# COPTIDICE: OFFLINE CONSTRAINED REINFORCEMENT LEARNING VIA STATIONARY DISTRIBUTION CORRECTION ESTIMATION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

We consider the offline constrained reinforcement learning (RL) problem, in which the agent aims to compute a policy that maximizes expected return while satisfying given cost constraints, learning only from a pre-collected dataset. This problem setting is appealing in many real-world scenarios, where direct interaction with the environment is costly or risky, and where the resulting policy should comply with safety constraints. However, it is challenging to compute a policy that guarantees satisfying the cost constraints in the offline RL setting, since the off-policy evaluation inherently has an estimation error. In this paper, we present an offline constrained RL algorithm that optimizes the policy in the space of the stationary distribution. Our algorithm, COptiDICE, directly estimates the stationary distribution corrections of the optimal policy with respect to returns, while constraining the cost upper bound, with the goal of yielding a cost-conservative policy for actual constraint satisfaction. Experimental results show that COptiDICE attains better policies in terms of constraint satisfaction and return-maximization, outperforming baseline algorithms.

## 1 INTRODUCTION

Reinforcement learning (RL) has shown great promise in a wide range of domains, such as complex games (Mnih et al., 2015; Silver et al., 2017) and robotic control (Lillicrap et al., 2016; Haarnoja et al., 2018). However, the need to interact with the environment during learning hinders its widespread application to many real-world problems for which executing exploratory behavior in the environment is costly or dangerous. Offline RL (also known as batch RL) (Lange et al., 2012; Levine et al., 2020) algorithms sidestep this problem and perform policy optimization solely from a set of pre-collected data. The use of existing (offline) data can make offline reinforcement learning applicable to real world systems and and has led to a sharp increase in interest in this paradigm.

Recent works on offline RL, however, mostly assume that the environment is modeled as a Markov decision process (MDP), and standard offline RL algorithms focus on reward-maximization only (Fujimoto et al., 2019; Wu et al., 2019; Kumar et al., 2019; Siegel et al., 2020; Wang et al., 2020; Kumar et al., 2020). In contrast, in real-world domains it is common that the behavior of the agent is subject to additional constraints beyond the reward. Consider, for example, autonomous driving or an industrial robot in a factory. Some behaviors may damage the agent itself or its surroundings, and safety constraints should thus be considered as part of the objective of a suitable reinforcement learning system. One of the ways to mathematically characterize a constrained RL problem is through the formalism of constrained Markov Decision Processes (CMDP) (Altman, 1999). In CMDPs taking an action incurs a cost as well as a reward, and the goal is to maximize the expected long-term reward while satisfying a bound on the expected long-term cost. In this work, we aim to solve the constrained decision making problem in the *offline* RL setting, to enable deployment in various safety-critical domains where direct learning interactions are infeasible.

Offline constrained RL inherits the difficulties of offline unconstrained RL, while introducing additional challenges. First, since the target policy being optimized deviates from the data-collection policy with no further data collection, distribution shift becomes the central difficulty. To mitigate the distributional shift, existing offline RL methods frequently adopt the pessimism principle: either

by explicit policy (and critic) regularization that penalizes deviation from the data-collection policy (Jaques et al., 2019; Wu et al., 2019; Kumar et al., 2019; Siegel et al., 2020; Wang et al., 2020; Lee et al., 2020; Kostrikov et al., 2021) or by reward penalty to the uncertain state-action regions (Kumar et al., 2020; Kidambi et al., 2020; Yu et al., 2020). Second, in offline *constrained* RL, the computed policy should satisfy the given cost constraints when it is deployed to the real environment. Unfortunately off-policy policy evaluation inherently has estimation errors, and it is therefore difficult to ensure that a policy estimated from a finite dataset will satisfy the constraint when executed in the environment. In addition, constrained policy optimization usually involves an additional optimization for the Lagrange multiplier associated with the cost constraints. Actor-critic-based constrained RL algorithms thus have to solve triple (i.e. critic, actor, Lagrange multiplier) intertwined optimization problems (Borkar, 2005; Tessler et al., 2019), which can be very unstable in practice.

In this paper, we present an offline constrained RL algorithm that optimizes the state-action stationary distribution directly, rather than the Q-function or the policy. We show that such treatment obviates the need for multiple estimators for the value and the policy, yielding a single optimization objective that is practically solvable. Still, naively constraining the cost value may result in severe constraint violation in the real environment, as we demonstrate empirically. We thus propose a method to constrain the *upper bound* of the cost value, aiming to compute a policy more robust in constraint violation, where the upper bound is computed in a way motivated by a recent advance in off-policy confidence interval estimation (Dai et al., 2020). Our algorithm, *Offline **C**onstrained Policy **Opti**mization via stationary **DI**stribution **C**orrection **E**stimation* (COptiDICE), estimates the stationary distribution corrections of the optimal policy that maximizes rewards while constraining the cost upper bound, with the goal of yielding a cost-conservative policy for better actual constraint satisfaction. COptiDICE computes the upper bound of the cost efficiently by solving an additional minimization problem. Experimental results show that COptiDICE attains a better policy in terms of constraint satisfaction and reward-maximization, outperforming several baseline algorithms.

## 2 BACKGROUND

A Constrained Markov Decision Process (CMDP) (Altman, 1999) is an extension of an MDP, formally defined by a tuple $M = \langle S, A, T, R, C = \{C_k\}_{1..K}, \hat{c} = \{\hat{c}_k\}_{1..K}, p_0, \gamma \rangle$, where $S$ is the set of states, $A$ is the set of actions, $T : S \times A \to \Delta(S)$ is a transition probability, $R : S \times A \to \mathbb{R}$ is the reward function, $C_k : S \times A \to \mathbb{R}$ is the $k$-th cost function with its corresponding threshold $\hat{c}_k \in \mathbb{R}$, $p_0 \in \Delta(S)$ is the initial state distribution, and $\gamma \in (0, 1]$ is the discount factor. A policy $\pi : S \to \Delta(A)$ is a mapping from state to distribution over actions. We will express solving CMDPs in terms of stationary distribution. For a given policy $\pi$, the stationary distribution $d^\pi$ is defined by:

$$d^\pi(s,a) = \begin{cases} (1-\gamma) \sum_{t=0}^{\infty} \gamma^t \Pr(s_t = s, a_t = a) & \text{if } \gamma < 1, \\ \lim_{T\to\infty} \frac{1}{T+1} \sum_{t=0}^{T} \Pr(s_t = s, a_t = a) & \text{if } \gamma = 1, \end{cases} \tag{1}$$

where $s_0 \sim p_0,\ a_t \sim \pi(s_t),\ s_{t+1} \sim T(s_t, a_t)$ for all timesteps $t \geq 0$. We define the value of the policy as $V_R(\pi) := \mathbb{E}_{(s,a)\sim d^\pi}[R(s,a)] \in \mathbb{R}$ and $V_C(\pi) := \mathbb{E}_{(s,a)\sim d^\pi}[C(s,a)] \in \mathbb{R}^K$. Constrained RL aims to learn an optimal policy that maximizes the reward while bounding the costs up to the thresholds by interactions with the environment:

$$\max_\pi \ V_R(\pi) \ \text{ s.t. } V_{C_k}(\pi) \leq \hat{c}_k \quad \forall k = 1, \ldots, K \tag{2}$$

Lagrangian relaxation is typically employed to solve Eq. (2), leading to the unconstrained problem:

$$\min_{\lambda \geq 0} \max_\pi V_R(\pi) - \lambda^\top (V_C(\pi) - \hat{c}) \tag{3}$$

where $\lambda \in \mathbb{R}^K$ is the Lagrange multiplier and $\hat{c} \in \mathbb{R}^K$ is the vector-valued cost threshold. The inner maximization in Eq. (3) corresponds to computing an optimal policy that maximizes scalarized rewards $R(s,a) - \lambda^\top C(s,a)$, due to the linearity of the value function with respect to the reward function. The outer minimization corresponds to balancing the cost penalty in the scalarized reward function: if the current policy is violating the $k$-th cost constraint, $\lambda_k$ increases so that the cost is penalized more in the scalarized reward, and vice versa.

In the offline RL setting, online interaction with the environment is not allowed, and the policy is optimized using the fixed offline dataset $D = \{(s_0, s, a, r, c, s')_i\}_{i=1}^N$ collected with one or more (unknown) data-collection policies. The empirical distribution of the dataset is denoted as $d^D$, and we will abuse the notation $d^D$ for $s \sim d^D$, $(s, a) \sim d^D$, $(s, a, s') \sim d^D$. We abuse $(s_0, s, a, s') \sim d^D$ for $s_0 \sim p_0, (s, a, s') \sim d^D$. We denote the space of data samples $(s_0, s, a, s')$ as $X$.

A naive way to solve (3) in an offline manner is to adopt an actor-critic based offline RL algorithm for $\max_\pi V_{R-\lambda^\top C}(\pi)$ while jointly optimizing $\lambda$. However, the intertwined training procedure of off-policy actor-critic algorithms often suffers from instability due to the compounding error incurred by bootstrapping out-of-distribution action values in an offline RL setting (Kumar et al., 2019). The instability would be exacerbated when the nested optimization for $\lambda$ is added.

## 3 Offline Constrained RL via Stationary Distribution Correction Estimation

In this section, we present our offline constrained RL algorithm, ***C**onstrained Policy **Opti**mization via stationary **DI**stribution **C**orrection **E**stimation* (COptiDICE). The derivation of our algorithm starts by augmenting the standard linear program for CMDP (Altman, 1999) with an additional $f$-divergence regularization[1]:

$$\max_d \mathbb{E}_{(s,a)\sim d}[R(s,a)] - \alpha D_f(d||d^D) \tag{4}$$

$$\text{s.t. } \mathbb{E}_{(s,a)\sim d}[C_k(s,a)] \le \hat{c}_k \qquad \forall k = 1, \dots, K \tag{5}$$

$$\sum_{a'} d(s', a') = (1-\gamma)p_0(s') + \gamma \sum_{s,a} d(s,a)T(s'|s,a) \qquad \forall s' \tag{6}$$

$$d(s,a) \ge 0 \qquad \forall s, a, \tag{7}$$

where $D_f(d||d^D) := \mathbb{E}_{(s,a)\sim d^D}\left[f\left(\frac{d(s,a)}{d^D(s,a)}\right)\right]$ is the $f$-divergence between the distribution $d$ and the dataset distribution $d^D$, and $\alpha > 0$ is the hyperparameter that controls the degree of pessimism, i.e. how much we penalize the distribution shift, a commonly adopted principle for offline RL (Nachum et al., 2019b; Kidambi et al., 2020; Yu et al., 2020; Lee et al., 2021). We assume that $d^D > 0$ and $f$ is a strictly convex and continuously differentiable function with $f(1) = 0$. Note that when $\alpha = 0$, the optimization (4-7) reduces to the standard linear program for CMDPs. The Bellman-flow constraints (6-7) ensure that $d$ is the stationary distribution of a some policy, where $d(s, a)$ can be interpreted as a normalized discounted occupancy measure of $(s, a)$. Thus, we seek the stationary distribution of an optimal policy that maximizes the reward value (4) while bounding the cost values (5). Once the optimal solution $d^*$ has been estimated, its corresponding optimal policy is obtained by $\pi^*(a|s) = \frac{d^*(s,a)}{\sum_{a'} d^*(s,a')}$.

Now, consider the Lagrangian for the constrained optimization problem (4-7):

$$\begin{aligned}\min_{\lambda\ge 0,\nu} \max_{d\ge 0} \mathbb{E}_{(s,a)\sim d}[R(s,a)] - \alpha D_f(d||d^D) - \sum_{k=1}^K \lambda_k\big(\mathbb{E}_{(s,a)\sim d}[C_k(s,a)] - \hat{c}_k\big) \\ - \sum_{s'} \nu(s')\Big[\sum_{a'} d(s',a') - (1-\gamma)p_0(s') - \gamma \sum_{s,a} d(s,a)T(s'|s,a)\Big]\end{aligned} \tag{8}$$

where $\lambda \in \mathbb{R}_+^K$ is the Lagrange multiplier for the cost constraints (5), and $\nu(s) \in \mathbb{R}$ is the Lagrange multiplier for the Bellman flow constraints (6). Solving (8) in its current form requires evaluation of $T(s'|s, a)$ for $(s, a) \sim d$, which is not accessible in the offline RL setting. To make the optimization tractable, we rearrange the terms so that the direct dependence on $d$ is eliminated, while introducing new optimization variables $w$ that represent stationary distribution corrections:

$$\begin{aligned}&\min_{\lambda\ge 0,\nu} \max_{d\ge 0} \mathbb{E}_{\substack{(s,a)\sim d \\ s'\sim T(s,a)}}\left[R(s,a) - \lambda^\top C(s,a) + \nu(s') - \nu(s)\right] - \alpha \mathbb{E}_{(s,a)\sim d^D}\left[f\left(\frac{d(s,a)}{d^D(s,a)}\right)\right] \\ &\qquad + (1-\gamma)\mathbb{E}_{s_0\sim p_0}[\nu(s_0)] + \lambda^\top \hat{c} \\ &= \min_{\lambda\ge 0,\nu} \max_{w\ge 0} \mathbb{E}_{(s,a)\sim d^D}\left[w(s,a)e_{\lambda,\nu}(s,a) - \alpha f(w(s,a))\right] + (1-\gamma)\mathbb{E}_{s_0\sim p_0}[\nu(s_0)] + \lambda^\top \hat{c}\end{aligned} \tag{9}$$

[1]For simplicity, we only consider discounted CMDPs ($\gamma < 1$) here, but the result can be generalized to the case of average reward-cost CMDPs ($\gamma = 1$) (Appendix A).

where $e_{\lambda,\nu}(s,a) := R(s,a) - \lambda^\top C(s,a) + \gamma \mathbb{E}_{s' \sim T(s,a)}[\nu(s')] - \nu(s)$ is the advantage function by regarding $\nu$ as a state value function, and $w(s,a) := \frac{d(s,a)}{d^D(s,a)}$ is the stationary distribution correction. Every term in Eq. (9) can be estimated from samples in the offline dataset $D$:

$$\min_{\nu,\lambda \geq 0} \max_{w \geq 0} \mathbb{E}_{(s_0,s,a,s') \sim d^D} \left[ w(s,a) \hat{e}_{\lambda,\nu}(s,a,s') - \alpha f(w(s,a)) + (1-\gamma)\nu(s_0) \right] + \lambda^\top \hat{c} \tag{10}$$

where $\hat{e}_{\lambda,\nu}(s,a,s') := R(s,a) - \lambda^\top C(s,a) + \gamma \nu(s') - \nu(s)$ is the advantage estimate using a single sample. As a consequence, (10) can be optimized in a fully offline manner. Moreover, exploiting the strict convexity of $f$, we can further derive a closed-form solution for the inner maximization in (9) as follows. All the proofs can be found in Appendix B.

**Proposition 1.** *For any $\nu$ and $\lambda$, the closed-form solution of the inner maximization problem in* (9) *is given by:*

$$w^*_{\lambda,\nu}(s,a) = (f')^{-1}\left(\tfrac{1}{\alpha} e_{\lambda,\nu}(s,a)\right)_+ \textit{ where } x_+ = \max(0,x) \tag{11}$$

Finally, by plugging the closed-form solution (11) into (9), we obtain the following convex minimization problem:

$$\min_{\lambda \geq 0, \nu} L(\lambda,\nu) = \mathbb{E}_{(s,a)\sim d^D}\left[ w^*_{\lambda,\nu}(s,a) e_{\lambda,\nu}(s,a) - \alpha f(w^*_{\lambda,\nu}(s,a)) \right] + (1-\gamma)\mathbb{E}_{s_0 \sim p_0}[\nu(s_0)] + \lambda^\top \hat{c} \tag{12}$$

To sum up, by operating in the space of stationary distributions, constrained (offline) RL can in principle be solved by solving a single convex minimization (12) problem. This is in contrast to existing constrained RL algorithms that manipulate both Q-function and policy, and thus require solving triple optimization problems for the actor, the critic, and the cost Lagrange multiplier with three different objective functions. Note also that when $\lambda$ is fixed and treated as a constant, (12) reduces to OptiDICE (Lee et al., 2021) for unconstrained RL with the scalarized rewards $R(s,a) - \lambda^\top C(s,a)$, without considering the cost constraints. In order to meet the constraints, $\lambda$ should also be optimized, and the procedure of (12) can be understood as joint optimization of:

$$\nu \leftarrow \arg\min_{\nu} L(\lambda,\nu) \qquad \text{(OptiDICE for } R - \lambda^\top C) \tag{13}$$

$$\lambda \leftarrow \arg\min_{\lambda \geq 0} \lambda^\top \big( \hat{c} - \underbrace{\mathbb{E}_{(s,a)\sim d^D}[w^*_{\lambda,\nu}(s,a) C(s,a)]}_{\approx V_C(\pi)} \big) \qquad \text{(Cost Lagrange multiplier)}$$

Once the optimal solution of (12), $(\lambda^*, \nu^*)$, is computed, $w^*_{\lambda^*,\nu^*}(s,a) = \frac{d^{\pi^*}(s,a)}{d^D(s,a)}$ is also derived by (11), which is the stationary distribution correction between the stationary distribution of the optimal policy for the CMDP and the dataset distribution.

### 3.1 Cost-conservative Constrained Policy Optimization

Our first method based on (12) relies on off-policy evaluation (OPE) using DICE to ensure cost constraint satisfaction, i.e. $\mathbb{E}_{(s,a)\sim d^D}[w_{\lambda,\nu}(s,a)C(s,a)] \approx \mathbb{E}_{(s,a)\sim d^\pi}[C(s,a)] \leq \hat{c}$. However, as we will see later, constraining the cost value estimate naively can result in constraint violation when deployed to the real environment. This is due to the fact that an off-policy value estimate based on a finite dataset inevitably has estimation error. Reward estimation error may be tolerated as long as the value estimates are useful as policy improvement signals: it may be sufficient to maintain the relative order of action values, while the absolute values matter less. For the cost value constraint, we instead rely on the estimated value directly.

To make a policy robust against cost constraint violation in an offline setting, we consider the constrained policy optimization scheme that exploits the *upper bound* of the cost value estimate:

$$\max_{\pi} \hat{V}_R(\pi) \quad \text{s.t. } \mathrm{UpperBound}(\hat{V}_{C_k}(\pi)) \leq \hat{c}_k \quad \forall k \tag{14}$$

Then, the key question is how to estimate the upper bound of the policy value. One natural way is to exploit bootstrap confidence interval (Efron & Tibshirani, 1993; Hanna et al., 2017). We can

construct bootstrap datasets $D_i$ by resampling from $D$ and run an OPE algorithm on each $D_i$, which yields population statistics for confidence interval estimation $\{\hat{V}_C(\pi)_i\}_{i=1}^m$. However, this procedure is computationally very expensive since it requires solving $m$ OPE tasks. Instead, we take a different approach in a more computationally efficient way motivated by CoinDICE (Dai et al., 2020), a recently proposed DICE-family algorithm for off-policy confidence interval estimation. Specifically, given that our method estimates the stationary distribution corrections $w(s,a) \approx \frac{d^\pi(s,a)}{d^D(s,a)}$ of the target policy $\pi$, we consider the following optimization problem for each cost function $C_k$:

$$\max_{\tilde{p}\in\Delta(X)} \mathbb{E}_{(s_0,s,a,s')\sim\tilde{p}}[w(s,a)C_k(s,a)] \tag{15}$$

$$\text{s.t. } D_{\mathrm{KL}}(\tilde{p}(s_0,s,a,s')||d^D(s_0,s,a,s')) \le \epsilon \tag{16}$$

$$\sum_{a'} \tilde{p}(s',a')w(s',a') = (1-\gamma)\tilde{p}_0(s') + \gamma \sum_{s,a} \tilde{p}(s,a)w(s,a)\tilde{p}(s'|s,a) \quad \forall s' \tag{17}$$

where $\tilde{p}(s_0,s,a,s') = \tilde{p}_0(s_0)\tilde{p}(s,a)\tilde{p}(s'|s,a)$ is the distribution over data samples $(s_0,s,a,s') \in X$ which lies in the simplex $\Delta(X)$, and $\epsilon > 0$ is the hyperparameter. In essence, we want to adversarially optimize the distribution over data samples $\tilde{p}$ so that it overestimates the cost value by (15). At the same time, we enforce that the distribution $\tilde{p}$ should not be perturbed too much from the empirical data distribution $d^D$ by the KL constraint (16). Lastly, the perturbation of distribution should be done in a way that maintains compatibility with the Bellman flow constraint. The constraint (17) is analogous to the Bellman flow constraint (6) by noting that $\tilde{p}(s,a)w(s,a) = \tilde{p}(s,a)\frac{d^\pi(s,a)}{d^D(s,a)} \approx d^\pi(s,a)$. In this optimization, when $\epsilon = 0$, the optimal solution is simply given by $\tilde{p}^* = d^D$, which yields the vanilla OPE result via DICE, i.e. $\mathbb{E}_{(s,a)\sim d^D}[w(s,a)C_k(s,a)]$. For $\epsilon > 0$, the cost value will be overestimated more as $\epsilon$ increases. Through a derivation similar to that for obtaining (12), we can simplify the constrained optimization into a single unconstrained minimization problem as follows. We denote $(s_0,s,a,s')$ as $x$ for notational brevity.

**Proposition 2.** *The constrained optimization problem (15-17) can be reduced to solving the following unconstrained minimization problem:*

$$\min_{\tau\ge 0,\chi} \ell_k(\tau,\chi;w) = \tau \log \mathbb{E}_{x\sim d^D}\left[\exp\left(\tfrac{1}{\tau}\big(w(s,a)(C_k(s,a)+\gamma\chi(s')-\chi(s)) + (1-\gamma)\chi(s_0)\big)\right)\right] + \tau\epsilon \tag{18}$$

*where $\tau \in \mathbb{R}_+$ corresponds to the Lagrange multiplier for the constraint* (16)*, and $\chi(s) \in \mathbb{R}$ corresponds to the Lagrange multiplier for the constraint* (17)*. In other words, $\min_{\tau\ge 0,\chi} \ell(\tau,\chi) = \mathbb{E}_{(s,a)\sim\tilde{p}^*}[w(s,a)C_k(s,a)]$ where $\tilde{p}^*$ is the optimal perturbed distribution of the problem (15-17). Also, for the optimal solution $(\tau^*,\chi^*)$, $\tilde{p}^*$ is given by:*

$$\tilde{p}^*(x) \propto d^D(x) \underbrace{\exp\left(\tfrac{1}{\tau^*}\big(w(s,a)(C_k(s,a)+\gamma\chi^*(s')-\chi^*(s)) + (1-\gamma)\chi^*(s_0)\big)\right)}_{=:\,\omega^*(x)\ \textit{(unnormalized weight for } x=(s_0,s,a,s'))} \tag{19}$$

Note that every term in (18) can be estimated only using samples of the offline dataset $D$, thus it can be optimized in a fully offline manner. This procedure can be understood as computing the weights for each sample while adopting *reweighting* in the DICE-based OPE, i.e. $\mathrm{UpperBound}(\hat{V}_C(\pi)) = \mathbb{E}_{x\sim d^D}[\tilde{\omega}^*(x)w(s,a)C(s,a)]$ where $\tilde{\omega}^*(x) = (\text{normalized } \omega^*(x) \text{ of (19)})$. The weights are given non-uniformly so that the cost value is overestimated to the extent controlled by $\epsilon$.

**Remark.** *CoinDICE (Dai et al., 2020) solves the similar optimization problem to estimate an upper cost value of the target policy $\pi$ as follows:*

$$\max_{w\ge 0} \min_{\tau\ge 0,\nu} \tau \log \mathbb{E}_{\substack{x\sim d^D \\ a_0\sim\pi(s_0) \\ a'\sim\pi(s')}}\left[\exp\left(\tfrac{1}{\tau}\big(w(s,a)(C_k(s,a)+\gamma\nu(s',a')-\nu(s',a')) + (1-\gamma)\nu(s_0,a_0)\big)\right)\right] + \tau\epsilon \tag{20}$$

*It is proven that* (20) *provides an asymptotic $(1-\alpha)$-upper-confidence-interval of the policy value if $\epsilon := \frac{\xi_\alpha}{N}$ where $\xi_\alpha$ is the $(1-\alpha)$-quantile of the $\chi^2$-distribution with 1 degree of freedom (Dai et al., 2020). Compared to our optimization problem* (18)*, CoinDICE's* (20) *involves the additional outer maximization, which is for estimating $w(s,a) = \frac{\tilde{d}^\pi(s,a)}{d^D(s,a)}$, the stationary distribution corrections of the target policy $\pi$. In contrast, we consider the case when $w$ is given, thus solving the inner minimization alone is enough.*

Finally, we are ready to present our final algorithm COptiDICE, an offline constrained RL algorithm that maximizes rewards while bounding the *upper* cost value, with the goal of computing a policy robust against cost violation. COptiDICE addresses (14) by solving the following joint optimization.

$$\begin{aligned}
\nu &\leftarrow \arg\min_{\nu} L(\lambda, \nu) && \text{(OptiDICE for } R - \lambda^\top C) \qquad (21)\\
\tau, \chi &\leftarrow \arg\min_{\tau \geq 0, \chi} \textstyle\sum_{k=1}^{K} \ell_k(\tau_k, \chi_k; w^*_{\lambda,\nu}) && \text{(Upper cost value estimation)}\\
\lambda &\leftarrow \arg\min_{\lambda \geq 0} \lambda^\top \big(\hat{c} - \underbrace{\ell(\tau, \chi; w^*_{\lambda,\nu})}_{\approx \mathrm{UpperBound}(\hat{V}_C(\pi))}\big) && \text{(Cost Lagrange multiplier)}
\end{aligned}$$

Compared to (13), the additional minimization for $(\tau, \chi)$ is introduced to estimate the upper bound of cost value.

### 3.2 Policy Extraction

Our algorithm estimates the stationary distribution corrections of the optimal policy, rather than directly obtaining the policy itself. Since the stationary distribution corrections do not provide a direct way to sample an action, we need to extract the optimal policy $\pi^*$ from $w^*(s,a) = \frac{d^{\pi^*}(s,a)}{d^D(s,a)}$, in order to select actions when deployed. For finite CMDPs, it is straightforward to obtain $\pi^*$ by $\pi^*(a|s) = \frac{d^{\pi^*}(s,a)}{\sum_{a'} d^{\pi^*}(s,a')} = \frac{d^D(s,a) w^*(s,a)}{\sum_{a'} d^D(s,a) w^*(s,a)}$. However, the same method cannot directly be applied to continuous CMDPs due to the intractability of computing the normalization constant. For continuous CMDPs, we instead extract the policy using importance-weighted behavioral cloning:

$$\max_{\pi} \mathbb{E}_{(s,a)\sim d^{\pi^*}}[\log \pi(a|s)] = \mathbb{E}_{(s,a)\sim d^D}[w^*(s,a) \log \pi(a|s)] \qquad (22)$$

which maximizes the log-likelihood of actions to be selected by the optimal policy $\pi^*$.

### 3.3 Practical Algorithm with Function Approximation

For continuous or large CMDPs, we represent our optimization variables using neural networks. The Lagrange multipliers $\nu$ and $\chi$ are networks parameterized by $\theta$ and $\phi$ respectively: $\nu_\theta : S \to \mathbb{R}$ is a feedforward neural network that takes a state as an input and outputs a scalar value, and $\chi_\phi : S \to \mathbb{R}^K$ is defined similarly. $\lambda \in \mathbb{R}^K_+$ and $\tau \in \mathbb{R}^K_+$ are represented by $K$-dimensional vectors. For the policy $\pi_\psi$, we use a mixture density network (Bishop, 1994) where the parameters of a Gaussian mixture model are output by the neural network. The parameters of the $\nu_\theta$ network are trained by minimizing the loss:

$$\begin{aligned}
\min_{\theta} J_\nu(\theta) =& \mathbb{E}_{x\sim d^D}\big[\hat{w}(s,a,s')(R(s,a) - \lambda^\top C(s,a) + \gamma \nu_\theta(s') - \nu_\theta(s)) \qquad (23)\\
&- \alpha f(\hat{w}(s,a,s')) + (1-\gamma)\nu_\theta(s_0)\big] + \lambda^\top \hat{c}
\end{aligned}$$

where $\hat{w}(s,a,s') := (f')^{-1}\big(\frac{1}{\alpha}(R(s,a) - \lambda^\top C(s,a) + \gamma\nu_\theta(s') - \nu_\theta(s))\big)_+$. While $J^\lambda_\nu$ can be a biased estimate of $L(\lambda, \nu)$ in (12) in general due to $(f')^{-1}(\mathbb{E}[\cdot]) \neq \mathbb{E}[(f')^{-1}(\cdot)]$, we can show that $J^\lambda_\nu$ is an upper bound of $L(\nu, \lambda)$ (i.e. we minimize the upper bound), and $J^\lambda_\nu = L(\nu, \lambda)$ holds if transition dynamics are deterministic (e.g. Mujoco control tasks). The parameters of the $\chi_\phi$ network and $\tau$ can be trained by:

$$\begin{aligned}
\min_{\tau \geq 0, \phi} \sum_{k=1}^{K} \tau \log \mathbb{E}_{x\sim d^D}\Big[\exp\big(\tfrac{1}{\tau}\big(\hat{w}(s,a,s')(C_k(s,a) + \gamma\chi_{\phi,k}(s') - \chi_{\phi,k}(s)) \qquad (24)\\
+ (1-\gamma)\chi_{\phi,k}(s_0)\big)\big)\Big] + \tau_k \epsilon
\end{aligned}$$

This involves a logarithm outside of the expectation, which implies that mini-batch approximations would introduce a bias. Still, we adopt the simple mini-batch approximation for computational efficiency, with a moderately large batch size (e.g. 1024), which worked well in practice. The empirical form of the loss we use is given by:

$$\begin{aligned}
\min_{\tau \geq 0, \phi} J_{\tau,\chi}(\tau, \phi) =& \mathbb{E}_{\mathrm{batch}(D)\sim D}\Big[\sum_{k=1}^{K} \tau \log \mathbb{E}_{x\sim \mathrm{batch}(D)}\big[\exp\big(\tfrac{1}{\tau}\big(\hat{w}(s,a,s')\cdot \qquad (25)\\
&(C_k(s,a) + \gamma\chi_{\phi,k}(s') - \chi_{\phi,k}(s)) + (1-\gamma)\chi_{\phi,k}(s_0)\big)\big)\big] + \tau_k\epsilon\Big]
\end{aligned}$$

Lastly, $\lambda$ and the policy parameter $\psi$ are optimized by:

$$\min_{\lambda \geq 0} J_\lambda(\lambda) = \lambda^\top (\hat{c} - J_{\tau,\chi}(\tau, \phi)) \tag{26}$$

$$\min_{\psi} J_\pi(\psi) = -\mathbb{E}_{x \sim d^D}[\hat{w}(s, a, s') \log \pi_\psi(a|s)] \tag{27}$$

The complete pseudo-code is described in Appendix C, where every parameter is optimized jointly.

# 4 EXPERIMENTS

## 4.1 TABULAR CMDPS (RANDOMLY GENERATED CMDPS)

We first probe how COptiDICE can improve the reward performance beyond the data-collection policy while satisfying the given cost constraint via repeated experiments. We follow a protocol similar to the random MDP experiment in (Laroche et al., 2019; Lee et al., 2021) but with cost as an additional consideration. We conduct repeated experiments for 10K runs, and for each run, a CMDP $M$ is generated randomly with the cost threshold $\hat{c} = 0.1$. We test with two types of data-collection policy $\pi_D$, a constraint-satisfying policy (i.e. $V_C(\pi_D) = 0.09$) and a constraint-violating policy (i.e. $V_C(\pi_D) = 0.11$). Then, a varying number of trajectories $N \in \{10, 20, 50, 100, 200, 500, 1000, 2000\}$ are collected from the sampled CMDP using the constructed data-collection policy $\pi_D$, which constitutes the offline dataset $D$. Finally, the offline dataset $D$ is given to each offline constrained RL algorithm, and its reward and cost performance is evaluated. For the reward, we evaluate the normalized performance of the policy $\pi$ by $\frac{V_R(\pi) - V_R(\pi_D)}{V_R(\pi^*) - V_R(\pi_D)} \in (-\infty, 1]$ to see the performance improvement of $\pi$ over $\pi_D$ intuitively, where $\pi^*$ is the optimal policy of the underlying CMDP. More details can be found in Appendix D.

Offline *constrained* RL has been mostly unexplored, thus lacks published baseline algorithms. We consider the following two baselines. First, `Baseline` denotes the algorithm that constructs a maximum-likelihood estimation (MLE) CMDP $\hat{M}$ using $D$ and then solves the MLE CMDP using a tabular CMDP solver (LP solver) (Altman, 1999). Second, `C-SPIBB` is the variant of SPIBB (an offline RL method for tabular MDPs) (Laroche et al., 2019), where we modified SPIBB to deal with the cost constraint by Lagrange relaxation. Specifically, C-SPIBB solves the offline RL problem with respect to $R - \lambda C$ via SPIBB while updating $\lambda$ in the direction of $(\hat{V}_C(\pi_{\text{SPIBB}}) - \hat{c})$, where $\hat{V}_C(\pi_{\text{SPIBB}})$ is evaluated using the MLE CMDP.

Figures 1a-b presents the result when the data-collection policy is constraint-satisfying. When the size of the offline dataset is very small, `Baseline` severely violates the cost constraint when its computed policy is deployed to the real environment (red curve in Figure 1b), and it even fails to improve the reward performance over $\pi_D$ (red curve in Figure 1a). This result is expected since Baseline overfits to the MLE CMDP, exploiting the highly uncertain state-actions. This can cause a significant gap between the stationary distribution of the optimized policy computed in $M$ and the one computed in $\hat{M}$, leading to failure in both reward performance improvement and cost constraint satisfaction. To prevent such distributional shift, offline RL algorithms commonly adopt the pessimism principle, encouraging staying close to the data support. We can observe that such pessimism principle is also effective in offline constrained RL: both C-SPIBB (orange) and the naive version of COptiDICE (green) show consistent policy improvement over the data-collection policy while showing better constraint satisfaction. Still, the pessimism principle alone is not sufficient to ensure constraint satisfaction, raising the need for additional treatment for cost-conservativeness. Finally, our COptiDICE (blue) shows much stricter cost satisfaction than other baseline algorithms while outperforming Baseline and C-SPIBB in terms of reward performance.

Figures 1c-d presents the result when the data-collection policy is constraint-violating, where all the baseline algorithms exhibit severe cost violation. This result shows that if the agent is encouraged to stay close to the constraint-violating policy, it may negatively affect the constraint satisfaction, although the pessimism principle was beneficial in terms of reward maximization. In this situation, only COPtiDICE (blue) could meet the given constraint in general, which demonstrates the effectiveness of our proposed method of constraining the upper bound of cost value. Although COptiDICE (blue) shows reward performance degradation when the size of dataset is very small, this is natural in that it sacrifices the reward performance to lower the cost value to meet the constraint conservatively, and it still outperforms the baseline in this low-data regime.

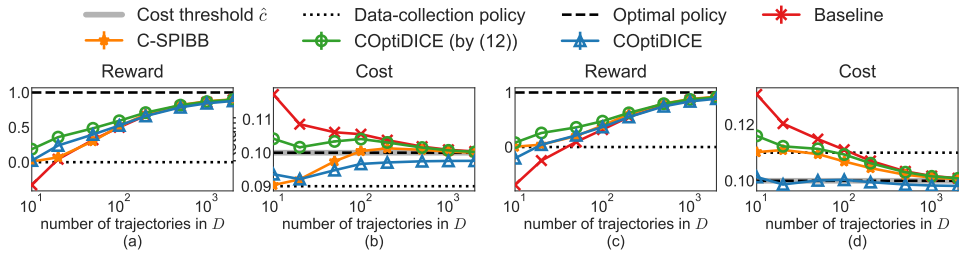


Figure 1: Result of tabular COptiDICE and baseline algorithms in random tabular CMDPs for the varying number of trajectories and two types of data-collection policies. Plots (a)-(b) correspond to the case when the data-collection policy is constraint-satisfying, while the data-collection policy is constraint-violating in the last two plots (c)-(d). The mean of normalized reward performance and the mean of cost value are reported for 10K runs, where the error bar denotes the standard error.

### 4.2 Continuous control tasks (RWRL Suite)

We also evaluate COptiDICE on domains from the Real-World Reinforcement Learning (RWRL) suite (Dulac-Arnold et al., 2020), where the safety constraints are employed. The cost of 1 is given if the task-specific safety constraint is violated at each time step, and the goal is to compute a policy that maximizes rewards while bounding the average cost up to $\hat{c}$. Per-domain safety constraints and the cost constraint thresholds are given in Appendix D. Due to lack of an algorithm that addresses offline constrained RL in continuous action space, we compare COptiDICE with simple baseline algorithms, i.e. BC: A simple behavior-cloning agent, and CRR (Wang et al., 2020): a state-of-the-art (unconstrained) offline RL algorithm.

Since there is no standard dataset for offline constrained RL, we collected data using online constrained RL agents (C-DMPO; the constrained variant of Distributional MPO) (Abdolmaleki et al., 2018; Mankowitz et al., 2021). We trained the online C-DMPO with various cost thresholds $\hat{c}$ and saved checkpoints at regular intervals, which constitutes the pool of policy checkpoints. Then, we generated datasets, where each of them consists of $\beta \times 100\%$ constraint-satisfying trajectories and the rest constraint-violating trajectories. The trajectories were sampled by policies in the pool of policy checkpoints. Since there is a trade-off between reward and cost in general, the dataset is a mixture of low-reward-low-cost trajectories and high-reward-high-cost trajectories.

Figure 2 presents our results in RWRL tasks, where the dataset contains mostly constraint-satisfying trajectories ($\beta = 0.8$), with some cost-violating trajectories. This type of dataset is representative of many practical scenarios: the data-collecting agent behaves safely in most cases, but sometimes exhibit exploratory behaviors which can be leveraged for potential performance improvement. Due to the characteristics of these datasets, BC (orange) generally yields constraint-satisfying policy, but its reward performance is also very low. CRR (red) significantly improves reward performance over BC, but it does not ensure that the constraints are satisfied. The naive COptiDICE algorithm (green) takes the cost constraint into account but nevertheless frequently violates the constraint due to limitations of OPE. Finally, COptiDICE (blue) computes a policy that is more robust to cost violation than other algorithms, while significantly outperforming BC in terms of reward performance. We observe that the hyperparameter $\epsilon$ in (16) controls the degree of cost-conservativeness as expected: larger values of $\epsilon$ lead to overestimates of the cost value, yielding a more conservative policy.

To study the dependence on the characteristics of the dataset we experiment with different values of $\beta$. Figures 3a-b show the result for $\beta = 0.8$ (low-reward-low-cost data), Figures 3c-d for $\beta = 0.5$, and Figures 3e-f for $\beta = 0.2$ (high-reward-high-cost data). These results show the expected trend: more high-reward-high-cost data leads to a joint increase in rewards and costs of all agents. The vanilla offline constrained RL algorithm (green), encouraged to stay close to the data, also suffers from severe constraint violation when most of the trajectories are given as the constraint-violating ones, which is similar to the result of Figure 1c-1d. Finally, COptiDICE (blue) demonstrates more robust behavior to avoid constraint violations across dataset configurations, highlighting the effectiveness of our method constraining the upper bound of the cost value.

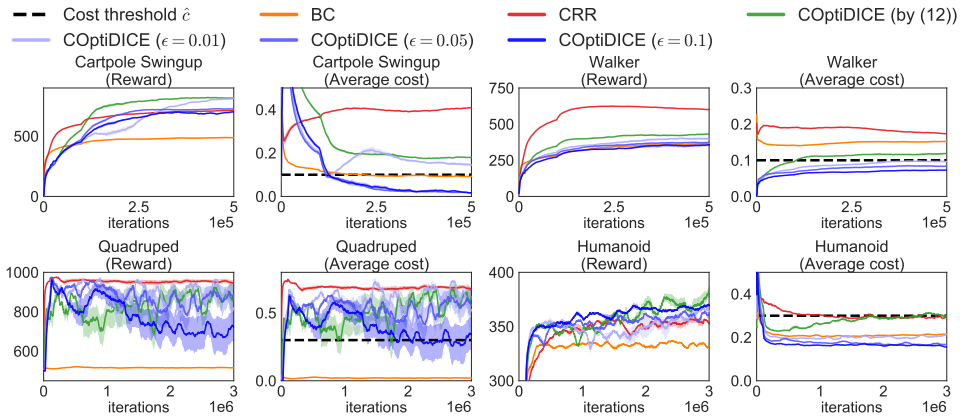


Figure 2: Result of RWRL control tasks. For each task, we report the reward return and the average cost. The results are averaged over 5 runs, and the shaded area represents the standard error.

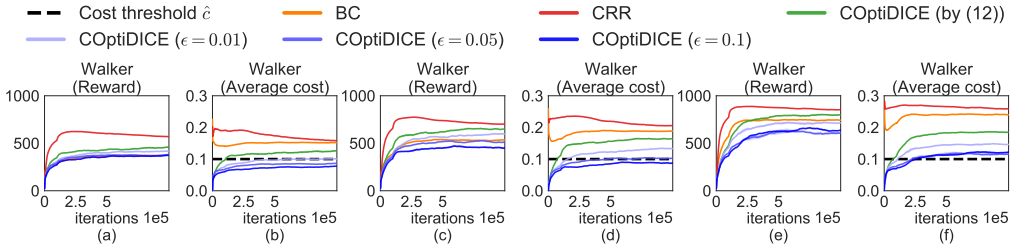


Figure 3: Result on RWRL walker using three dataset configurations with different levels of constraint satisfaction $\beta$: for (a)-(b) the data is obtained with $\beta = 0.8$ (low-reward-low-cost data), (c)-(d) with $\beta = 0.5$, and (e)-(f) with $\beta = 0.2$ (high-reward-high-cost data).

## 5 Discussion and Conclusion

The notion of safety in RL has been captured in various forms such as risk-sensitivity (Chow et al., 2015; Urpí et al., 2021; Yang et al., 2021), Robust MDP (Iyengar, 2005; Tamar et al., 2014), and Constrained MDP (Altman, 1999), among which we focus on CMDP as it provides a natural formalism to encode safety specifications (Ray et al., 2019). Most of the existing constrained RL algorithms (Achiam et al., 2017; Tessler et al., 2019; Satija et al., 2020) are on-policy algorithms, which cannot be applied to the offline setting directly. A recent exception is the work by Le et al. (2019) that also aims to solve constrained RL in an offline setting, though its method is limited to discrete action spaces and relies on solving an MDP completely as an inner optimization, which is inefficient. It also relies on the vanilla OPE estimate of the policy cost, which could result in severe constraint violation when deployed. Lastly, in a work done concurrently to ours, Xu et al. (2021) also exploits overestimated cost value to deal with the cost constraint, but its hyperparameter that controls overestimation is subtle for scaling of the reward and the cost; thus it might be hard to tune for various tasks in practice.

In this work, we have presented a DICE-based offline constrained RL algorithm, COptiDICE. DICE-family algorithms have been proposed for off-policy evaluation (Nachum et al., 2019a; Zhang et al., 2020a;b; Yang et al., 2020b; Dai et al., 2020), imitation learning (Kostrikov et al., 2019), offline policy selection (Yang et al., 2020a), and RL (Nachum et al., 2019b; Lee et al., 2021), but none of them is for *constrained* RL. Our first contribution was a derivation that constrained offline RL can be tackled by solving a single minimization problem. We demonstrated that such approach, in its simplest form, suffers from constraint violation in practice. To mitigate the issue, COptiDICE instead constrains the cost upper bound, which is estimated in a way that exploits the distribution correction $w$ obtained by solving the RL problem. Such reuse of $w$ eliminates the nested optimization in CoinDICE (Dai et al., 2020), and COptiDICE can be optimized efficiently as a result. Experimental results demonstrated that our algorithm achieved better trade-off between reward maximization and constraint satisfaction than several baselines, across domains and conditions.

## REFERENCES


Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Remi Munos, Nicolas Heess, and Martin Riedmiller. Maximum a posteriori policy optimisation. In *International Conference on Learning Representations (ICLR)*, 2018.

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 22–31. PMLR, 06–11 Aug 2017.

Eitan Altman. *Constrained Markov Decision Processes*. Chapman and Hall, 1999.

Christopher M. Bishop. Mixture density networks. Technical report, 1994.

V.S. Borkar. An actor-critic algorithm for constrained markov decision processes. *Systems & Control Letters*, 54(3):207–213, 2005.

Yinlam Chow, Aviv Tamar, Shie Mannor, and Marco Pavone. Risk-sensitive and robust decision-making: a cvar optimization approach. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 28, 2015.

Bo Dai, Ofir Nachum, Yinlam Chow, Lihong Li, Csaba Szepesvari, and Dale Schuurmans. CoinDICE: Off-policy confidence interval estimation. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 9398–9411, 2020.

Gabriel Dulac-Arnold, Nir Levine, Daniel J. Mankowitz, Jerry Li, Cosmin Paduraru, Sven Gowal, and Todd Hester. An empirical investigation of the challenges of real-world reinforcement learning. 2020.

Bradley Efron and Robert J. Tibshirani. *An Introduction to the Bootstrap*. Number 57 in Monographs on Statistics and Applied Probability. Chapman & Hall/CRC, Boca Raton, Florida, USA, 1993.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2019.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of Wasserstein GANs. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018.

Josiah P. Hanna, Peter Stone, and Scott Niekum. Bootstrapping with models: Confidence intervals for off-policy evaluation. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, AAMAS*, pp. 538–546. ACM, 2017.

Matt Hoffman, Bobak Shahriari, John Aslanides, Gabriel Barth-Maron, Feryal Behbahani, Tamara Norman, Abbas Abdolmaleki, Albin Cassirer, Fan Yang, Kate Baumli, Sarah Henderson, Alex Novikov, Sergio Gómez Colmenarejo, Serkan Cabi, Caglar Gulcehre, Tom Le Paine, Andrew Cowie, Ziyu Wang, Bilal Piot, and Nando de Freitas. Acme: A research framework for distributed reinforcement learning. *arXiv preprint arXiv:2006.00979*, 2020. URL https://arxiv.org/abs/2006.00979.

Garud N. Iyengar. Robust dynamic programming. *Mathematics of Operations Research*, 2005.

Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind Picard. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog, 2019.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. MOReL : Model-based offline reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Ilya Kostrikov, Ofir Nachum, and Jonathan Tompson. Imitation learning via off-policy distribution matching. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*, 2019.

Ilya Kostrikov, Rob Fergus, Jonathan Tompson, and Ofir Nachum. Offline reinforcement learning with fisher divergence critic regularization. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 5774–5783. PMLR, 18–24 Jul 2021.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy Q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Sascha Lange, Thomas Gabel, and Martin Riedmiller. *Reinforcement learning: State-of-the-art*. Springer Berlin Heidelberg, 2012.

Romain Laroche, Paul Trichelair, and Remi Tachet Des Combes. Safe policy improvement with baseline bootstrapping. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2019.

Hoang Le, Cameron Voloshin, and Yisong Yue. Batch policy learning under constraints. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3703–3712. PMLR, 09–15 Jun 2019.

Byungjun Lee, Jongmin Lee, Peter Vrancx, Dongho Kim, and Kee-Eung Kim. Batch reinforcement learning with hyperparameter gradients. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5725–5735. PMLR, 13–18 Jul 2020.

Jongmin Lee, Wonseok Jeon, Byungjun Lee, Joelle Pineau, and Kee-Eung Kim. OptiDICE: Offline policy optimization via stationary distribution correction estimation. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 6120–6130. PMLR, 18–24 Jul 2021.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems, 2020.

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

Daniel J. Mankowitz, Dan A. Calian, Rae Jeong, Cosmin Paduraru, Nicolas Heess, Sumanth Dathathri, Martin Riedmiller, and Timothy Mann. Robust constrained reinforcement learning for continuous control with model misspecification, 2021.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

Ofir Nachum, Yinlam Chow, Bo Dai, and Lihong Li. DualDICE: Behavior-agnostic estimation of discounted stationary distribution corrections. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019a.

Ofir Nachum, Bo Dai, Ilya Kostrikov, Yinlam Chow, Lihong Li, and Dale Schuurmans. AlgaeDICE: Policy gradient from arbitrary experience. *arXiv preprint arXiv:1912.02074*, 2019b.

Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking Safe Exploration in Deep Reinforcement Learning. 2019.

Harsh Satija, Philip Amortila, and Joelle Pineau. Constrained Markov decision processes via backward value functions. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pp. 8502–8511, 13–18 Jul 2020.

Noah Y. Siegel, Jost Tobias Springenberg, Felix Berkenkamp, Abbas Abdolmaleki, Michael Neunert, Thomas Lampe, Roland Hafner, Nicolas Heess, and Martin Riedmiller. Keep doing what worked: Behavioral modelling priors for offline reinforcement learning, 2020.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of Go without human knowledge. *Nature*, 550:354–359, 2017.

Aviv Tamar, Shie Mannor, and Huan Xu. Scaling up robust mdps using function approximation. In *Proceedings of the 31st International Conference on International Conference on Machine Learning (ICML)*, 2014.

Chen Tessler, Daniel J. Mankowitz, and Shie Mannor. Reward constrained policy optimization. In *International Conference on Learning Representations (ICLR)*, 2019.

Núria Armengol Urpí, Sebastian Curi, and Andreas Krause. Risk-averse offline reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2021.

Ziyu Wang, Alexander Novikov, Konrad Zolna, Josh S Merel, Jost Tobias Springenberg, Scott E Reed, Bobak Shahriari, Noah Siegel, Caglar Gulcehre, Nicolas Heess, and Nando de Freitas. Critic regularized regression. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 7768–7778, 2020.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning, 2019.

Haoran Xu, Xianyuan Zhan, and Xiangyu Zhu. Constraints penalized q-learning for safe offline reinforcement learning, 2021.

Mengjiao Yang, Bo Dai, Ofir Nachum, George Tucker, and Dale Schuurmans. Offline policy selection under uncertainty, 2020a.

Mengjiao Yang, Ofir Nachum, Bo Dai, Lihong Li, and Dale Schuurmans. Off-policy evaluation via the regularized lagrangian. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020b.

Qisong Yang, Thiago D. Simão, Simon H Tindemans, and Matthijs T. J. Spaan. Wcsac: Worst-case soft actor critic for safety-constrained reinforcement learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12):10639–10646, May 2021.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. MOPO: Model-based offline policy optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Ruiyi Zhang, Bo Dai, Lihong Li, and Dale Schuurmans. GenDICE: Generalized offline estimation of stationary values. In *Proceedings of the 8th International Conference on Learning Representations (ICLR)*, 2020a.

Shangtong Zhang, Bo Liu, and Shimon Whiteson. GradientDICE: Rethinking generalized offline estimation of stationary values. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2020b.

# A ALGORITHM FOR UNDISCOUNTED CMDP

For $\gamma = 1$, the optimization problem (4-7) should be modified by adding an additional normalization constraint $\sum_{s,a} d(s,a) = 1$.

$$\max_d \mathbb{E}_{(s,a)\sim d}[R(s,a)] - \alpha D_f(d||d^D) \tag{28}$$

$$\text{s.t. } \mathbb{E}_{(s,a)\sim d}[C_k(s,a)] \leq \hat{c}_k \qquad \forall k = 1,\dots,K \tag{29}$$

$$\sum_{a'} d(s',a') = (1-\gamma)p_0(s') + \gamma \sum_{s,a} d(s,a)T(s'|s,a) \qquad \forall s' \tag{30}$$

$$d(s,a) \geq 0 \qquad \forall s,a \tag{31}$$

$$\sum_{s,a} d(s,a) = 1 \tag{32}$$

Then, we consider the Lagrangian:

$$\begin{aligned}\min_{\lambda\geq 0,\nu} \max_{d\geq 0} &\mathbb{E}_{(s,a)\sim d}[R(s,a)] - \alpha D_f(d||d^D) - \sum_{k=1}^{K} \lambda_k \big(\mathbb{E}_{(s,a)\sim d}[C_k(s,a)] - \hat{c}_k\big) \\ &- \sum_{s'} \nu(s') \Big[\sum_{a'} d(s',a') - (1-\gamma)p_0(s') - \gamma \sum_{s,a} d(s,a)T(s'|s,a)\Big] - \mu\Big[\sum_{s,a} d(s,a) - 1\Big]\end{aligned} \tag{33}$$

where $\mu \in \mathbb{R}$ is the Lagrange multiplier for the normalization constraint (32). Then, we rearrange the terms so that the direct dependence on $d$ is eliminated, while introducing new optimization variables $w(s,a) = \frac{d(s,a)}{d^D(s,a)}$ that represent stationary distribution corrections:

$$\begin{aligned}&\min_{\lambda\geq 0,\nu,\mu} \max_{d\geq 0} \mathbb{E}_{\substack{(s,a)\sim d \\ s'\sim T(s,a)}} \big[R(s,a) - \lambda^\top C(s,a) + \nu(s') - \nu(s) - \mu\big] - \alpha \mathbb{E}_{(s,a)\sim d^D}\Big[f\Big(\tfrac{d(s,a)}{d^D(s,a)}\Big)\Big] \\ &\qquad + (1-\gamma)\mathbb{E}_{s_0\sim p_0}[\nu(s_0)] + \lambda^\top \hat{c} + \mu \\ &= \min_{\lambda\geq 0,\nu,\mu} \max_{w\geq 0} \mathbb{E}_{(s,a)\sim d^D}\big[w(s,a)(e_{\lambda,\nu}(s,a) - \mu) - \alpha f(w(s,a))\big] + (1-\gamma)\mathbb{E}_{s_0\sim p_0}[\nu(s_0)] + \lambda^\top \hat{c} + \mu\end{aligned} \tag{34}$$

which yields the minmax optimization problem that can be optimized in a fully offline manner. Due to the strict convexity of $f$, we can also derive the closed form solution of the inner maximization problem:

$$w^*_{\lambda,\nu,\mu}(s,a) = (f')^{-1}\Big(\tfrac{1}{\alpha}(e_{\lambda,\nu}(s,a) - \mu)\Big)_+ \tag{35}$$

which simplify the the minmax problem (34) into the following single minimization problem:

$$\begin{aligned}\min_{\lambda\geq 0,\nu,\mu} L(\lambda,\nu,\mu) = &\mathbb{E}_{(s,a)\sim d^D}\big[w^*_{\lambda,\nu,\mu}(s,a)(e_{\lambda,\nu}(s,a) - \mu) - \alpha f(w^*_{\lambda,\nu,\mu}(s,a))\big] \\ &+ (1-\gamma)\mathbb{E}_{s_0\sim p_0}[\nu(s_0)] + \lambda^\top \hat{c} + \mu\end{aligned} \tag{36}$$

Once the optimal solution $(\lambda^*, \nu^*, \mu^*)$ are obtained, we can also compute the stationary distribution corrections of the optimal policy using (35). Still, naively using (36) would similarly result in cost violation when the resulting policy is deployed to the real environment. We therefore adopt constraining the upper bound of cost value:

$$\nu, \mu \leftarrow \arg\min_{\nu,\mu} L(\lambda,\nu,\mu) \qquad \text{(RL for } R - \lambda^\top C \text{ using (36))} \tag{37}$$

$$\tau, \chi \leftarrow \arg\min_{\tau\geq 0,\chi} \textstyle\sum_{k=1}^{K} \ell_k(\tau_k, \chi_k; w^*_{\lambda,\nu,\mu}) \qquad \text{(Upper cost value estimation using (18))}$$

$$\lambda \leftarrow \arg\min_{\lambda\geq 0} \lambda^\top \big(\hat{c} - \underbrace{\ell(\tau,\chi;w^*_{\lambda,\nu,\mu})}_{\approx \text{UpperBound}(\hat{V}_C(\pi))}\big) \qquad \text{(Cost Lagrange multiplier)}$$

which completes the brief description of COptiDICE for $\gamma = 1$.

## B PROOFS

**Proposition 1.** *For any $\nu$ and $\lambda$, the closed-form solution of*

$$\max_{w\geq 0} L(w,\lambda,\nu) := \mathbb{E}_{(s,a)\sim d^D}\left[w(s,a)e_{\lambda,\nu}(s,a) - \alpha f(w(s,a))\right] + (1-\gamma)\mathbb{E}_{s_0\sim p_0}[\nu(s_0)] + \lambda^\top \hat{c} \tag{38}$$

*is given by:*

$$w^*_{\lambda,\nu}(s,a) = (f')^{-1}\left(\tfrac{1}{\alpha}e_{\lambda,\nu}(s,a)\right)_+ \textit{ where } x_+ = \max(0,x) \tag{39}$$

*Proof.* For a fixed $\lambda$ and $\nu$, we write the dual of the maximization problem $\max_{w\geq 0} L(w,\lambda,\nu)$:

$$\max_{w}\min_{\kappa\geq 0} L(w,\lambda,\nu) + \sum_{s,a}\kappa(s,a)w(s,a).$$

By the strong duality, it is sufficient to consider KKT condition for $(w^*,\kappa^*)$.

*Condition 1 (Primal feasibility)* $w^*(s,a)\geq 0\ \forall s,a$

*Condition 2 (Dual feasibility)* $\kappa^*(s,a)\geq 0\ \forall s,a$

*Condition 3 (Stationarity)* $d^D(s,a)(e_{\lambda,\nu}(s,a) - \alpha f'(w^*(s,a)) + \kappa^*(s,a)) = 0\ \forall s,a$

*Condition 4 (Complementary slackness)* $w^*(s,a)\kappa^*(s,a) = 0\ \forall s,a$

By *Stationarity* and $d^D > 0$,

$$\begin{aligned} & f'(w^*(s,a)) = \tfrac{1}{\alpha}(e_{\lambda,\nu}(s,a) + \kappa^*(s,a))\ \forall s,a \\ \Leftrightarrow\ & w^*(s,a) = (f')^{-1}\left(\tfrac{1}{\alpha}(e_{\lambda,\nu}(s,a) + \kappa^*(s,a))\right)\ \forall s,a. \end{aligned}$$

For each $(s,a)$, consider either $w^*(s,a) > 0$ or $w^*(s,a) = 0$.

(When $w^*(s,a) > 0$): $\kappa^*(s,a) = 0$ holds by *Complementary slackness*, which means:

$$w^*(s,a) = (f')^{-1}\left(\tfrac{1}{\alpha}e_{\lambda,\nu}(s,a)\right) > 0.$$

Note that *Dual feasibility* holds. Since $f'$ is a strictly increasing function, $e_\nu(s,a) > \alpha f'(0)$.

(When $w^*(s,a) = 0$): By *Stationarity* and *Dual feasibility*,

$$\begin{aligned} & \kappa^*(s,a) = \alpha f'(0) - e_{\lambda,\nu}(s,a) \geq 0 \\ \Leftrightarrow\ & e_\nu(s,a) \leq \alpha f'(0) \end{aligned}$$

This concludes $w^*_\nu(s,a) = (f')^{-1}\left(\tfrac{1}{\alpha}e_{\lambda,\nu}(s,a)\right)_+$. □

**Proposition 2.** *The constrained optimization problem (15-17) can be reduced to solving the following unconstrained minimization problem:*

$$\min_{\tau\geq 0,\chi}\ \ell_k(\tau,\chi;w) = \tau\log\mathbb{E}_{x\sim d^D}\left[\exp\left(\tfrac{1}{\tau}\left(w(s,a)(C_k(s,a) + \gamma\chi(s') - \chi(s)) + (1-\gamma)\chi(s_0)\right)\right)\right] + \tau\epsilon \tag{40}$$

*where $\tau\in\mathbb{R}_+$ corresponds to the Lagrange multiplier for the constraint* (16)*, and $\chi(s)\in\mathbb{R}$ corresponds to the Lagrange multiplier for the constraint* (17)*. In other words, $\min_{\tau\geq 0,\chi}\ell(\tau,\chi) = \mathbb{E}_{(s,a)\sim\tilde{p}^*}[w(s,a)C_k(s,a)]$ where $\tilde{p}^*$ is the optimal perturbed distribution of the problem (15-17). Also, for the optimal solution $(\tau^*,\chi^*)$, $\tilde{p}^*$ is given by:*

$$\tilde{p}^*(x) \propto d^D(x)\underbrace{\exp\left(\tfrac{1}{\tau^*}\left(w(s,a)(C_k(s,a) + \gamma\chi^*(s') - \chi^*(s)) + (1-\gamma)\chi^*(s_0)\right)\right)}_{=:\ \omega^*(x)\ \textit{(unnormalized weight for } x=(s_0,s,a,s'))} \tag{41}$$

*Proof.* For the given constrained optimization problem:

$$\max_{\tilde{p}\in\Delta(X)} \mathbb{E}_{(s_0,s,a,s')\sim\tilde{p}}[w(s,a)C_k(s,a)] \tag{42}$$

$$\text{s.t. } D_{\mathrm{KL}}(\tilde{p}(s_0,s,a,s')||d^D(s_0,s,a,s')) \leq \epsilon \tag{43}$$

$$\sum_{a'} \tilde{p}(s',a')w(s',a') = (1-\gamma)\tilde{p}_0(s') + \gamma \sum_{s,a} \tilde{p}(s,a)w(s,a)\tilde{p}(s'|s,a) \quad \forall s' \tag{44}$$

We consider the Lagrangian:

$$\begin{aligned}\max_{\tau\geq 0,\chi,\zeta} \max_{\tilde{p}\geq 0} &\sum_x \tilde{p}(s_0,s,a,s')[w(s,a)C_k(s,a)] - \tau\Big(\sum_x \tilde{p}(s_0,s,a,s')\Big[\log \tfrac{\tilde{p}(s_0,s,a,s')}{d^D(s_0,s,a,s')}\Big] - \epsilon\Big) \\ &- \sum_{s'} \chi(s')\Big[\sum_{a'} \tilde{p}(s',a')w(s',a') - (1-\gamma)\tilde{p}_0(s') - \gamma\sum_{s,a}\tilde{p}(s,a)w(s,a)\tilde{p}(s'|s,a)\Big] \\ &- \zeta\Big[\sum_x \tilde{p}(s_0,s,a,s') - 1\Big]\end{aligned} \tag{45}$$

where $\tau \in \mathbb{R}_+$ is the Lagrange multiplier for KL constraint (43), $\chi(s') \in \mathbb{R}$ is the Lagrange multiplier for (44), and $\zeta \in \mathbb{R}$ is the Lagrange multiplier for the normalization constraint that ensures $\sum_x \tilde{p}(s_0,s,a,s') = 1$. Then, we rearrange (45) by:

$$\begin{aligned}\max_{\tau\geq 0,\chi,\zeta} \max_{\tilde{p}\geq 0} g(\tau,\chi,\tilde{p}) = &\sum_x \tilde{p}(x)\big[w(s,a)\big(C_k(s,a) + \gamma\chi(s') - \chi(s)\big) + (1-\gamma)\chi(s_0) - \tau\log\tfrac{\tilde{p}(x)}{d^D(x)}\big] \\ &+ \tau\epsilon - \zeta\Big[\sum_x \tilde{p}(x) - 1\Big]\end{aligned} \tag{46}$$

Then, we can compute the closed form solution for the inner-maximization problem. Consider $\frac{\partial g(\tau,\chi,\tilde{p})}{\partial \tilde{p}(x)} = 0$ should hold for each $x$. Then,

$$\frac{\partial g(\tau,\chi,\tilde{p})}{\partial \tilde{p}(x)} = w(s,a)\big(C_k(s,a) + \gamma\chi(s') - \chi(s)\big) + (1-\gamma)\chi(s_0) - \tau\log\tfrac{\tilde{p}(x)}{d^D(x)} + \tau - \zeta = 0 \tag{47}$$

$$\Rightarrow \tilde{p}(x) \propto d^D(x)\exp\Big(\tfrac{1}{\tau}\big(w(s,a)\big(C_k(s,a) + \gamma\chi(s') - \chi(s)\big) + (1-\gamma)\chi(s_0)\big)\Big) \tag{48}$$

with some normalization constant described in terms of $\zeta$. Then, by plugging (48) into (46) with eliminating $\zeta$, we obtain the result. (41) is also by direct application of (48).

□

## C PSEUDOCODE OF COPTIDICE

**Algorithm 1** COptiDICE

**Input:** An offline dataset $D = \{(s_0,s,a,r,c,s')_i\}_{i=1}^N$, a learning rate $\eta$.

Initialize parameter vectors $\theta, \phi, \lambda, \tau, \psi$.
**for** each gradient step **do**
  Sample mini-batches from $D$.
  Compute gradients and perform SGD update:
    $\theta \leftarrow \theta - \eta\nabla_\theta J_\nu(\theta)$ (Eq. (23)) $\phi \leftarrow \phi - \eta\nabla_\phi J_{\tau,\chi}(\tau,\phi)$ (Eq. (25))
    $\tau \leftarrow \big[\tau - \eta\nabla_\tau J_{\tau,\chi}(\tau,\phi)\big]_+$ (Eq. (25)) $\lambda \leftarrow \big[\lambda - \eta J_\lambda(\lambda)\big]_+$ (Eq. (26))
    $\psi \leftarrow \psi - \eta\nabla_\psi J_\pi(\psi)$ (Eq. (27))
**end for**

# D EXPERIMENTAL SETTINGS

## D.1 RANDOM CMDPS

For random CMDP experiments, We follow a similar experimental protocol as (Laroche et al., 2019; Lee et al., 2021) with additional consideration of cost constraint.

**Random CMDP generation** For each run, we constructed a random CMDP with $|S| = 50, |A| = 4, \gamma = 0.95$, and a fixed initial state $s_0$. The transition probability is constructed randomly with connectivity of 4, i.e. for each $(s, a)$, we sample 4 states uniformly, and then, the transition probabilities to those states are determined by $\mathrm{Dirichlet}(1, 1, 1, 1)$. The reward of 1 is given to a single state that minimizes the optimal policy's reward value at $s_0$, and 0 is given anywhere else. This reward design can be roughly understood as choosing a goal state that is most difficult to reach from $s_0$. The cost function is generated randomly, i.e. $C(s, a) \sim \mathrm{Beta}(0.2, 0.2)$ for $\forall s \in S, a \in \{a_2, a_3, a_4\}$ and $C(s, a_1) = 0$ to ensure existence of a feasible policy of the CMDP. Lastly, the cost threshold $\hat{c} = 0.1$ is used.

**Data-collection policy generation** To construct the data-collection policy, we consider the following entropy-regularized CMDP:

$$\max_d \mathbb{E}_{(s,a)\sim d}[R(s, a)] - \sigma \mathbb{E}_{(s,a)}[\log d(s, a)] \tag{49}$$

$$\text{s.t. } \mathbb{E}_{(s,a)\sim d}[C(s, a)] \leq \hat{c}_D \tag{50}$$

where $\hat{c}_D \in \{0.09, 0.11\}$ is the hyperparameter that determines the cost value of $\pi_D$. Starting from $\sigma = 0$, we gradually increase $\sigma$ (i.e. adding stochasticity to policy, which adversarially affect the reward performance) until the reward performance of the computed policy reaches $0.9 \times V_R(\pi^*)$ where $\pi^*$ is the optimal policy of the given CMDP. After the data-collection policy $\pi_D$ is generated, we sample trajectories using $\pi_D$, which constitutes the offline dataset $D$. We conducted experiments for a varying number of trajectories, i.e. (the number of trajectories) $\in \{10, 20, 50, 100, 200, 500, 1000, 2000\}$. Each episode is terminated either when 50 time steps have reached or the agent reached the goal state that yields a non-zero reward.

**Hyperparameters** For C-SPIBB, we used $N_\wedge = 5$. For COptiDICE, we used $\alpha = \frac{1}{N}$ and $\epsilon = \frac{0.1}{N}$, where $N$ denotes the number of trajectories in $D$. We also used $f(x) = \frac{1}{2}(x - 1)^2$, which corresponds to $\chi^2$-divergence.

## D.2 RWRL CONTROL TASKS

**Network architecture and hyperparameters** We used the ACME framework (Hoffman et al., 2020). For the $\nu_\theta$ network and the $\chi_\phi$ network, we used `LayerNormMLP` with hidden sizes of $[512, 512, 256]$. For the policy network $\pi_\psi$, we used the network architecture used in CRR (Wang et al., 2020). Specifically, we the $\pi_\psi$ network consists of 4 `ResidualMLP` blocks with hidden size of 1024 and a mixture of Gaussians policy head with 5 mixture components. We used the batch size of 1024. We use Adam optimizer with learning rate 3e-4. Similar to ValueDICE (Kostrikov et al., 2021), we additionally adopt gradient penalties (Gulrajani et al., 2017) for the $\nu_\theta$ network and the $\chi_\phi$ network with coefficient 10e-5. We performed grid search for $\alpha \in \{0.01, 0.05, 0.1\}$ for each domain. We used the following $f$ as in OptiDICE (Lee et al., 2021):

$$f(x) = \begin{cases} x \log x - x + 1 & \text{if } 0 < x < 1 \\ \frac{1}{2}(x - 1)^2 & \text{if } x \geq 1 \end{cases}$$

**Task specification** We conduct experiments on domains using the RWRL suite with safety-spec. The *safety coefficient* is a flag in the RWRL suite with safety-spec, and its value can be between 0.0 and 1.0. Lowering the value of the flag incurs more safety-constraint violation (cost of 1). Originally, each domain has multiple types of safety constraints, but we use only one of them, which was the hardest safety constraint to be satisfied by an online constrained RL agent. In summary, we used the following task specifications (safety coefficients, name of the used safety constraint, cost threshold $\hat{c}$).

- Cartpole (realworld-swingup): safety-coeff=0.3, `slider_pos`, $\hat{c} = 0.1$,
- Walker (realworld-walk): safety-coeff=0.3, `joint_velocity_constraint` $\hat{c} = 0.1$.
- Quadruped (realworld-walk): safety-coeff=0.5, `joint_angle_constraint` $\hat{c} = 0.3$.
- Humanoid (realworld-walk): safety-coeff=0.5, `joint_angle_constraint`, $\hat{c} = 0.3$.

The offline dataset consists trajectories of 1000 episodes for Cartpole and Walker, and 5000 episodes for Quadruped and Humanoid.