{0.0001, 0.001, 0.01, 0.1, 1.0\}$ (learning rate dual variable). For CRPO, the learning rate of policy varied as $\alpha_\pi = \{0.001, 0.01, 0.05, 0.1, 0.5, 0.75\}$ and tolerance parameter $\eta = \{0, 0.25\}$.




(a) OG (b) CV

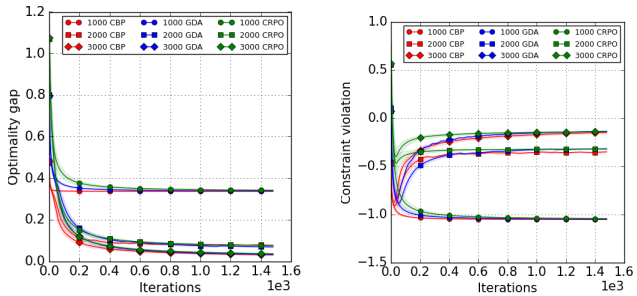
Figure 6: **Model-based in tabular case:** OG and CV for CB, GDA and CRPO with best hyperparameters. The results are averaged over 5 runs with 95% confidence interval. We assume that we have access to true CMDP. The best hyperparameter has the least OG and satisfies the condition $CV \in [-0.25, 0]$. We also show the performance of baseline GDA with theoretical $\alpha_\pi = \sqrt{\frac{2\log|A|}{T}}\frac{1-\gamma}{1+U}$ and $\alpha_\lambda = \frac{U(1-\gamma)}{\sqrt{T}}$. Here, $U = \frac{2}{\zeta(1-\gamma)}$. This is shown in blue dashed line.

**Model-free setting:** Here, we test the performance of algorithms in the model-free setting (don't have access to true CMDP model). We use TD(0) based sampling approach (Sutton, 1988) to estimate the $Q$ action-value function. We sample data for all $(s, a) \in \mathcal{S} \times \mathcal{A}$. In Fig. 7, we observe the effect on performance by varying the number of samples for $Q$ action-value estimation. Here, we consider one-hot encoded features (no overlapping of features). We observe that CBP consistently converges faster than its counterpart in the sampling-based approach. Further, it also matches the expectation that the performance improves with the increase in number of samples. In Figs. 8 and 9, we show the robustness of CBP with hyperparameter sensitivity and environment misspecification respectively. The hyperparameters used for these experiments are presented in Table 1 in Appendix C.4.

## D.2 LINEAR SETTING

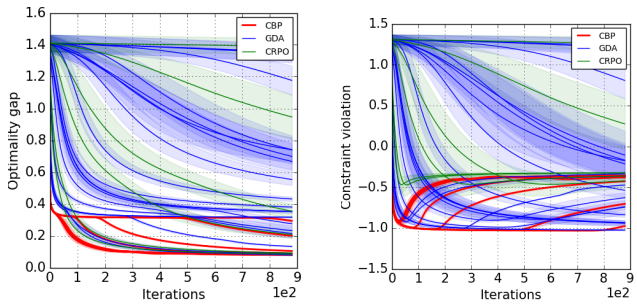
**Gridworld environment:** We use $5 \times 5$ gridworld environment as show in Fig. 5. Tile coding is used to learn the feature representation for every $(s, a)$ pair in the environment. Number of tilings used are 1 and we vary the tiling size to change the dimension of the features (feature overlap for multiple $(s, a)$ pairs). In Fig. 10 we show hyperparameter sensitivity on performance for all the three algorithms with different $d$ dimension of features. The values of all other parameters were kept fixed. Similar observation holds here, CBP is robust to varying values of hyperparameters. The range of hyperparameter is similar to one in Fig. 8.

**G-experimental design for gridworld environment:** In Fig. 11 we show the performance with G-experimental design (Appendix C.2). Here subset of $(s, a) \in \mathcal{C}$ pairs are chosen from a *coreset*.




(a) OG (b) CV

Figure 7: **Model-free tabular case:** We don't have access to true CMDP model here. We vary the number of samples $= \{1000, 2000, 3000\}$ for $Q$ value estimation to observe the effect on the performance. The results are averaged over 5 runs. The performance improves with increase in samples. CBP consistently performs better than the baselines GDA and CRPO.




(a) OG (b) CV

Figure 8: **Sensitivity to hyperparameters in model-free gridworld environment:** Performance with different hyperparameters for CB, GDA and CRPO. The results are averaged over 5 runs with 95% confidence interval. We used 2000 samples for $\forall(s,a) \in \mathcal{S} \times \mathcal{A}$ to estimate $Q$ values for both reward and cost. The results are demonstrated on one-hot features (with no feature overlap). The hyperparameters for CB are $\alpha_\lambda = \{1, 2, 5, 8, 15, 50, 100, 300, 500\}$. The hyperparameter for GDA are $\alpha_\pi = \{0.001, 0.01, 0.1, 1.0\}$ (learning rates for policy) and $\alpha_\lambda = \{0.0001, 0.001, 0.01, 0.1, 1.0\}$ (learning rate for dual variable). The hyperparameter for CRPO are $\alpha_\pi = \{0.001, 0.01, 0.05, 0.1, 0.5, 0.75\}$. We use $\eta = 0.0$ for CRPO. The key observation is that CBP is robust against the variations in hyperparameters with a smaller variance in performance against multiple runs.

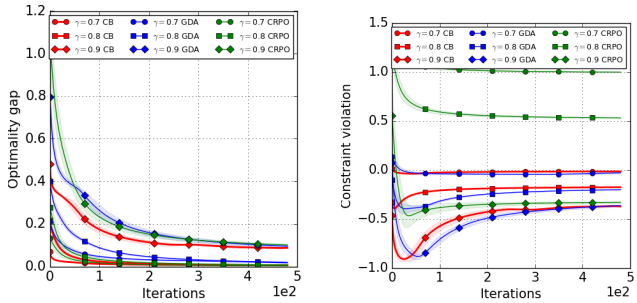
**Exploration in continuous state-spaces:** We used G-experimental design for the discrete state-action environment in the previous section. However, such a procedure is difficult to implement for the continuous state-action spaces we consider in this section. In order to achieve enough exploration in practice, similar to Xu et al. (2021), we use entropy regularization (Geist et al., 2019; Cen et al., 2021) for the policy updates. Specifically, for a specified regularization parameter $\nu$, our task is to find a sequence of policies $\{\pi_0, \pi_1, \ldots, \pi_{T-1}\}$ that minimize the regularized primal regret,

$$\mathcal{R}^p_\nu(\pi^*, T) := \sum_{t=0}^{T-1} \sum_{s=0}^{\mathcal{S}-1} \left[ \langle \pi^*(\cdot|s) - \pi_t(\cdot|s), \hat{Q}^t_r + \lambda_t \hat{Q}^t_c \rangle + \nu \, d^{\pi_t}(s) \sum_{a \in \mathcal{A}} \pi(a|s) \log(\pi(a|s)) \right].$$

It can be easily seen (Geist et al., 2019) that the form of the algorithm updates remain the same, but the action-value functions for policy $\pi$ need to be redefined to depend on the "effective reward" equal to $r(s,a) - \nu \log \pi(a|s)$. Therefore, the new $\hat{Q}^t_l$ with exploration is equal to $\hat{Q}^t_l(s,a) = \hat{Q}^t_r(s,a) + \lambda_t \hat{Q}^t_c(s,a) - \nu \log \pi_t(a|s)$.

**Cartpole environment** : We added two constraint rewards $(c_1, c_2)$ to the classic OpenAI gym Cartpole environment. (1) Cart receives a $c_1 = 0$ constraint reward value when enters the area $[-2.4, -2.2], [-1.3, -1.1], [1.1, 1.3], [2.2, 2.4]$, else receive $c_1 = +1$. (2) When the angle of the cart is less than 4 degrees receive $c_2 = +1$, else everywhere $c_2 = 0$. Each episode length is no longer than 200.

We used tile coding (Sutton and Barto, 2018) to discretize the continuous state space of the environment. The



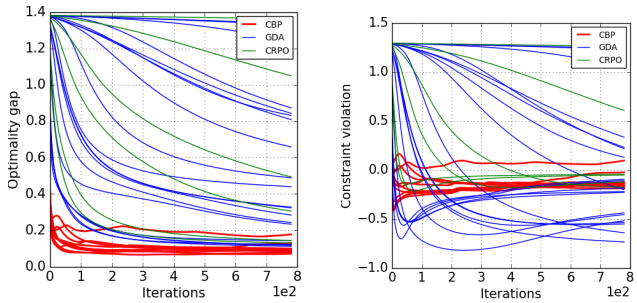

(a) OG (b) CV

Figure 9: **Environment misspecification in model-free gridworld by varying discount factor $\gamma$:** We only introduce sampling error by estimating $Q$ function with 2000 samples for all $(s, a)$ pair for all three algorithms (no feature overlap). We vary the discount factor $\gamma = \{0.7, 0.8\}$ to observe the effects of environment misspecification on CBP, GDA and CRPO. We keep the hyperparameters fixed for all the algorithms, similar to the one for original CMDP with $\gamma = 0.9$. The results are averaged over 5 runs with 95% confidence interval.The hyperparameters used are reported in Table 1. We observe that CRPO does not even satisfy constraint for case when $\gamma = 0.8$ $(CV > 0)$. Further, our CBP converges consistently faster than the baselines.

dimension of the features is $2^{12}$. We used 8 number of tilings with each grid size $4 \times 4$. For exploration, we used entropy regularization with entropy coefficient as $\nu = 0.001$.

We conducted the experiments with following $\alpha_\lambda$ parameter value of CBP $\{0.1, 0.5, 5, 50, 100, 250, 500\}$. For GDA, we varied the learning rate of policy $\alpha_\pi = \{0.1, 0.01, 0.001, 0.0001\}$ and learning rate of dual variable $\alpha_\lambda = \{0.1, 0.01, 0.001, 0.0001\}$. For CRPO baseline, the following values of learning rate of policy $\alpha_\pi = \{0.001, 0.005, 0.01, 0.05, 0.1, 0.5\}$ are experimented with. We kept the tolerance parameter $\eta$ of CRPO as $\{0, 10\}$. The best hyperparamter of CBP is $\alpha_\pi = 0.5$. For GDA it is $\alpha_\pi = 0.01, \alpha_\lambda = 0.0001$. CRPO has best hyperparameter $\alpha_\pi = 0.1, \eta = 10$.

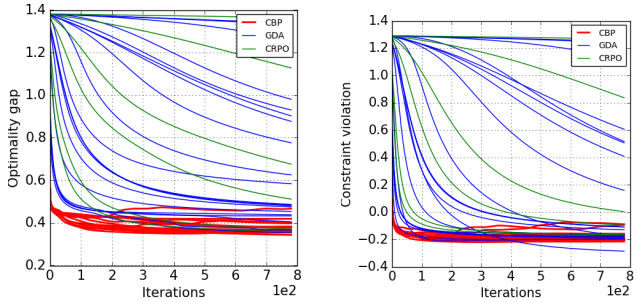
Similarly, we also conducted sensitivity of all three algorithms with entropy regularization $\nu$. In Fig. 12, we vary $\nu$ coefficient and observe effect on the performance.




(a) OG ($d = 40$)



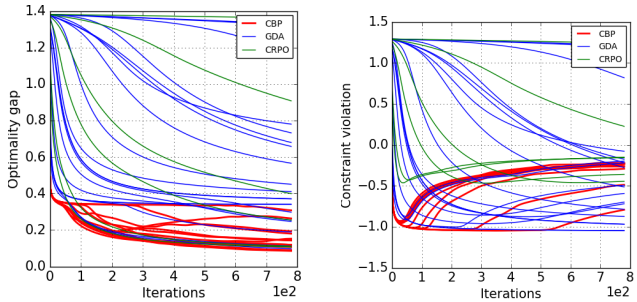

(b) CV ($d = 40$)




(c) OG ($d = 56$)




(d) CV ($d = 56$)

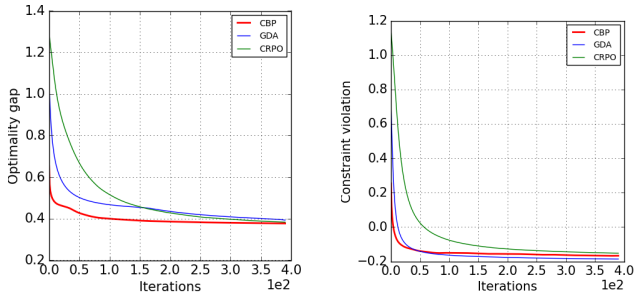



(e) OG ($d = 80$)




(f) CV ($d = 80$)

Figure 10: **Linear function approximation in gridworld environment:** We approximate the $Q$ value using LSTDQ with 300 samples for each $(s, a)$ pair. The dimension of the features (denoted by $d$) are varied to observe the sensitivity to a range of hyperparameter values. We kept all the other parameters fixed. We use (a,b) $d = 40$, (b,c) $d = 56$, (e,f) $d = 80$ dimension features respectively. CBP is consistently robust to variation in the hyperparameters for different dimensions as compared to baselines GDA and CRPO.

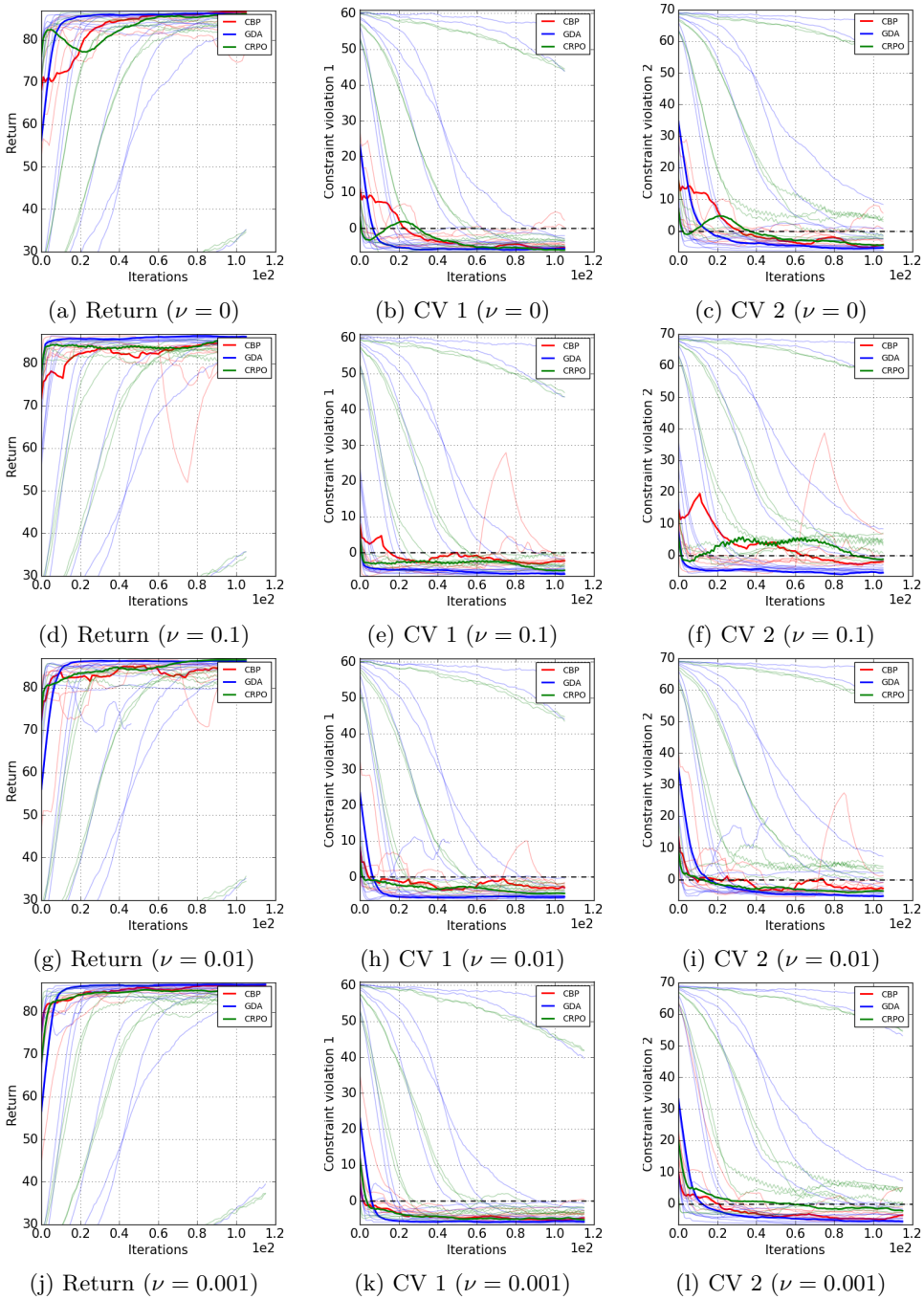



(a) OG




(b) CV

Figure 11: **G-experimental design:** We show the performance with G-experimental design, where subset of $(s, a) \in \mathcal{C}$ are chosen to learn the weight vectors of $Q$ function. Here, we are in model-free setting with $d = 56$ dimensional features in LFA. We used 300 samples for $c \in \mathcal{C}$ to learn the estimate of $Q$ for all three algorithms.




(a) Return ($\nu = 0$) (b) CV 1 ($\nu = 0$) (c) CV 2 ($\nu = 0$)

(d) Return ($\nu = 0.1$) (e) CV 1 ($\nu = 0.1$) (f) CV 2 ($\nu = 0.1$)

(g) Return ($\nu = 0.01$) (h) CV 1 ($\nu = 0.01$) (i) CV 2 ($\nu = 0.01$)

(j) Return ($\nu = 0.001$) (k) CV 1 ($\nu = 0.001$) (l) CV 2 ($\nu = 0.001$)

Figure 12: **Cartpole environment:** We show the sensitivity to entropy regularization $\nu$ for all three algorithms CBP, GDA and CRPO. Different rows corresponds to different value of $\nu = \{0, 0.1, 0.01, 0.001\}$. Darker lines show the performance with the best hyperparameters. Lighter shade lines show performance with other values of hyperparameters. The range of hyperparameter for CBP is $\alpha_\lambda = \{0.1, 0.5, 5, 50, 100, 250, 500\}$. For GDA, we vary learning rates of both policy and dual variable as $\{0.1, 0.01, 0.001, 0.0001\}$. For CRPO, we vary $\alpha_\pi = \{0.1, 0.5, 0.01, 0.001, 0.005\}$ and tolerance hyperparameter as $\eta = \{0, 10\}$.