Mirror Descent Policy Optimisation for Robust Constrained Markov Decision Processes

Anonymous authors
Paper under double-blind review

Abstract

Safety is an essential requirement for reinforcement learning systems. The newly emerging framework of robust constrained Markov decision processes allows learning policies that satisfy long-term constraints while providing guarantees under epistemic uncertainty. This paper presents mirror descent policy optimisation for robust constrained Markov decision processes (RCMDPs), making use of policy gradient techniques to optimise both the policy (as a maximiser) and the transition kernel (as an adversarial minimiser) on the Lagrangian representing a constrained MDP. In the oracle-based RCMDP setting, we obtain an $\mathcal{O}\left(\frac{1}{T}\right)$ convergence rate for the squared distance as a Bregman divergence, and an $\mathcal{O}\left(e^{-T}\right)$ convergence rate for entropy-regularised objectives. In the sample-based RCMDP setting, we obtain an $\tilde{\mathcal{O}}\left(\frac{1}{T^{1/3}}\right)$ convergence rate. Experiments confirm the benefits of mirror descent policy optimisation in constrained and unconstrained optimisation, and significant improvements are observed in robustness tests when compared to baseline policy optimisation algorithms.

1 Introduction

Reinforcement learning (RL) traditionally forms policies that maximise the total reward, within the Markov decision process (MDP) framework. Two important aspects, often overlooked by RL algorithms, are the safety of the policy, in terms of satisfying behavioral constraints, or the robustness of the policy to environment mismatches. In continuous control applications, for instance, there are often constraints on the spaces the agent may go and on the objects it may not touch, and there is the presence of the simulation-reality gap.

Safe RL techniques have been primarily proposed within the constrained Markov decision process (CMDP) formalism (Altman, 1999). In infinite MDPs, which are characterised by large or continuous state and action spaces, traditional solution techniques include linear programming and dynamic programming with Lagrangian relaxation. Following their empirical performance in high-dimensional control tasks (Lillicrap et al., 2015; Mnih et al., 2016), policy gradient algorithms have become a particularly popular option for safe RL in control applications (Achiam et al., 2017; Yang et al., 2020; Ding et al., 2020; Liu et al., 2021; Xu et al., 2021; Paternain et al., 2023; Ying et al., 2023; Wu et al., 2024). As summarised in Table 1, constrained policy gradient techniques often come with provable guarantees regarding global optimum convergence and constraint-satisfaction.

The robustness of mismatches in training and test transition dynamics has been the main subject of interest in the robust MDP (RMDP) framework (Iyengar, 2005; Nilim & Ghaoui, 2005), which formulates robust policies by training them on the worst-case dynamics model within a plausible set called the uncertainty set. The framework has led to a variety of theoretically sound policy gradient approaches (Ho et al., 2021; Zhang et al., 2021; Kumar et al., 2023; Kuang et al., 2022; Wang & Petrik, 2024; Li et al., 2023; Zhou et al., 2023; Wang et al., 2023). Recently, the approach has also branched out into the robust constrained MDP (RCMDP) framework (Russel et al., 2020; 2023; Wang et al., 2022; Bossens, 2024; Zhang et al., 2024; Sun et al., 2024), which provides the same robustness notion to CMDPs, where they provide worst-case guarantees on the performance, the constraints, or a combination thereof.

Recent work has shown the benefit of mirror descent policy optimisation techniques in theoretical guarantees. In robust MDPs, state-of-the-art results are obtained for mirror descent based policy gradient (Wang et al., 2023), showing an $\mathcal{O}(1/T)$ convergence rate in an oracle-based setting. In constrained MDPs, an $\mathcal{O}(\log(T)/T)$

Table 1: Overview of related work.

(a) Related CMDP policy optimisation algorithms

Algorithm	Description	Results
CPO (Achiam et al., 2017)	FIM-approximated trust region solved with convex dual program, line search	Monotonically non-decreasing value and constraint-satisfaction throughout the algorithm
RCPO (Tessler et al., 2019)	primal-dual with projection onto convex set (box constraints), three time-scale (actor,critic,multiplier)	convergence to feasible local optimum
PCPO (Yang et al., 2020)	FIM-approximated trust region with projection to feasible set	cf. CPO, and convergence to feasible local optimum
NPG-PD (Ding et al., 2020)	primal-dual, natural policy gradient ascent with projected subgradient descent on constraint-violation	global convergence with $\mathcal{O}(1/T^{1/2})$ for value and constraint-satisfaction (softmax); global convergence rate $\mathcal{O}(1/T^{1/2})$ for value and $\mathcal{O}(1/T^{1/4})$ for constraint-satisfaction (general)
CRPO (Xu et al., 2021)	value update in interior, constraint reduction in exterior	global convergence with $\mathcal{O}(1/T^{1/2})$ for value and constraint-satisfaction (softmax)
PMD-PD (Liu et al., 2021)	primal-dual, mirror descent, two time-scales (policy, multiplier)	global convergence rate $\mathcal{O}(\log(T)/T)$ for value and constraint-satisfaction (softmax)
Sample-based PMD-PD (Liu et al., 2021)	same as above in the sample-based setting	global convergence rate $\tilde{\mathcal{O}}(T^{-1/3})$ for value and constraint-satisfaction (softmax)

(b) Related RMDP and RCMDP policy optimisation algorithms.

Algorithm	Description	Results
Lyapunov-based RCPG (Russel et al., 2023)	primal-dual, linear programming for inner problem	convergence to local optimum Lyapunov stable policy, Lyapunov RCMDPs
RPD (Wang et al., 2022)	primal-dual, projected descent with robust TD for inner problem	local convergence rate $\mathcal{O}(1/T^{1/4})$, δ-contamination RCMDPs
RC-PO (Sun et al., 2024)	PCPO, projected gradient descent on parametrised dynamics for separate worst-case value and constraint inner problems	cf. CPO, (s,a)-rectangular RCMDPs
RMCPMD (Wang & Petrik, 2024)	primal-dual mirror descent approach for policy and transition dynamics	global convergence rate $\mathcal{O}(e^{-T})$ to robust value (direct parametrisation); global convergence rate $\mathcal{O}(\frac{1}{T})$ to robust value (softmax), s- and (s,a)-rectangular RMDPs
Robust PMD-PD (ours)	PMD-PD with Lagrangian transition mirror ascent	global convergence rate $\mathcal{O}(\frac{1}{T})$ to robust Lagrangian (softmax, squared distance as a Bregman divergence) and $\mathcal{O}(e^{-T})$ to entropy-regularised robust Lagrangian (softmax), s- and (s,a)-rectangular RCMDPs
Robust Sample-based PMD-PD (ours)	same as above in the sample-based setting	global convergence rate $\tilde{\mathcal{O}}(\frac{1}{T^{1/3}})$ for robust Lagrangian, value, and constraint-satisfaction (softmax, KL divergence as Bregman divergence), s- and (s,a)-rectangular RCMDPs)

convergence rate is obtained in the oracle-based setting and an $\tilde{\mathcal{O}}(T^{1/3})$ convergence rate in the sample-based setting.

In terms of coping with continuous state and action spaces, recent approaches include double sampling uncertainty sets, integral probability metrics uncertainty sets, and Wasserstein uncertainty sets (Zhou et al., 2023; Abdullah et al., 2019; Hou et al., 2020).

In this paper, we formulate a policy gradient algorithm for a robust constrained MDP setting, with the aim of providing formal guarantees on the convergence rate. We formulate policy gradient techniques to optimise both the policy (as a maximiser) and the transition kernel (as an adversarial minimiser) on the Lagrangian representing a constrained MDP. For a theoretical analysis, we contribute Robust (Sample-based) PMD, which extends the theory of Liu et al. (2021) and Wang et al. (2023) to the RCMDP setting. We primarily focus on the softmax parametrisation in the theoretical analysis. In the oracle-based RCMDP setting, we obtain an $\mathcal{O}(\frac{1}{T})$ convergence rate for the squared distance as a Bregman divergence, and $\mathcal{O}(e^{-T})$ for entropy-regularised objectives. In the sample-based RCMDP setting, we obtain an $\mathcal{O}(\frac{1}{T^{1/3}})$ convergence rate. For a practical implementation, we formulate MDPO-Robust-Lagrangian (MDPO-Robust-Lag for short), which introduces a robust Lagrangian to Mirror Descent Policy Optimisation (MDPO) (Tomar et al., 2022). We evaluate MDPO-Robust-Lagrangian empirically by comparing it to robust-constrained variants of PPO-Lagrangian (Ray et al., 2019) and RMCPMD (Wang et al., 2023), and find significant improvements in the penalised return in worst-case and average-case test performance on dynamics in the uncertainty set.

2 Preliminaries

The setting of the paper is based on robust constrained Markov Decision Processes (RCMDPs). Formally, an RCMDP is given by a tuple $(\mathcal{S}, \mathcal{A}, \rho, c_0, c_{1:m}, \mathcal{P}, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\rho \in \Delta(\mathcal{S})$ is the starting distribution, $c_0 : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the cost function, $c_i : \mathcal{S} \times \mathcal{A} \to \mathbb{R}, i \in [m]$ are constraint-cost functions, $\mathcal{P}$ is the uncertainty set, and $\gamma \in [0, 1)$ is a discount factor. In RCMDPs, at any time step $l = 0, \ldots, \infty$, an agent observes a state $s_l \in \mathcal{S}$, takes an action $a_l \in \mathcal{A}$ based on its policy $\pi \in \Pi = (\Delta(\mathcal{A}))^{\mathcal{S}}$, and receives cost $c_0(s_l, a_l)$ and constraint-cost signals $c_j(s_l, a_l)$ for all $j \in [m] = \{1, \ldots, m\}$. Following the action, the next state is sampled from a transition dynamics model $p \in \mathcal{P}$ according to $s_{l+1} \sim p(\cdot|s_l, a_l)$. The transition dynamics model p is chosen from the uncertainty set $\mathcal{P}$ to solve a minimax problem. In particular,

denoting

$$V^j_{\pi,p}(\rho) := \mathbb{E}_{s_0\sim\rho}\left[\sum_{l=0}^{\infty}\gamma^t c_j(s_l,a_l)|\pi,p\right] \tag{1}$$

and

$$V_{\pi,p}(\rho) := \mathbb{E}_{s_0\sim\rho}\left[\sum_{l=0}^{\infty}\gamma^t c_0(s_l,a_l)|\pi,p\right], \tag{2}$$

the goal of the agent is to optimise a minimax objective of the form

$$\min_{\pi}\left\{\Phi(\pi) := \sup_{p\in\mathcal{P}} V_{\pi,p}(\rho) \quad \text{s.t.} \quad V^j_{\pi,p}(\rho)\leq 0,\ \forall j\in[m]\right\}. \tag{3}$$

The objective has an equivalent unconstrained formulation according to the Lagrangian

$$\min_{\pi}\left\{\Phi(\pi) := \sup_{p\in\mathcal{P}}\max_{\lambda\geq 0} V_{\pi,p}(\rho) + \sum_{j=1}^{m}\lambda_j V^j_{\pi,p}(\rho)\right\}. \tag{4}$$

If the inner optimisation problem is solved as p_k at iteration k, this in turn corresponds to an equivalent value function over a traditional MDP (Mei et al., 2020; Bossens, 2024) according to the Lagrangian value function,

$$\mathbf{V}_{\pi,p_k}(s) = \mathbb{E}\left[\sum_{l=0}^{\infty}\gamma^l\left(c_0(s_l,a_l)+\sum_{j=1}^{m}\lambda_{k,j}c_j(s_l,a_l)\right)\Big|s_0=s,\pi,p_k,\right], \tag{5}$$

which has a single cost function defined by $\mathbf{c}_k(s,a) = c_0(s,a)+\sum_{j=1}^{m}\lambda_{k,j}c_j(s,a)$ and no constraint-costs. The multiplier λ_k denotes the vector of dual variables at iteration k and $\lambda_{k,j}$ denotes the multiplier for constraint $j\in[m]$ at iteration k. Occasionally, in proofs where the iteration is not needed, the iteration index will be omitted.

A convenient definition used in the following is the discounted state visitation distribution

$$d^{\pi_k,p_k}_{\rho}(s) = (1-\gamma)\mathbb{E}_{s_0\sim\rho}\left[\sum_{l=0}^{\infty}\gamma^l\mathbb{P}(s_l=s|s_0,\pi_k,p_k)\right], \tag{6}$$

and the discounted state-action visitation distribution

$$d^{\pi_k,p_k}_{\rho}(s,a) = (1-\gamma)\mathbb{E}_{s_0\sim\rho}\left[\sum_{l=0}^{\infty}\gamma^l\mathbb{P}(s_l=s,a_l=a|s_0,\pi_k,p_k)\right]. \tag{7}$$

To denote the cardinality of sets $\mathcal{S}$ and $\mathcal{A}$, we use the convention $S := |\mathcal{S}|$ and $A := |\mathcal{A}|$.

3 PMD-PD for Constrained MDPs

We briefly review the Policy Mirror Descent-Primal Dual (PMD-PD) algorithm (Liu et al., 2021) for constrained MDPs. PMD-PD uses natural policy gradient with parametrised softmax policies, which is equivalent to the mirror descent.

At iteration $k\in[K]$ and policy update $t\in[t_k]$ within the iteration, PMD-PD defines the regularised state value function,

$$\tilde{\mathbf{V}}^{\alpha}_{\pi_k^t,p}(s) = \mathbb{E}\left[\sum_{l=0}^{\infty}\gamma^l\left(\tilde{\mathbf{c}}_k(s_l,a_l)+\alpha\log\left(\frac{\pi_k^t(a_l|s_l)}{\pi_k(a_l|s_l)}\right)\right)\Big|s_0=s,\pi_k^t,p\right], \tag{8}$$

where p is a fixed transition kernel, $\tilde{\mathbf{c}}_k(s_l, a_l) = c_0(s_l, a_l) + \sum_{j=1}^{m} \left(\lambda_{k,j} + \eta_\lambda V^j_{\pi_k, p_k}\right) c_j(s_l, a_l)$ and $\alpha \geq 0$ is the regularisation coeffient. The regularisation implements a weighted Bregman divergence which is similar to the KL-divergence. The augmented Lagrangian multiplier $\tilde{\lambda}_{k,j} = \lambda_{k,j} + \eta_\lambda V^j_{\pi_k, p_k}$ is used to improve convergence results by providing a more smooth Lagrangian compared to techniques that clip the Lagrangian multiplier. At the end of each iteration $\lambda_{k,j}$ is updated according to $\lambda_{k,j} = \max\left\{\lambda_{k,j} + \eta_\lambda V^j_{\pi_{k+1}, p_{k+1}}, -\eta_\lambda V^j_{\pi_{k+1}, p_{k+1}}\right\}$.

Analogous to 8, the regularised state-action value function is given by

$$\tilde{\mathbf{Q}}^{\alpha}_{\pi^t_k, p}(s, a) = \tilde{\mathbf{c}}_k(s, a) + \alpha \log\left(\frac{1}{\pi_k(a|s)}\right) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)}\left[\mathbf{V}^{\alpha}_{\pi^t_k, p}(s')\right] . \tag{9}$$

While the bold indicates the summation over $1 + m$ terms (i.e. the costs and constraint-costs) as in Eq. 5, the tilde notations will be used throughout the text to indicate the use of the augmented Lagrangian multipliers $\tilde{\lambda}_k$. However, in Section 4.2 and 4.3, where we do not exploit the augmented property, we state the results generically so they can be applied to both traditional and augmented Lagrangians. Note that since all the costs and constraint-costs range in $[-1, 1]$,

$$|\tilde{\mathbf{c}}_k(s, a)| \leq 1 + \sum_{j=1}^{m} \lambda_{k,j} + \frac{m\eta_\lambda}{1 - \gamma} . \tag{10}$$

Consequently, for any set of multipliers $\lambda \in \mathbb{R}^m$, the Lagrangian is bounded by $[-F_\lambda/(1-\gamma), F_\lambda/(1-\gamma)$ based on the constant

$$F_\lambda := 1 + \sum_{j=1}^{m} \lambda_j + \frac{\eta_\lambda m}{(1 - \gamma)} , \tag{11}$$

for the augmented Lagrangian, and according to

$$F_\lambda := 1 + \sum_{j=1}^{m} \lambda_j , . \tag{12}$$

for the traditional Lagrangian.

Note that this can be equivalently written based on the weighted Bregman divergence (equivalent to a pseudo KL-divergence over occupancy distributions),

$$B_{d_\rho^{\pi^t_k, p}}(\pi^t_k, \pi_k) = \sum_{s \in \mathcal{S}} d_\rho^{\pi^t_k, p}(s) \sum_{a \in \mathcal{A}} \pi^t_k(a|s) \frac{\log(\pi^t_k(a|s))}{\log(\pi_k(a|s))} , \tag{13}$$

according to

$$\tilde{\mathbf{Q}}^{\alpha}_{\pi^t_k, p}(s, a) = \mathbf{Q}_{\pi^t_k}(s, a) + \frac{\alpha}{1 - \gamma} B_{d_\rho^{\pi^t_k}}(\pi^t_k, \pi_k) . \tag{14}$$

This way, PMD-PD applies entropy regularisation with respect to the previous policy as opposed to the uniformly randomized policy used in Cen et al. (2022), allowing PMD-PD to converge to the optimal unregularised policy as opposed to the optimal regularised (and sub-optimal unregularised) policy.

With softmax parametrisation, the policy is defined as

$$\pi_\theta(a|s) = \frac{e^{\theta_{s,a}}}{\sum_{a'} e^{\theta_{s,a'}}} . \tag{15}$$

Under this parametrisation, the mirror descent update over the unregularised augmented Lagrangian with the KL-divergence as the Bregman divergence can be formulated for each state independently according to

$$\begin{aligned}\pi_k^{t+1}(a|s) &= \arg\min_{\pi}\left\{\left\langle \frac{\partial \tilde{\mathbf{V}}_{\pi_k^t,p}(\rho)}{\partial\theta(s,\cdot)}, \pi(\cdot|s)\right\rangle + \frac{1}{\eta'} D_{\mathrm{KL}}(\pi(\cdot|s), \pi_k^t(\cdot|s))\right\} \\ &= \arg\min_{\pi}\left\{\frac{1}{1-\gamma} d_\rho^{\pi_k^t,p}(s)\left\langle \tilde{\mathbf{Q}}_{\pi_k^t,p}(s,\cdot), \pi(\cdot|s)\right\rangle + \frac{1}{\eta'} D_{\mathrm{KL}}(\pi(\cdot|s), \pi_k^t(\cdot|s))\right\} \\ &= \arg\min_{\pi}\left\{\left\langle \tilde{\mathbf{Q}}_{\pi_k^t,p}(s,\cdot), \pi(\cdot|s)\right\rangle + \frac{1}{\eta} D_{\mathrm{KL}}(\pi(\cdot|s), \pi_k(\cdot|s))\right\}, \end{aligned} \tag{16}$$

where the last step defines $\eta = \eta'(1-\gamma)/d_\rho^{\pi_k,p}(s)$ and subsequently removes the constant $\frac{1}{1-\gamma} d_\rho^{\pi_k^t,p}(s)$ from the minimisation problem. Treating $\tilde{\mathbf{Q}}_{\pi_k^t,p}$ as any other value function, Eq. 16 is shown to be equivalent to the natural policy gradient update, as discussed by works in traditional MDPs (Zhan et al., 2023; Cen et al., 2022). In particular, based on Lemma 6 in Cen et al. (2022), the update rule derives from the properties of the softmax parametrisation in Eq. 15,

$$\pi_k^{t+1}(a|s) = \frac{1}{Z_k^t(s)} (\pi_k^t(a|s))^{1-\frac{\eta\alpha}{1-\gamma}} e^{\frac{-\eta \tilde{\mathbf{Q}}^{\alpha}_{\pi_k^t,p}(s,a)}{1-\gamma}}, \tag{17}$$

where π_k^{t+1} is the $t+1$'th policy at the k'th iteration of the algorithm and $Z_k^t(s)$ is the normalisation constant.

Denoting $T = \sum_{k=1}^K t_k$, the oracle version (with perfect gradient info) converges at rate $\mathcal{O}(\log(T)/T)$ in value and constraint violation. PMP-PD Zero (oracle version aiming for zero constraint violation) achieves 0 violation with the same convergence rate for the value. Finally, a version with imperfect gradient information, the so-called sample-based version, was shown to converge at a rate $\tilde{\mathcal{O}}(1/T^{1/3})$, where T denotes the total number of samples, thereby improving on model-free results which have a convergence rate $\mathcal{O}(1/T^{1/4})$.

4 Robust Sample-based PMD-PD for Robust Constrained MDPs

Having reviewed the performance guarantees of PMD-PD, we now derive the Robust Sample-based PMD-PD algorithm for robust constrained MDPs. The algorithm performs robust training based on Transition Mirror Ascent (TMA) (Wang & Petrik, 2024) over the robust Lagrangian objective (Bossens, 2024). Using TMA provides a principled way to compute adversarial dynamics while maintaining the transition dynamics within the bounds of the uncertainty sets. Such uncertainty sets allow for rich parametrisation such as Gaussian mixtures, which can estimate rich probability distributions, and entropy-based parametrisations, which are motivated by the optimal form of KL-divergence based uncertainty sets (see Table 7 for these examples). The algorithm takes place in the sample-based setting, where the state(-action) values are approximated from a limited number of finite trajectories. With a regret bound $\tilde{\mathcal{O}}(\frac{1}{T^{1/3}})$ results are in line with the Sample-based PMD-PD algorithm. The resulting algorithm is given in Algorithm 1. While the first subsections develop theory for the oracle setting, where exact values and policy gradients are given, the subsequent subsection (Section 4.4) derives convergence results for the sample-based setting, where values and policy gradients are estimates. The last subsection then proposes a practical implementation.

4.1 General assumptions

Before listing the general assumptions in the theoretical analysis, we require the definition of rectangular uncertainty sets.

Definition 1 (Rectangularity)**.** *An uncertainty set is (s,a)-rectangular if it can be decomposed as $\mathcal{P} = \bigtimes_{(s,a)\in\mathcal{S}\times\mathcal{A}} \mathcal{P}_{s,a}$ where $\mathcal{P}_{s,a} \subseteq \Delta(\mathcal{S})$. Similarly, it is s-rectangular if it can be decomposed as $\mathcal{P} = \bigtimes_{s\in\mathcal{S}} \mathcal{P}_s$ where $\mathcal{P}_s \subseteq \Delta(\mathcal{S})$.*

As shown below, the assumptions of our analysis are common and straightforward.

Algorithm 1 Robust Sample-based PMD-PD (discrete setting)

Inputs: Discount factor $\gamma \in [0,1)$, error tolerance $\epsilon > 0$, error tolerance for LTMA $\epsilon_0' > 0$, failure probability $\delta \in [0,1)$, learning rate $\eta = \frac{1-\gamma}{\alpha}$, dual learning rate $\eta_\lambda = 1.0$, and penalty coefficients $\alpha = \frac{2\gamma^2 m \eta_\lambda}{(1-\gamma)^3}$ and $\alpha_p > 0$.

Initialise: π_0 as uniform random policy, $\lambda_{0,j} = \max\left\{0, -\eta_\lambda \hat{V}_{\pi_0,p_0}(\rho)\right\}$, p_0 as the nominal transition kernel $\bar{p}$.

for $k \in \{0,\dots,K-1\}$ **do**

  for $t \in \{0,\dots,t_k-1\}$ **do**

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

      Generate $M_{Q,k}$ samples of length $N_{Q,k}$ based on π_k^t and p_k (following Lemma 9).

      Estimate the value: e.g. for softmax parametrisation,

      $$\hat{\mathbf{Q}}^{\alpha}_{\pi_k^t,p_k}(s,a) = \tilde{\mathbf{c}}_k(s,a) + \alpha\log(\tfrac{1}{\pi_k(a|s)}) + \tfrac{1}{M_{Q,k}}\sum_{j=1}^{M_{Q,k}}\sum_{l=1}^{N_{V,k}-1}\gamma^l\left[\tilde{\mathbf{c}}_k(s_l^j,a_l^j) + \alpha\sum_{a'}\pi_k^t(a'|s_l^j)\log(\tfrac{\pi_k^t(a'|s_l^j)}{\pi_k(a'|s_l^j)})\right].$$

    end for

    ▷ Policy mirror descent

    if Using Softmax parametrisation **then**

      $$\pi_k^{t+1}(a|s) \leftarrow (\pi_k^t(a|s))^{1-\frac{\eta\alpha}{1-\gamma}}\exp\left(-\eta\frac{\hat{\mathbf{Q}}^{\alpha}_{\pi_k^t,p_k}(s,a)}{1-\gamma}\right) \quad \forall (s,a)\in\mathcal{S}\times\mathcal{A}.$$

    else

      ▷ Use general Bregman divergence

      $\theta_k^{t+1} \leftarrow \arg\min_{\theta\in\Theta}\eta\left\langle\nabla_\theta\tilde{\mathbf{Q}}_{\pi_\theta,p_k},\theta\right\rangle + \alpha B_{d_\rho^{\pi_\theta,p_k}}(\theta,\theta_k)$ ▷ separate value and divergence estimates

      Define $\pi_k^{t+1} := \pi_{\theta_k^{t+1}}$.

    end if

  end for

  Define $\pi_{k+1} := \pi_k^{t_k}$

  ▷ Update transition matrix via Lagrangian TMA (error tolerance fixed to ϵ_0' or $\epsilon_{k+1}' = \gamma\epsilon_k'$)

  for $t' \in \{0,\dots,t_k'-1\}$ **do**

    $\xi_k^{t'+1} \leftarrow \arg\max_{\xi\in\mathcal{U}}\eta_p\left\langle\nabla_\xi\tilde{\mathbf{V}}_{\pi_k,p_k^{t'}}(\rho),\xi\right\rangle - \alpha_p B_{d_\rho^{\pi,p_k^{t'}}}(\xi,\xi_k^{t'})$ ▷ Monte Carlo or other techniques

    Define $p_k^{t'+1} := p_{\xi_k^{t'+1}}$

  end for

  Define $p_{k+1} := p_k^{t_k'}$

  ▷ Augmented update of Lagrangian multipliers

  Generate $M_{V,k+1}$ samples of length $N_{V,k+1}$ based on π_{k+1} and p_{k+1} starting from $s_0\sim\rho$ (following Lemma 9).

  Estimate $\hat{V}^i_{\pi_{k+1},p_{k+1}}(\rho) = \frac{1}{M_{V,k+1}}\sum_{j=1}^{M_{Q,k+1}}\sum_{l=1}^{N_{V,k+1}}\gamma^l c_i(s_l^j,a_l^j)$ for all $i\in[m]$.

  $\lambda_{k+1,i} \leftarrow \max\left\{-\eta_\lambda\hat{V}_{\pi_{k+1},p_{k+1}}(\rho), \lambda_{k,i} + \eta_\lambda\hat{V}^i_{\pi_{k+1},p_{k+1}}(\rho)\right\} \quad \forall i\in[m]$

end for

Assumption 1 (Bounded costs and constraint-costs)**.** *Costs and constraint-costs are bounded in* $[-1, 1]$.

This assumption implies that the Lagrangian at any time introduces a factor $\mathcal{O}(m)$ compared to a traditional value function.

Assumption 2 (Sufficient exploration)**.** *The initial state distribution satisfies* $\mu(s) > 0$ *for all* $s \in \mathcal{S}$.

This assumption ensures that when sampling from some μ instead of the initial state distribution ρ, any importance sampling corrections with μ and d_μ^π are finite. In particular, it ensures that for any transition kernel p and any policy π that the mismatch coefficient $M_p(\pi) := \left\| \frac{d_\mu^{\pi,p}}{\mu} \right\|_\infty$ is finite. This is also useful for robust MDPs since $M := \sup_{p\in\mathcal{P},\pi\in\Pi} M_p(\pi)$ is also finite.

Assumption 3 (Slater's condition)**.** *There exists a slack variable* $\zeta > 0$ *such that for any* $p \in \mathcal{P}$*, there exists a policy* $\bar{\pi}$ *with* $V_{\bar{\pi},p}^j(\rho) \leq -\zeta$ *for all* $j \in [m]$.

Slater's condition implies strict feasibility of the constraints, zero duality gap, and complementary slackness. However, we do not require knowledge of ζ in our analysis. The condition is slightly more restrictive than in traditional CMDPs due to the requirement of the condition to hold for all $p \in \mathcal{P}$. However, it is a reasonable assumption since it should typically be assumed for primal-dual algorithms on the transition kernel of interest; the worst-case dynamic p^* would typically have a smaller slack anyway.

Assumption 4 (Convex and rectangular uncertainty set)**.** *The uncertainty set is s-rectangular or* (s, a)*-rectangular (Def. 1) and is convex, such that for any pair of transition kernel parameters* ξ, ξ'*, and any* $\alpha \in [0, 1]$*, if* $p_\xi, p_{\xi'} \in \mathcal{P}$ *then also* $p_{\alpha\xi+(1-\alpha)\xi'} \in \mathcal{P}$.

The convex set assumption ensures updates to the transition kernel will remain in the interior so long as it is on a line between two interior points, a common assumption for mirror descent which is also needed for TMA. This is satisifed for most common uncertainty sets, e.g. ℓ_1 and ℓ_2 sets centred around a nominal value. The use of rectangularity assumptions allows us to efficiently use performance difference lemmas for transition kernels as well as make use of results for the optimality of TMA.

Note that when using such parametrised transition kernels, it is convenient to formulate an uncertainty set in the parameter space, which we denote as $\mathcal{U}_\xi$. A few examples of parametrised transition kernels (PTKs) and their associated parametrised uncertainty sets can be found in Table 7, including the Entropy PTK (Wang & Petrik, 2024), which presents a parametrisation related to the optimal parametrisation for KL divergence uncertainty sets of the form $\mathcal{P}_{s,a} = \{p : D_{\mathrm{KL}}(p, \bar{p}(\cdot|s, a)) \leq \kappa\}$ (Nilim & Ghaoui, 2005), and the Gaussian mixture PTK (Wang & Petrik, 2024), which is able to capture a rich set of distributions for each state-action pair.

4.2 Global optimality of Lagrangian TMA

As shown below, Lagrangian TMA demonstrably solves the maximisation problem in the robust objective (see Eq. 4) by performing mirror ascent on a suitable PTK. Note that the analysis follows closely that of Wang & Petrik (2024), where the Lagrangian and the weighted Bregman divergence are explicitly separated in this subsection.

The first step of our analysis of Lagrangian TMA (LTMA) is to derive the gradient of the transition dynamics.

Lemma 1 (Lagrangian transition gradient theorem)**.** *The transition kernel has the gradient*

$$\nabla_\xi \mathbf{V}_{\pi,p}(\rho) = \frac{1}{1-\gamma} \mathbb{E}_{s\sim d_\rho^{\pi,p}}[\nabla_\xi \log p(s'|s, a)(\mathbf{c}(s, a) + \gamma \mathbf{V}_{\pi,p}(s'))|\pi, p]\,. \tag{18}$$

Proof. The result follows from Theorem 5.7 in Wang & Petrik (2024) by considering the Lagrangian as a value function (see Eq. 5). □

Note that the Lagrangian adversarial policy gradient in Bossens (2024) follows similar reasoning but there are two differences in the setting. First, due to the formulation of costs as $c_0(s, a, s')$ rather than $c_0(s, a)$,

the state-action value of every time step appears rather than the value of every next time step. Second, we show the proof for a discounted Lagrangian of a potentially infinite trajectory rather than an undiscounted T-step trajectory. Third, in the following, we apply it not only to the traditional Lagrangian but also to the augmented Lagrangian and the augmented regularised Lagrangian.

The second step of our analysis of LTMA to show that optimising the transition dynamics p_ξ while fixing the policy π and the Lagrangian multipliers yields a bounded regret w.r.t. the global optimum.

Lemma 2 (Regret of LTMA)**.** *Let $\epsilon' > 0$, let π be a fixed policy, let $\lambda \in \mathbb{R}^m$ be a fixed vector of multipliers for each constraint, let p_t be the transition kernel at iteration t, and let p_0 be the transition kernel at the start of LTMA. Moreover, let $p^* = \arg\min_{p\in\mathcal{P}} \mathbf{V}_{\pi,p}(\rho)$, let $M := \sup_{p\in\mathcal{P},\pi\in\Pi} M_p(\pi)$ be an upper bound on the mismatch coefficient, and let $\eta_p > 0$ be the learning rate and $\alpha_p > 0$ be the penalty parameter for LTMA. Then for any starting distribution $\rho \in \Delta(\mathcal{S})$ and any $s \in \mathcal{S}$, the regret of LTMA is given by*

$$\mathbf{V}_{\pi,p^*}(\rho) - \mathbf{V}_{\pi,p_t}(\rho) \leq \frac{2F_\lambda}{t}\left(\frac{M}{(1-\gamma)^2} + \frac{1}{\eta_p(1-\gamma)} B_{d_\rho^{\pi,p^*}}(p^*, p_0)\right), \tag{19}$$

where F_λ is defined according to Eq. 12 for the traditional Lagrangian and on Eq. 11 for the augmented Lagrangian. Moreover, if $\eta_p/\alpha_p \geq \frac{(1-\gamma)}{2F_\lambda} B_{d_\rho^{\pi,p^*}}(p_*, p_0)$ and $t \geq 2F_\lambda \frac{M+1}{\epsilon'(1-\gamma)^2}$, an ϵ'-precise transition kernel is found such that

$$\mathbf{V}_{\pi,p^*}(\rho) - \mathbf{V}_{\pi,p_t}(\rho) \leq \epsilon' .$$

Proof. Following Theorem 5.5 in Wang & Petrik (2024), it follows for any value function with cost in $[0, 1]$ that

$$V_{\pi,p^*}(\rho) - V_{\pi,p_t}(\rho) \leq \frac{1}{t}\left(\frac{M}{(1-\gamma)^2} + \frac{\alpha_p}{\eta_p(1-\gamma)} B_{d_\rho^{\pi,p^*}}(p^*, p_0)\right). \tag{20}$$

Reformulating the Lagrangian as a value function (see Eq. 5), considering the bounds on the absolute value of the constraint-cost Eq. 10, and observing that the bounds on the (un-)augmented Lagrangian lie in $[-F_\lambda, F_\lambda]$ (see Eq. 11 and 12), the result can be scaled to obtain

$$\mathbf{V}_{\pi,p^*}(\rho) - \mathbf{V}_{\pi,p_t}(\rho) \leq 2F_\lambda\left(\frac{M}{(1-\gamma)^2} + \frac{\alpha_p}{\eta_p(1-\gamma)} B_{d_\rho^{\pi,p^*}}(p^*, p_0)\right).$$

From the settings of $\eta_p/\alpha_p \geq \frac{(1-\gamma)}{2F_\lambda} B_{d_\rho^{\pi,p^*}}(p^*, p_0)$ and $t \geq 2F_\lambda \frac{M+1}{\epsilon'(1-\gamma)^2}$, we obtain

$$\begin{aligned}\mathbf{V}_{\pi,p^*}(\rho) - \mathbf{V}_{\pi,p_t}(\rho) &\leq \frac{2F_\lambda}{t}\left(\frac{M+1}{(1-\gamma)^2}\right) \\ &\leq \epsilon' .\end{aligned}$$

□

4.3 Global optimality of the Lagrangian policy update in RCMDPs

Since at each loop of TMA, it is possible to obtain arbitrary precision, we now turn to proving the robust objective regret upper bound for the policy π_θ in the softmax parametrisation. Our technique for RCMDPs is similar to the results for RMDPs (Wang & Petrik, 2024), as our proof is based on finding a correspondence between the robust objective and the nominal objective and then making use of traditional MDP results by Mei et al. (2020).

4.3.1 Lagrangian policy gradient

To extend the analysis of RMDPs and traditional MDPs to the constrained setting, we first develop a few key lemmas for Lagrangian-based policy gradient. These results will then be used for analysing the convergence of RCMDPs under squared error and KL-divergence based Bregman divergences.

Lemma 3 (Policy gradient for softmax parametrisation)**.** *Let π be a policy parametrised according to*

$$\pi_\theta(a|s) = \frac{\exp(\theta_{s,a})}{\sum_{a' \in \mathcal{A}} \theta_{s,a'}}$$

and let $\mathbf{A}_{\pi_\theta}(s,a) = \mathbf{Q}_{\pi_\theta}(s,a) - \mathbf{V}_{\pi_\theta}(s)$ be the advantage function. Then the Lagrangian policy gradient is given by

$$\frac{\partial \mathbf{V}_{\pi_\theta}(\rho)}{\partial \theta_{s,a}} = \frac{1}{1-\gamma} d_\rho^{\pi,p}(s) \pi_\theta(a|s) \mathbf{A}_{\pi_\theta}(s,a)\,. \tag{21}$$

Proof. The proof follows from Eq. 10 in Agarwal et al. (2021) and by considering the Lagrangian as a value function. □

4.3.2 Regret of mirror descent policy optimisation with robust Lagrangian

When using the squared error as a distance-generating function, the mirror descent update is equivalent to traditional gradient descent (see Table 5). This can be exploited to derive results for RCMDPs. In particular, using smoothness and continuity properties, an equivalent gradient descent sequence can be derived to provide guarantees based on traditional gradient descent over a traditional MDP.

We first restate the supporting lemma from Wang & Petrik (2024).

Lemma 4 (Existence of equivalent gradient descent sequence, Lemma B.5 in Wang & Petrik (2024))**.** *For any $t' \geq 0$, any $\theta \in \mathbb{R}^d$, and any optimisation problem $\min_{\theta \in \mathbb{R}} f(\theta)$ in which $f : \mathbb{R}^d \to \mathbb{R}$ is L_θ-Lipschitz and l_θ-smooth, there exists a sequence $\{\theta_t\}_{t \geq 0}$, generated by the gradient descent update with constant learning rate $\eta > 0$*

$$\theta_{t+1} = \theta_t - \eta \nabla f(\theta_t)\,, \tag{22}$$

such that $\theta_{t'} = \theta$.

Without entropy regularisation Using Lemma 4, we derive an RCMDP analogue to Theorem 5.5 in Wang & Petrik (2024). It can be applied directly to softmax parametrisations with squared error Bregman divergence and indirectly to the KL-divergence Bregman divergence as shown later in Section 4.4. We first state three important properties that are essential to the proof of the convergence rate.

First, the Lagrangian function has essential smoothness and continuity properties.

Lemma 5 (Smooth and continuous Lagrangian)**.** *Under the softmax parametrisation, the Lagrangian $\mathbf{V}(\theta)$ with multipliers $\lambda \in \mathbb{R}^m$ is l_θ-smooth and L_θ-Lipschitz continuous with*

$$l_\theta(\lambda) = \frac{16 F_\lambda}{(1-\gamma)^3} \tag{23}$$

and

$$L_\theta(\lambda) = \frac{2\sqrt{2} F_\lambda}{(1-\gamma)^2}\,. \tag{24}$$

Proof. From Lemma 4.4 in Wang & Petrik (2024), the value function based on costs in $[0,1]$ has $L_\theta = \frac{\sqrt{2}}{(1-\gamma)^2}$ and $l_\theta = \frac{8}{(1-\gamma)^3}$. Since the Lagrangian is based on costs and constraint-costs in $[-1,1]$, the result follows after using the normalisation by $2F_\lambda$ (see Eq. 11 and 12. □

Second, we note that under a fixed set of Lagrangian multipliers, standard convergence rate results for softmax policies can be used.

Lemma 6 (Convergence rate for softmax policies for Lagrangian gradient descent)**.** *For Lagrangian gradient descent over logits, and Lemma 5, with settings of $M = \inf_{p \in \mathcal{P}, \pi \in \Pi} M_p(\pi)$, $U = \inf_{p \in \mathcal{P}} U_p$, fixed $\lambda_k = \lambda_{k-1} = \lambda$ for all $j \in [m]$, and $\eta = \frac{1}{l_\theta((\lambda)}$ according to Theorem 5 that for a fixed transition kernel $p \in \mathcal{P}$,*

$$\mathbf{V}_{\pi_{\theta_t},p}(\rho) - \mathbf{V}_{\pi_{\theta^*},p}(\rho) \leq \left\| \frac{1}{\mu} \right\|_\infty \frac{32 S M^2 F_\lambda}{U_p (1-\gamma)^6 t}\,. \tag{25}$$

As a third and last auxiliary result, we note that the RCMDP objective in Eq. 3 can be expressed in terms of the optimal (adversarial) transition dynamics at macro-iteration k can be expressed in terms of the transition dynamics found by LTMA and its error tolerance.

Lemma 7. *Let Φ be the objective in Eq. 3 and π^* its optimal solution. Then for any macro-step $k \geq 0$,*

$$\Phi(\pi_k) - \Phi(\pi^*) \leq \mathbf{V}_{\pi_k, p_k}(\rho) + \epsilon'_k - \mathbf{V}_{\pi^*, p_k}(\rho)\,, \tag{26}$$

where $\epsilon'_k > 0$ is the error tolerance for LTMA at iteration k.

Proof. The proof follows from the definition

$$\begin{aligned}
\Phi(\pi_k) - \Phi(\pi^*) &= \sup_{p \in \mathcal{P}} \mathbf{V}_{\pi_k, p}(\rho) - \sup_{p' \in \mathcal{P}} \mathbf{V}_{\pi^*, p'}(\rho) \\
&\leq \mathbf{V}_{\pi_k, p_k}(\rho) + \epsilon'_k - \sup_{p \in \mathcal{P}} \mathbf{V}_{\pi^*, p}(\rho) \quad \text{(error tolerance of LTMA)} \\
&\leq \mathbf{V}_{\pi_k, p_k}(\rho) + \epsilon'_k - \mathbf{V}_{\pi^*, p_k}(\rho)\,.
\end{aligned}$$

□

We now turn to the theorem, which matches Wang & Petrik (2024) up to a constant related to the range of the Lagrangian. The additional factor $\frac{1}{1-\gamma} \left\| \frac{1}{\mu} \right\|_\infty$ is consistent with the analysis of Mei et al. (2020), accounting for the possible mismatch between the initial sampling distribution μ and the true initial distribution ρ of the MDP.

Theorem 1 (Regret of mirror descent policy optimisation with robust Lagrangian)**.** *Using constant step size $\eta = 1/l_\theta$, Lagrangian gradient descent over logits yields at any macro-step $k \geq 0$*

$$\Phi(\pi_k) - \Phi(\pi^*) \leq \left\| \frac{1}{\mu} \right\|_\infty \frac{32 S M^2 F_{\lambda_k}}{U(1-\gamma)^6 k} + \epsilon'_k\,, \tag{27}$$

where $\epsilon'_k > 0$ is the tolerance of LTMA at macro-step k.

Proof. Since for the softmax parametrisation, mirror descent is equivalent to traditional gradient descent (see Table 5), the robust Lagrangian policy updates will produce a sequence of parameters $\{\theta_k\}_{k=1}^n$ obtained from the process

$$\theta_{k+1} = \theta_k - \eta \nabla \mathbf{V}_{\pi_{\theta_k}, p_k}(\rho)\,. \tag{28}$$

Due to the smoothness and continuity of the Lagrangian value in Lemma 5 and Lemma 4, it follows that for any $k' \geq 1$, there exists a parameter sequence $\left\{\hat{\theta}_k\right\}_{k=1}^{k'}$ obtained from non-robust policy gradient descent with nominal dynamics $\hat{p} = p_{k'}$ and Lagrangian multiplier $\hat{\lambda} = \lambda_{k'}$ such that $\hat{\theta}_{k'} = \theta_{k'}$, which is the following process with constant transition kernel:

$$\hat{\theta}_{k+1} = \hat{\theta}_k - \eta \nabla \mathbf{V}_{\pi_{\hat{\theta}_k}, \hat{p}}(\rho; \hat{\lambda})\,. \tag{29}$$

Applying Theorem 6 and accounting for the worst-case dynamics by letting $U = \inf_{p \in \mathcal{P}} U_p$ and $M = \sup_{p \in \mathcal{P}}$, we obtain

$$\mathbf{V}_{\pi_{k'}, p_{k'}}(\rho) - \mathbf{V}_{\pi^*, p_{k'}}(\rho) \leq \left\| \frac{1}{\mu} \right\|_\infty \frac{32 S M^2 F_{\lambda_{k'}}}{U(1-\gamma)^6 k'}\,. \tag{30}$$

Combining with Eq. 26 from Lemma 7, we obtain

$$\Phi(\pi_{k'}) - \Phi(\pi^*) \leq \left\| \frac{1}{\mu} \right\|_\infty \frac{32 S M^2 F_{\lambda_{k'}}}{U(1-\gamma)^6 k'} + \epsilon'_{k'}\,. \tag{31}$$

□

Theorem 1 implies an $\mathcal{O}(T^{-1})$ convergence rate to the optimal robust Lagrangian.

With entropy regularisation We now analyse the case where an additional negative entropy term is added to the Lagrangian. The result is useful to consider an additional term to the objective but also to consider the performance of mirror descent with KL-divergence based Bregman divergence. In the former case, the objective is formulated in terms of the entropy-regularised value function $\mathbf{V}^{\alpha}_{\pi,p}(\rho) - \tau\mathcal{H}(\rho,\pi,p)$ based on the discounted entropy

$$\mathcal{H}(\rho,\pi,p) = -\mathbb{E}\left[\sum_{l=0}^{\infty}\gamma^l \log(\pi(a_l|s_l))|\pi,p\right], \tag{32}$$

such that deterministic policies are penalised more strongly. In the latter case, the objective is based on $\mathbf{V}_{\pi,p}(\rho) - \tau\mathcal{H}(\rho,\pi,p)$ but it is useful to note the interpretation of PMD-PD as the entropy-regularised objective with costs based on $\tilde{\mathbf{c}}_k(s,a) - \tau\log(\pi_k(a|s))$,

$$\mathbf{V}^{\tau}_{\pi,p}(\rho) = \mathbb{E}\left[\sum_{l=0}^{\infty}\gamma^l\left(\underbrace{\mathbf{c}_k(s_l,a_l) + \tau\log\left(\frac{1}{\pi_k(a_l|s_l)}\right)}_{\text{cost}} + \underbrace{\tau\log(\pi(a_l|s_l))}_{\text{entropy regularisation}}\right)|\pi,p\right].$$

The aforementioned properties of smoothness and continuity can be easily extended to the entropy-regularised Lagrangian.

Lemma 8 (Smooth and continuous entropy-regularised Lagrangian)**.** *Let $\tau > 0$, $\tau = \Theta(\frac{1}{\log(A)})$ and let F_λ be chosen according to Eq. 11 or Eq. 12 (depending on the use of augmentation). Then under the softmax parametrisation, the entropy-regularised Lagrangian $\mathbf{V}(\theta) + \tau\mathcal{H}(\rho,\pi_\theta)$ is l_θ-smooth and L_θ-Lipschitz continuous with*

$$l_\theta = \mathcal{O}(\frac{F_\lambda}{(1-\gamma)^2}) \tag{33}$$

and

$$L_\theta = \mathcal{O}(\frac{F_\lambda}{(1-\gamma)^3}). \tag{34}$$

Proof. Let $k \geq 1$. From Lemma 5,

$$|\mathbf{V}(\theta') - \mathbf{V}(\theta) - \langle\nabla\mathbf{V}(\theta), \theta' - \theta\rangle| \leq \frac{16F_\lambda}{(1-\gamma)^3} \quad \forall\theta,\theta' \in \Theta.$$

From Lemma 14 in Mei et al. (2020), it follows that

$$|\mathcal{H}(\theta') - \mathcal{H}(\theta) - \langle\nabla\mathcal{H}(\theta), \theta' - \theta\rangle| \leq \frac{4 + 8\log(A)}{(1-\gamma)^3} \quad \forall\theta,\theta' \in \Theta.$$

Combining both, via triangle inequality we obtain

$$|\mathbf{V}(\theta') + \mathcal{H}(\theta') - (\mathbf{V}(\theta') + \mathcal{H}(\theta)) - \langle\nabla(\mathbf{V}+\mathcal{H})(\theta), \theta' - \theta\rangle| \leq \frac{16F_\lambda + \tau\,(4 + 8\log(A))}{(1-\gamma)^3} \quad \forall\theta,\theta' \in \Theta.$$

It follows that $\mathbf{V}(\theta) + \mathcal{H}(\theta)$ is l_θ-smooth with $l_\theta = \mathcal{O}(\frac{F_\lambda}{(1-\gamma)^3})$.

For Lipschitz continuity, a similar argument can be made. Note that since the entropy ranges in $[0, \log(A)]$, the Lipschitz bound on the discounted entropy is given by

$$|\mathcal{H}(\theta') - \mathcal{H}(\theta)| \leq \frac{\log(A)}{1-\gamma}.$$

From derivations in Lemma 4.4 in Wang et al. (2023),

$$\|\nabla\log(\pi(a|s))\| \leq \sqrt{2}.$$

Denoting $\mathbf{V}^\tau$ as the regularised objective, we have

$$\begin{aligned}\max_{\theta,\theta'} \left\| \frac{\mathbf{V}^\tau(\theta) - \mathbf{V}^\tau(\theta')}{\theta - \theta'} \right\| &= \max_{\theta} \left\| \frac{\partial \mathbf{V}^\tau(\theta)}{\partial \theta(s,\cdot)} \right\| \\ &\leq \frac{1}{(1-\gamma)} \left\| \nabla \log(\pi(a|s)) \right\| \left\| \mathbf{V}^\tau(\theta) \right\| \\ &\leq \frac{2\sqrt{2}(F_\lambda + \log(A))}{(1-\gamma)^2} \\ &= \mathcal{O}\left(\frac{F_\lambda}{(1-\gamma)^2} \right) .\end{aligned}$$

□

The following theorem follows directly from Theorem 6 of Mei et al. (2020), allowing an exponential convergence rate under gradient descent with entropy regularisation.

Theorem 2 (Convergence rate for softmax policies for Lagrangian gradient descent with entropy regularisation)**.** *Let* Π *be the softmax parametrisation. For Lagrangian gradient descent over logits and the conditions of Lemma 8, with settings of* $M = \inf_{p\in\mathcal{P},\pi\in\Pi} M_p(\pi)$*,* $U = \inf_{p\in\mathcal{P}} U_p$*, and* $\eta = \frac{1}{l_\theta}$ *according to Theorem 5, it follows that for any* $t \geq 0$*, for a fixed transition kernel* $p \in \mathcal{P}$*, and fixed* $\lambda_k = \lambda_{k-1}$*,*

$$\mathbf{V}^\tau_{\pi_t,p}(\rho) - \mathbf{V}^\tau_{\pi^*_\tau,p}(\rho) = \mathcal{O}\left(\left\| \frac{1}{\mu} \right\|_\infty \frac{F_\lambda}{(1-\gamma)} e^{-C\,(t-1)} \right) , \tag{35}$$

where $C = \frac{\eta}{S} \min_s \mu(s) U^2 M^{-1}$ *and* $\pi^*_\tau \in \arg\max_{\pi\in\Pi} \mathbf{V}^\tau_{\pi,p}(\rho)$*.*

Finally, we apply Theorem 2 to the robust Lagrangian problem 4, or more specifically, to the robust entropy-regularised Lagrangian problem

$$\min_\pi \left\{ \Phi^\tau(\pi) := \sup_{p\in\mathcal{P}} \max_{\lambda\geq 0} \mathbf{V}^\tau_{\pi,p}(\rho) \right\} . \tag{36}$$

Theorem 3 (Regret of mirror descent policy optimisation with robust Lagrangian with entropy regularisation)**.** *Using constant step size* $\eta = 1/l_\theta$*, Lagrangian gradient descent over softmax logits with entropy regularisation yields at any macro-step* $k \geq 0$

$$\Phi^\tau(\pi_k) - \Phi^\tau(\pi^*_\tau) = \mathcal{O}\left(\left\| \frac{1}{\mu} \right\|_\infty \frac{F_\lambda}{(1-\gamma)^2} e^{-C\,(k-1)} + \epsilon'_k \right) , \tag{37}$$

where F_λ *is set according to Eq. 11 or Eq. 12 depending on the use of (un-)augmented Lagrangian,* $\pi^*_\tau \in \arg\min_{\pi\in\Pi} \sup_{p\in\mathcal{P}} \mathbf{V}^\tau_{\pi,p}(\rho)$*, and* $\epsilon'_k > 0$ *is the tolerance of LTMA at macro-step* k *with respect to* $\sup_{p\in\mathcal{P}} \mathbf{V}^\tau_{\pi,p}(\rho)$*.*

Proof. Analogous to the proof of Theorem 1, we bound the regret by

$$\Phi^\tau(\pi_k) - \Phi^\tau(\pi^*_\tau) \leq \mathbf{V}^\tau_{\pi_{\theta_k},p_k}(\rho) + \epsilon_k - \mathbf{V}^\tau_{\pi^*_\tau,p_k}(\rho) ,$$

and formulate an equivalent gradient descent process

$$\hat{\theta}_{k+1} = \hat{\theta}_k - \eta \nabla \mathbf{V}^\tau_{\pi_{\hat{\theta}_k},\hat{p}}(\rho;\hat{\lambda}) . \tag{38}$$

Applying Theorem 2, we obtain for any $k' \geq 1$ that

$$\Phi^\tau(\pi_{k'}) - \Phi^\tau(\pi^*_\tau) = \mathcal{O}\left(\left\| \frac{1}{\mu} \right\|_\infty \frac{F_\lambda}{(1-\gamma)} e^{-C\,(k'-1)} + \epsilon'_{k'} \right) . \tag{39}$$

□

Theorem 2 implies an $\mathcal{O}(e^{-T})$ convergence rate to the optimal regularised robust Lagrangian, which is not necessarily optimal on the original robust Lagrangian problem.

4.4 Sample-based analysis

Having shown results for the oracle-based setting where values and policy gradients are given, we now turn to the sample-based analysis, where values and policy gradients need to be estimated from finite-sample data are accounted for in the convergence rate computations. The analysis modifies the techniques from Sample-based PMD-PD (Liu et al., 2021) by accounting for the mismatch between the nominal transition kernel and the optimal transition kernel in the min-max problem in Eq. 3. Their analysis makes use of a weighted Bregman divergence term formulated in Eq. 13. While the weighted Bregman divergence is not a proper Bregman divergence over policies, it is over the occupancy distribution, i.e. $B_{d_\rho^{\pi,p}}(\pi,\pi') = B(d_\rho^{\pi,p}, d_\rho^{\pi',p})$, as shown in Lemma 10 of Liu et al. (2021). The complexity analysis below is slightly simplified by dropping purely problem-dependent constants γ, η, ζ from the big-O statements.

As sample-based algorithms work with approximate value functions, Lemma 9 describes the conditions under which one can obtain an ϵ-approximation of the cost functions and the regularised augmented Lagrangian. The same settings as sample-based PMD in Lemma 15 of Liu et al. (2021) transfer to the robust sample-based PMD algorithm. We rephrase the lemma to include updated transition dynamics.

Lemma 9 (Value function approximation, Lemma 15 of Liu et al. (2021) rephrased)**.** *Let $k \geq 0$ be the macro-step of the PMD-PD algorithm. With parameter settings*

$$
\begin{aligned}
&K = \Theta\left(\frac{1}{\epsilon}\right), \quad t_k = \Theta\left(\log(\max\left\{1, \|\lambda_k\|_1\right\}\right), \quad \delta_k = \Theta\left(\frac{\delta}{K t_k}\right), \\
&M_{V,k} = \Theta\left(\frac{\log(1/\delta_k)}{\epsilon^2}\right), \quad N_{V,k} = \Theta\left(\log_{1/\gamma}\left(\frac{1}{\epsilon}\right)\right), \\
&M_{Q,k} = \Theta\left(\frac{(\max\left\{1, \|\lambda_k\|_1\right\} + \epsilon t_k)^2 \log(1/\delta_k)}{\epsilon^2}\right), \quad N_{Q,k} = \Theta\left(\log_{1/\gamma}\left(\frac{(\max\left\{1, \|\lambda_k\|_1\right\}}{\epsilon}\right)\right),
\end{aligned}
$$

the approximation is ϵ-optimal, i.e.

$$
\begin{aligned}
&\textbf{\textit{a)}} \quad |\hat{V}^i_{\pi_k^t, p_k}(\rho) - V^i_{\pi_k^t, p_k}(\rho)| \leq \epsilon \qquad \forall i = 1 \in [k] \\
&\textbf{\textit{b)}} \quad |\hat{\mathbf{Q}}^\alpha_{\pi_k^t, p_k}(s,a) - \tilde{\mathbf{Q}}^\alpha_{\pi_k^t, p_k}(s,a)| \leq \epsilon \qquad \forall (s,a) \in \mathcal{S} \times \mathcal{A},
\end{aligned}
$$

with probability $1-\delta$.

Proof. The full proof can be found in Appendix C.5. □

Denoting $\tilde{\mathbf{V}}$ as the unregularised augmented Lagrangian, we can show a relation between two otherwise unrelated policies based on their divergence to two consecutive policy iterates.

Lemma 10 (Bound on regularised augmented Lagrangian under approximate entropy-regularised NPG, Lemma 18 in Liu et al. (2021))**.** *Assume the same settings as in Lemma 9. Then the regularised augmented Lagrangian is bounded by*

$$
\tilde{\mathbf{V}}_{\pi_{k+1}, p_k}(s) + \frac{\alpha}{1-\gamma} B_{d_\rho^{\pi_{k+1}, p_k}}(\pi_{k+1}, \pi_k) \leq \tilde{\mathbf{V}}_{\pi, p_k}(s) + \frac{\alpha}{1-\gamma}\left(B_{d_\rho^{\pi, p_k}}(\pi, \pi_k) - B_{d_\rho^{\pi, p_k}}(\pi, \pi_{k+1})\right) + \Theta(\epsilon)
$$

Proof. The proof makes use of Lemma 18. In particular, it notes that $K = \Theta(1/\epsilon)$ and the value difference to π_k^* is bounded by $1/K$ after t_k steps. After applying the pushback property (Lemma 15) to π, π_k, and π_k^*, and derives an upper bound on the Bregman divergence w.r.t. π_k^* based on $B_{d_\rho^{\pi, p_k}}(\pi, \pi_{k+1}) + \left\|\log(\pi_k^*) - \log(\pi_k^{t+1})\right\|_\infty$, where the second term is $\Theta(1/K)$. We refer to Lemma 18 of Liu et al. (2021) for the detailed proof. □

Below we prove the sample complexity of robust sample-based PMD-PD. For policy updates, we use estimates of the Q-values and show the overall sample complexity for obtaining an average regret in the value, constraint-cost, and Lagrangian bounded by $\mathcal{O}(\epsilon)$. Since the theory on TMA is not yet well developed, the analysis will assume the oracle setting for transition kernel updates, in which case we simply count the number of

iterations. However, more generally, the analysis below holds whenever the number of samples within the LTMA loops does not exceed that of the rest of the algorithm (i.e. $\tilde{\mathcal{O}}(\epsilon^{-3})$ as will be shown below).

Theorem 4 (Sample complexity of Robust Sample-based PMD-PD)**.** *Choose parameter settings according to Lemma 9, and let* $\alpha = \frac{2\gamma^2 m\eta_\lambda}{(1-\gamma)^3}$, $\eta_\lambda = 1$, *and* $\eta = \frac{1-\gamma}{\alpha}$. *Moreover, let* $\epsilon'_k > 0$ *be the upper bound on the LTMA error tolerance at macro-iteration* k *such that* $\epsilon'_k = \Theta(\epsilon)$, *and let* $\mathcal{P}$ *be an uncertainty set contained in an* ℓ_1 (s,a)*-rectangular uncertainty set around the nominal distribution* $\bar{p} = p_0$ *such that*

$$\mathcal{P}_{s,a} \subseteq \{p \in \Delta(\mathcal{S}) : \|p(\cdot|s,a) - \bar{p}(\cdot|s,a)\|_1 \leq \psi_{s,a}\} \quad \forall (s,a) \in \mathcal{S} \times \mathcal{A}\,, \tag{40}$$

and let $\Delta_p = \max_{s,a} \psi_{s,a} = \Theta\left(\frac{(1-\gamma)}{\alpha \log(A)}\right)$. *Then within a total number of queries to the generative model equal to* $T = \sum_{k=1}^{K}\left(M_{V,k}N_{V,k} + \sum_{t=0}^{t_k-1} M_{Q,k}N_{Q,k} + t'_k\right) = \tilde{\mathcal{O}}(\epsilon^{-3})$, *Robust Sample-based PMD-PD provides three guarantees.*

(a) regret upper bound:

$$\frac{1}{K}\sum_{k=1}^{K}\left(V_{\pi_k,p_k}(\rho) - V_{\pi^*,p_k}(\rho)\right) = \mathcal{O}(\epsilon)\,. \tag{41}$$

(b) constraint-cost upper bound:

$$\max_{j\in[m]} \frac{1}{K}\sum_{k=1}^{K} V^j_{\pi_k,p_k}(\rho) \leq \mathcal{O}(\epsilon)\,. \tag{42}$$

c) robust Lagrangian regret upper bound: *under the conditions of Lemma 2, and with a number of LTMA iterations* $t'_k = \Theta\left(\frac{F_{\lambda_k} M}{\epsilon'_k(1-\gamma}\right)$, *we have*

$$\frac{1}{K}\sum_{k=1}^{K}\Phi(\pi_k) - \Phi(\pi^*) = \mathcal{O}(\epsilon)\,.$$

Proof. **(a)** The proof uses similar derivations as in Theorem 3 of Liu et al. (2021) and then accounts for the performance difference due to transition kernels.

Note that

$$\begin{aligned}
&\tilde{\mathbf{V}}^{\alpha}_{\pi_{k+1},p_{k+1}}(\rho) \\
&= V_{\pi_{k+1},p_{k+1}}(\rho) + \left\langle \lambda_k + \eta_\lambda \hat{V}^{1:m}_{\pi_k,p_k}(\rho), V^{1:m}_{\pi_{k+1},p_{k+1}}(\rho)\right\rangle + \frac{\alpha}{1-\gamma} B_{d_\rho^{\pi_{k+1},p_{k+1}}}(\pi_{k+1},\pi_k) \\
&\leq V_{\pi^*,p_{k+1}}(\rho) + \left(\frac{\alpha}{1-\gamma}\left(B_{d_\rho^{\pi^*,p_{k+1}}}(\pi^*,\pi_k) - B_{d_\rho^{\pi^*,p_{k+1}}}(\pi^*,\pi_{k+1})\right) + \Theta(\epsilon)\right)\,,
\end{aligned}$$

where the last step follows from setting $\pi = \pi^*$ in Lemma 10 and noting that, since $\lambda_{k,j} + \eta_\lambda \hat{V}^j_{\pi_k,p_k}(\rho) \geq 0$ by property 2 of Lemma 12 and $V^j_{\pi^*,p}(\rho) \leq 0$ for any $j \in [m]$, their inner product will be negative.

It then follows that

$$\begin{aligned}
V_{\pi_{k+1},p_{k+1}}(\rho) - V_{\pi^*,p_k}(\rho) \leq \frac{\alpha}{1-\gamma}&\left(B_{d_\rho^{\pi^*,p_{k+1}}}(\pi^*,\pi_{k+1}) - B_{d_\rho^{\pi^*,p_{k+1}}}(\pi^*,\pi_{k+1})\right) + \Theta(\epsilon) \\
&- \left\langle \lambda_k + \eta_\lambda \hat{V}^{1:m}_{\pi_k,p_k}(\rho), V^{1:m}_{\pi_{k+1},p_{k+1}}(\rho)\right\rangle - \frac{\alpha}{1-\gamma} B_{d_\rho^{\pi_{k+1},p_{k+1}}}(\pi_{k+1},\pi_k)\,.
\end{aligned}$$

Filling in the lower bound on the inner product from Eq. 41 from Liu et al. (2021) (see also Lemma 13),

$$\begin{aligned}
&\left\langle \lambda_k + \eta_\lambda \hat{V}^{1:m}_{\pi_k,p_k}(\rho), V^{1:m}_{\pi_{k+1},p_{k+1}}(\rho)\right\rangle \geq \frac{1}{2\eta_\lambda}\left(\|\lambda_{k+1}\|^2 - \|\lambda_k\|^2\right) + \frac{\eta}{2}\left(\left\|V^{1:m}_{\pi_k,p_k}(\rho)\right\|^2 - \left\|V^{1:m}_{\pi_k+1,p_{k+1}}(\rho)\right\|^2\right) \\
&- 2\eta_\lambda \left\langle V^{1:m}_{\pi_{k+1},p_{k+1}}(\rho), \epsilon_{k+1}\right\rangle - \frac{\gamma^2\eta_\lambda}{(1-\gamma)^4} B_{d_\rho^{\pi_{k+1},p_{k+1}}}(\pi_{k+1},\pi_k)\,,
\end{aligned}$$

we get

$$
\begin{aligned}
&V_{\pi_{k+1},p_{k+1}}(\rho) - V_{\pi^*,p_{k+1}}(\rho) \leq (1+\Delta_p)\left(\frac{\alpha}{1-\gamma}\left(B_{d_\rho^{\pi^*,p_k}}(\pi^*,\pi_k) - B_{d_\rho^{\pi^*,p_k}}(\pi^*,\pi_{k+1})\right) + \Theta(\epsilon)\right) \\
&+ \frac{1}{2\eta_\lambda}\left(\|\lambda_k\|^2 - \|\lambda_{k+1}\|^2\right) + \frac{\eta}{2}\left(\left\|V^{1:m}_{\pi_k+1,p_{k+1}}(\rho)\right\|^2 - \left\|V^{1:m}_{\pi_k,p_k}(\rho)\right\|^2\right) \\
&+ 2\eta_\lambda\left\langle V^{i:m}_{\pi_{k+1},p_{k+1}}(\rho), \epsilon_{k+1}\right\rangle - \frac{\alpha(1-\gamma)^3 - \gamma^2\eta_\lambda}{(1-\gamma)^4} B_{d_\rho^{\pi_{k+1},p_{k+1}}}(\pi_{k+1},\pi_k) \\
&\leq \frac{\alpha}{1-\gamma}\left(B_{d_\rho^{\pi^*,p_k}}(\pi^*,\pi_k) - B_{d_\rho^{\pi^*,p_k}}(\pi^*,\pi_{k+1})\right) + \Theta(\epsilon) \\
&+ \frac{1}{2\eta_\lambda}\left(\|\lambda_k\|^2 - \|\lambda_{k+1}\|^2\right) + \frac{\eta}{2}\left(\left\|V^{1:m}_{\pi_k+1,p_{k+1}}(\rho)\right\|^2 - \left\|V^{1:m}_{\pi_k,p_k}(\rho)\right\|^2\right) \\
&+ 2\eta_\lambda\left\langle V^{i:m}_{\pi_{k+1},p_{k+1}}(\rho), \epsilon_{k+1}\right\rangle
\end{aligned}
$$

Note that the upper bound on $\eta_\lambda/2\left\|V^{1:m}_{\pi_k,p_k}(\rho) - V^{1:m}_{\pi_{k+1},p_{k+1}}(\rho)\right\|^2 \leq B_{d_\rho^{\pi_{k+1},p_{k+1}}}(\pi_{k+1},\pi_k)$ is analogous to Eq.16 of Liu et al. (2021) but with a divergence term depending on p_{k+1} rather than p_k. Note that the term $\frac{\alpha(1-\gamma)^3-\gamma^2\eta_\lambda}{(1-\gamma)^4} B_{d_\rho^{\pi_{k+1},p_{k+1}}}(\pi_{k+1},\pi_k) \geq 0$, so it drops from the upper bound in the subsequent analysis.

Due to the approximations and Lemma 13, the regret introduces a term

$$
\Delta_k = \Theta(\epsilon) + \langle\lambda_{k-1},\epsilon_k\rangle - \eta_\lambda\left\langle\epsilon_{k-1}, V^{1:m}_{\pi_k,p_k}(\rho)\right\rangle + \eta_\lambda\left\langle\epsilon_k + 2V^{1:m}_{\pi_k,p_k}(\rho), \epsilon_k\right\rangle .
$$

Using Lemma 21 and telescoping, it follows that

$$
\begin{aligned}
&\sum_{k=1}^{K}\left(V_{\pi_k,p_k}(\rho) - V_{\pi^*,p_k}(\rho)\right) \qquad (43)\\
&\leq \sum_{k=1}^{K}\frac{\alpha}{1-\gamma}\left(B_{d_\rho^{\pi^*,p_k}}(\pi^*,\pi_k) - B_{d_\rho^{\pi^*,p_k}}(\pi^*,\pi_{k+1})\right) + \Delta_k \\
&+ \frac{\eta_\lambda}{2}\left(\left\|V^{1:m}_{\pi_k,p_k}(\rho)\right\|^2 - \left\|V^{1:m}_{\pi_{k-1},p_{k-1}}(\rho)\right\|^2\right) + \frac{1}{2\eta_\lambda}\left(\|\lambda_{k-1}\|^2 - \|\lambda_k\|^2\right) \\
&\leq \sum_{k=1}^{K}\frac{\alpha}{1-\gamma}\left(B_{d_\rho^{\pi^*,p_k}}(\pi^*,\pi_{k-1}) - B_{d_\rho^{\pi^*,p_k}}(\pi^*,\pi_k) + (p_k - p_{k-1})\log(A)\right) + \Delta_k \\
&+ \frac{\eta_\lambda}{2}\left(\left\|V^{1:m}_{\pi_k,p_k}(\rho)\right\|^2 - \left\|V^{1:m}_{\pi_{k-1},p_{k-1}}(\rho)\right\|^2\right) + \frac{1}{2\eta_\lambda}\left(\|\lambda_{k-1}\|^2 - \|\lambda_k\|^2\right) \quad \text{(using Lemma 21 and Lemma 16)} \\
&\leq \frac{\alpha}{1-\gamma}\left(B_{d_\rho^{\pi^*,p_0}}(\pi^*,\pi_0) - B_{d_\rho^{\pi^*,p_K}}(\pi^*,\pi_K)\right) + \frac{\eta_\lambda}{2}\left(\left\|V^{1:m}_{\pi_K,p_K}(\rho)\right\|^2 - \left\|V^{1:m}_{\pi_0,p_0}(\rho)\right\|^2\right) \\
&+ \frac{1}{2\eta_\lambda}\left(\|\lambda_0\|^2 - \|\lambda_K\|^2\right) + \alpha\left\|p_K - p_0\right\|_1\mathcal{O}\left(\frac{\log(A)}{1-\gamma}\right) + \sum_{k=1}^{K}\Delta_k \\
&\leq \frac{\alpha}{1-\gamma}\left(B_{d_\rho^{\pi^*,p_0}}(\pi^*,\pi_0) - B_{d_\rho^{\pi^*,p_K}}(\pi^*,\pi_K)\right) + \frac{\eta_\lambda m}{(1-\gamma)^2} + \alpha\Delta_p\mathcal{O}\left(\frac{\log(A)}{1-\gamma}\right) + \sum_{k=1}^{K}\Delta_k , \qquad (44)
\end{aligned}
$$

where the last step follows from $\|\lambda_0\| \leq \frac{m\eta_\lambda^2}{(1-\gamma)^2}$ and $\|V_{\pi_K,p_K}(\rho)\|^2 \leq \frac{m}{(1-\gamma)^2}$. Since $\alpha = \frac{1-\gamma}{\eta}$, $\eta_\lambda = \frac{\alpha}{(1-\gamma)^3}2\gamma^2$, $\frac{\alpha}{1-\gamma}\log(A) + \frac{\eta_\lambda m}{(1-\gamma)^2} = \mathcal{O}(1)$. Moreover, since $\Delta_k = \mathcal{O}(\max\{1, \|\lambda_k\|_1\}\epsilon)$ and Lemma 22 shows that with probability $1-\delta$, $\|\lambda_k\| \leq \|\lambda_k\|_1 = \mathcal{O}(1)$, we have $\Delta_k = \mathcal{O}(1)$ with probability $1-\delta$. Finally, $\alpha\Delta_p\mathcal{O}\left(\frac{\log(A)}{1-\gamma}\right)$ is also $\mathcal{O}(1)$ since $\Delta_p = \Theta\left(\frac{(1-\gamma)}{\alpha\log(A)}\right)$. All terms then reduce to $\mathcal{O}(\epsilon)$ after division by $K = \Theta(\frac{1}{\epsilon})$.

(b) Denoting the approximation error at iteration $k \in [K]$ and constraint j as $\epsilon_{k,j}$, and observing that for any $j \in [m]$

$$\begin{aligned}\lambda_{k,j} &= \max\left\{-\eta_\lambda \hat{V}^j_{\pi_k,p_k}(\rho), \lambda_{k-1,j} + \eta_\lambda \hat{V}^j_{\pi_k,p_k}(\rho)\right\} \\ &\geq \lambda_{k-1,j} + \eta_\lambda \hat{V}^j_{\pi_{\theta_k},p_k}(\rho)\,, \end{aligned} \tag{45}$$

it follows that

$$\begin{aligned}\frac{1}{K}\sum_{k=1}^{K} V^j_{\pi_k,p_k}(\rho) &= \frac{1}{K}\sum_{k=1}^{K}\left(\hat{V}^j_{\pi_k,p_k}(\rho) - \epsilon_{k,j}\right) \\ &\leq \frac{1}{K}\sum_{k=1}^{K}\left(\frac{\lambda_{k,j}-\lambda_{k-1,j}}{\eta_\lambda} - \epsilon_{k,j}\right) \quad \text{(from Eq. 45)} \\ &\leq \frac{\lambda_{K,j}}{\eta_\lambda K} - \frac{1}{K}\sum_{k=1}^{K}\epsilon_{k,j} \quad \text{(telescoping and non-negativity condition in Lemma 12} \\ &\leq \frac{\|\lambda^*\| + \|\lambda^* - \lambda_K\|}{\eta_\lambda K} - \frac{1}{K}\sum_{k=1}^{K}\epsilon_{k,j} \quad \text{(since } \lambda_{K,j} \leq \|\lambda_K\| \leq \|\lambda^*\|_+ \|\lambda^* - \lambda_K\|) \\ &\leq \frac{\|\lambda^*\|_+ \|\lambda^* - \lambda_K\|}{\eta_\lambda K} + \mathcal{O}(\epsilon) \quad \text{(since by Lemma 9, } K = \Theta\left(\frac{1}{\epsilon}\right) \text{ and } \epsilon_{k,j} = \mathcal{O}(\epsilon)) \\ &= \mathcal{O}(1/K) + \mathcal{O}(\epsilon) \quad \text{(by Lemma 22)} \\ &= \mathcal{O}(\epsilon) \quad (\text{ since } K = \Theta\left(\frac{1}{\epsilon}\right))\,. \end{aligned}$$

c) For any $k \in [K]$, it follows from Lemma 2 that

$$t'_k = \Theta\left(\frac{F_{\lambda_k} M}{\epsilon'_k(1-\gamma)}\right)$$

is sufficient to obtain an ϵ'_k-optimal transition kernel from LTMA. Using Lemma 7, combining the errors from **a)** and **b)** for the cost and constraint-costs, and noting that $\epsilon'_k = \Theta(\epsilon)$, it follows that

$$\begin{aligned}\frac{1}{K}\sum_{k=1}^{K}(\Phi(\pi_k) - \Phi(\pi^*)) &\leq \frac{1}{K}\sum_{k=1}^{K}(\mathbf{V}_{\pi_k,p_k}(\rho) + \epsilon'_k - \mathbf{V}_{\pi^*,p_k}(\rho)) \\ &= \mathcal{O}(F_\lambda \epsilon + \epsilon) \\ &= \mathcal{O}(F_\lambda \epsilon)\,, \end{aligned}$$

where $F_\lambda = \max_{k\in[K]} F_{\lambda_k}$. Due to the results in Lemma 22, $\lambda_k = \mathcal{O}(1)$ and $F_\lambda = \mathcal{O}(1)$. Moreover $\eta_\lambda V^j_{\pi_k,p_k} \leq \frac{1}{1-\gamma} = \mathcal{O}(1)$ for all $j \in [m]$ and all $k \in [K]$, which concludes the proof for both the augmented and unaugmented case.

Sample complexity: Following the loop in Algorithm 1, plugging in the settings from Lemma 9), and omitting logarithmic factors, the total number of calls to the generative model is given by

$$\begin{aligned}T &= \sum_{k=1}^{K}\left(M_{V,k}N_{V,k} + \sum_{t=0}^{t_k-1} M_{Q,k}N_{Q,k} + t'_k\right) \\ &= \mathcal{O}(\epsilon^{-1})\left(\epsilon^{-2}\tilde{\mathcal{O}(1)} + \tilde{\mathcal{O}}(1)\epsilon^{-2}\tilde{\mathcal{O}}(1)\right) + \sum_{k=1}^{K} t'_k \\ &= \tilde{\mathcal{O}}(\epsilon^{-3} + \epsilon^{-1}\frac{F_{\lambda_k} M}{\epsilon'(1-\gamma)}) \\ &= \tilde{\mathcal{O}}(\epsilon^{-3})\,, \end{aligned}$$

where the last step follows from $\epsilon'^{-1} = \mathcal{O}(\epsilon^{-1})$, $F_\lambda = \mathcal{O}(1)$, and M is a problem-specific constant that depends on the space of policies and the uncertainty set. □

4.5 KL-based Mirror Descent in continuous state-action spaces

To apply the above analysis in continuous state-action spaces, we follow a similar algorithm but replace the tabular approximation with function approximation (see Algorithm 2). A further change is that we still seek to maintain similar theoretical results, so we motivate the extension to continuous state-action spaces using a continuous pseudo-KL divergence of occupancy. Lemma 10 in Liu et al. (2021) defines a pseudo-KL divergence of occupancy within a discrete state-action space, which is shown to be a Bregman divergence of occupancy distributions. To demonstrate the wider applicability of the theory, we extend this definition straightforwardly to continuous state-action spaces below.

Definition 2. ***Continuous pseudo-KL divergence of occupancy.*** *Define the generating function* $h : \Delta^A \to \mathbb{R}$ *according to*

$$h(d_\rho^{\pi,p}) = \int_{\mathcal{S}\times\mathcal{A}} d_\rho^{\pi,p}(s,a)\log(d_\rho^{\pi,p}(s,a))\mathrm{d}s\mathrm{d}a - \int_{\mathcal{S}} d_\rho^{\pi,p}(s)\log(d_\rho^{\pi,p}(s))\mathrm{d}s\,.$$

Then its Bregman divergence is given by the continuous pseudo KL-divergence, defined as

$$B(d_\rho^{\pi,p}, d_\rho^{\pi',p}; h) := \int_{\mathcal{S}\times\mathcal{A}} d_\rho^{\pi,p}(s,a)\log\left(\frac{d_\rho^{\pi,p}(s,a)/d_\rho^{\pi,p}(s)}{d_\rho^{\pi',p}(s,a)/d_\rho^{\pi',p}(s)}\right)\mathrm{d}s\,. \tag{46}$$

Moreover, it is equivalent to a weighted Bregman divergence over policies $B_{d_\rho^{\pi,p}}(\pi,\pi')$ *over continuous state-action spaces.*

Proof. From the definition in Eq.46 and $\nabla h(d_\rho^{\pi,p})|_{s,a} = \log(d_\rho^{\pi,p}(s,a)) - \log(d_\rho^{\pi,p}(s))$ it follows that

$$\begin{aligned}
B(d_\rho^{\pi,p}, d_\rho^{\pi',p}; h) &= h(d_\rho^{\pi,p}) - h(d_\rho^{\pi'}) - \left\langle \nabla h(d_\rho^{\pi',p}), d_\rho^{\pi,p} - d_\rho^{\pi',p}\right\rangle \\
&= \int_{\mathcal{S}\times\mathcal{A}} d_\rho^{\pi,p}(s,a)\log(d_\rho^{\pi,p}(s,a))\mathrm{d}s\mathrm{d}a - \int_{\mathcal{S}} d_\rho^{\pi,p}(s)\log(d_\rho^{\pi,p}(s))\mathrm{d}s \\
&- \int_{\mathcal{S}\times\mathcal{A}} d_\rho^{\pi'}(s,a)\log(d_\rho^{\pi',p}(s,a))\mathrm{d}s\mathrm{d}a + \int_{\mathcal{S}} d_\rho^{\pi',p}(s)\log(d_\rho^{\pi',p}(s))\mathrm{d}s \\
&- \int \left(d_\rho^{\pi,p}(s,a) - d_\rho^{\pi'}(s,a)\right)\left(\log(d_\rho^{\pi',p}(s,a)) - \log(d_\rho^{\pi',p}(s))\right)\mathrm{d}s\mathrm{d}a \\
&= \int_{\mathcal{S}\times\mathcal{A}} d_\rho^{\pi,p}(s,a)\log(d_\rho^{\pi,p}(s,a)/d_\rho^{\pi',p}(s,a))\mathrm{d}s\mathrm{d}a - \int_{\mathcal{S}} d_\rho^{\pi,p}(s)\log(d_\rho^{\pi,p}(s)/d_\rho^{\pi',p}(s))\mathrm{d}s \\
&= \int_{\mathcal{S}\times\mathcal{A}} d_\rho^{\pi,p}(s,a)\log\left(\frac{d_\rho^{\pi,p}(s,a)/d_\rho^{\pi,p}(s)}{d_\rho^{\pi',p}(s,a)/d_\rho^{\pi',p}(s)}\right)\mathrm{d}s\mathrm{d}a
\end{aligned}$$

The equivalence to the weighted Bregman divergence over policies is shown by extending Eq. 13 to continuous state-action spaces and noting that $d_\rho^{\pi,p}(s,a) = d_\rho^{\pi,p}(s)\pi(a|s)$. □

4.6 MDPO-Robust-Lag: a practical implementation

To implement the algorithm in practice, we use Mirror Descent Policy Optimisation (MDPO) (Tomar et al., 2022). On-policy MDPO optimises the objective

$$\pi_{k+1} \leftarrow \arg\max_\theta J_{\mathrm{MDPO}} := \mathbb{E}_{s\sim d_\rho^{\pi_k,p_k}(s)}\left[\mathbb{E}_{a\sim\pi_\theta}[\hat{A}_{\pi_k,p_k}(s,a)] - \alpha D_{\mathrm{KL}}(\pi_\theta(\cdot|s),\pi_k(\cdot|s))\right] \tag{47}$$

based on t_k steps of SGD steps per batch k, which corresponds to the macro-iteration. For macro-iteration k, it defines the policy gradient for $\theta = \theta_k^t$ at any iteration $t \in [t_k]$ as

$$\nabla J_{\mathrm{MDPO}}(\theta,\theta_k) = \mathbb{E}_{s\sim d_\rho^{\pi_{\theta_k},p_k}(s)}\left[\mathbb{E}_{a\sim\pi_{\theta_k}}\left[\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)}\nabla_\theta\log(\pi_\theta(a|s))\hat{A}_{\pi_{\theta_k},p_k}(s,a)\right] - \alpha\nabla_\theta D_{\mathrm{KL}}(\pi_\theta(\cdot|s),\pi_{\theta_k}(\cdot|s))\right], \tag{48}$$

Algorithm 2 Robust Sample-based PMD-PD (continuous setting)

Inputs: Discount factor $\gamma \in [0,1)$, sample sizes $M_{Q,k}, M_{V,k}$ and episode lengths $N_{Q,k}, N_{V,k}$ for all $k \in [K]$, learning rate η, dual learning rate η_λ, error tolerance for LTMA $\epsilon_0' > 0$, and penalty coefficients $\alpha > 0$ and $\alpha_p > 0$.

Initialise: π_0 as uniform random policy, $\lambda_{0,j} = \max\left\{0, -\eta_\lambda \hat{V}_{\pi_0,p_0}(\rho)\right\}$, p_0 as the nominal transition kernel $\bar{p}$.

for $k \in \{1,\dots,K\}$ **do**

  for $t \in \{0,\dots,t_k-1\}$ **do**

    Generate $M_{Q,k}$ samples of length $N_{Q,k}$ based on π_k^t and p_k.

    Update $\hat{\mathbf{Q}}_{\pi_k^t,k} \leftarrow \arg\min_{Q\in\mathcal{F}_Q} \frac{1}{M_{Q,k}} \sum_{j=1}^{M_{Q,k}} \left(Q(s_0^j,a_0^j) - \sum_{l=0}^{N_{V,k}-1} \gamma^l \left[\tilde{\mathbf{c}}_k(s_l^j,a_l^j)\right]\right)^2$.

    ▷ Policy mirror descent

    $\theta_k^{t+1} \leftarrow \arg\min_{\theta\in\Theta} \eta \left\langle \nabla_\theta \hat{\mathbf{Q}}_{\pi_k^t,p_k}, \theta \right\rangle + \alpha B_{d_\rho^{\pi_\theta,p_k}}(\theta,\theta_k)$. ▷ separate value and divergence estimates

    Define $\pi_k^{t+1} := \pi_{\theta_k^{t+1}}$

  end for

  Define $\pi_{k+1} := \pi_k^{t_k}$

  ▷ Update transition kernel with LTMA (error tolerance fixed to ϵ_0' or $\epsilon_{k+1}' = \gamma\epsilon_k'$)

  for $t' \in \{0,\dots,t_k'-1\}$ **do**

    $\xi_k^{t'+1} \leftarrow \arg\max_{\xi\in\mathcal{U}} \eta_p \left\langle \nabla_\xi \hat{\mathbf{V}}_{\pi_k^t,p_k^{t'}}(\rho), \xi \right\rangle - \alpha_p B_{d_\rho^{\pi_{k+1},p_k^{t'}}}(\xi,\xi_k^{t'})$. ▷ Monte Carlo or other techniques

    Define $p_k^{t'+1} := p_{\xi_k^{t'+1}}$.

  end for

  Define $p_{k+1} := p_k^{t_k'}$.

  ▷ Dual update of augmented Lagrangian multipliers

  Generate $M_{V,k+1}$ samples of length $N_{V,k+1}$ based on π_{k+1} and p_{k+1}.

  Estimate $\hat{\mathbf{V}}^i_{\pi_{k+1}}(\rho) \leftarrow \arg\min_{V\in\mathcal{F}_V} \frac{1}{M_{V,k+1}} \sum_{j=1}^{M_{V,k+1}} \left(V(s_0^j) - \sum_{l=1}^{N_{V,k+1}} \gamma^l c_i(s_l^j,a_l^j)\right)^2$ for all $i \in [m]$.

  $\lambda_{k+1,i} \leftarrow \max\left\{-\eta_\lambda \hat{V}^i_{\pi_{k+1}}(\rho), \lambda_{k,i} + \eta_\lambda \hat{V}^i_{\pi_{k+1}}(\rho)\right\}$ for all $i \in [m]$.

end for

where in our case the advantage $\hat{A}_{\pi_{\theta_k},p_k}(s,a)$ is implemented based on the Lagrangian. The use of the importance weight $\frac{\pi_\theta}{\pi_{\theta_k}}$ corrects for the deviation from the theory, where it is assumed that the data at each epoch is generated from π_θ rather than π_{θ_k}. However, there still remains some gap of the practical implementation to our theoretical algorithm in that the state occupancy of MDPO is given by the old policy whereas in theory it is based on the current. The implementation of advantage values is based on generalised advantage estimation (Schulman et al., 2018) and the critic estimates the value and the constraint-costs based on an MLP architecture with $1+m$ heads. The update of the multipliers is based on the observed constraint-costs in separate samples.

To form a robust-constrained variant of MDPO, we use Lagrangian relaxation similar to the above (i.e. Algorithm 1), which differs from Adversarial RCPG (Bossens, 2024) in that multiple constraints are considered and that we preserve the structure of the PMD-PD algorithm (Liu et al., 2021) and the use of LTMA similar follows the TMA implementation from Wang & Petrik (2024), which applies a projected gradient descent based on clipping to the uncertainty set bounds. Due to its similarity to PPO-Lagrangian (or PPO-Lag for short) (Ray et al., 2019) and the use of updates in the robust Lagrangian, we name the algorithm **MDPO-Robust-Lag**. While the MDPO-Robust-Lag is based on traditionally clipping the multiplier, we also implement a version with the augmented Lagrangian as suggested by the theory and shown in Algorithm 1 and 2, called **MDPO-Robust-Augmented-Lag**. The practical implementation of the KL estimate follows the standard implementation in StableBaselines3, i.e. $D_{\mathrm{KL}}(\pi(\cdot|s),\pi'(\cdot|s)) \approx \mathbb{E}\left[e^{\log(\pi(a|s))-\log(\pi'(a|s))} - 1 - (\log(\pi(a|s)) - \log(\pi'(a|s)))\right]$.

5 Experiments

To demonstrate the use of MDPO for RCMDPs, we assess MDPO-Robust-Lag on three RCMDP domains. We compare MDPO-Robust-Lag to robust-constrained algorithms from related algorithm families, namely Monte Carlo based optimisation (**RMCPMD** (Wang & Petrik, 2024) in particular) and proximal policy optimisation with function approximiation (**PPO-Lag** (Achiam et al., 2017) in particular). All the involved algorithm families (MDPO, MC, and PPO) are included with four variants, namely with and without Lagrangian (indicated by the -Lag suffix) and with and without robustness training (indicated by the R- prefix or the -Robust- infix). Additionally, the MDPO-Lag implementations also have Augmented-Lag variants which use the augmented Lagrangian as proposed in the theory section. Further implementation details of the algorithms included to the study can be found in Appendix D.

After training with the above-mentioned algorithms, the agents are subjected to a test which presents transition dynamics based on the distortion levels setup in Wang & Petrik (2024). The test procedure follows the same general procedure for all domains. For distortion level $x \in [0,1]$, the agent is subjected to a perturbation such that if the nominal model is ξ_c and the uncertainty set varies the parameter in $[\underline{\xi},\overline{\xi}]$, the transition dynamics kernel parameter of the test with distortion level x is given for each dimension by either $\xi_i = \xi_c + x^2(\overline{\xi}-\xi_c)$ or $\xi_i = \xi_c + x^2(\underline{\xi}-\xi_c)$. We slightly improve the test evaluation by considering all the $\pm$ directions rather than just a single one, amounting to a total number of test evaluations $N = n_{\text{test}} \times 2^n$, where n is the dimensionality of the uncertainty set parametrisation and n_{test} is the number of episodes per perturbation.

To summarise the results, we use the penalised return, a popular summary statistic for robust constrained RL (Mankowitz et al., 2020; Bossens, 2024). The statistic is defined for a maximisation problem as $R_{\text{pen}} = V(\rho) - \lambda_{\max}\sum_{j=1}^m \max(0, C_j(\rho))$ where $V(\rho)$ denotes the value (negative of the cost) from the starting distribution and $C_j(\rho)$ denotes the j'th constraint-cost from the starting distribution. In addition, we also formulate the *signed penalised return* $R^{\pm}_{\text{pen}} = V(\rho) - \sum_{j=1}^m \lambda_{\max} C_j(\rho)$. We note that the penalised return matches the objective of the constrained methods and the return matches the objective of the unconstrained methods. The signed penalised return might be especially useful for robust problems; as even our extensive testing procedure may not find the worst case in the uncertainty set, and it is also of interest to generalise even beyond the uncertainty set, a solution that has negative cost can be evaluated more positively since it is more likely not to have a positive cost in the worst-case transition kernel. The summary statistics reported included both the mean and the minimum since the minimum indicates the effectiveness in dealing with the

minimax problem (the worst-case robustness). However, as a note of caution, the minimum is challenging to establish accurately because even though we test many environments it may still not contain the worst-case environment.

We report the results based on two distinct sample budgets, one with a limited number of time steps (200–500 times the maximal number of time steps in the episode) and one with a large number of time steps (2000–5000 times the maximal number of time steps in the episode). This setup allows to assess the sample-efficiency and convergence properties more clearly.

An additional set of experiments is to establish which kind of schedule for α_k works best for MDPO algorithms, and the robust-(constrained) variants in particular (see Apppendix G). From this set of experiments, it is confirmed that a fixed setting works reasonably well across problems. For simplicity, this setting will be used for all MDPO based algorithms throughout the remaining experiments.

5.1 Cartpole

To assess our algorithm on an RCMDP, we introduce the robust constrained variant of the well-known Cartpole problem (Barto et al., 1983; Brockman et al., 2016) by modifying the RMDP from Wang & Petrik (2024). The problem involves a mechanical simulation of a frictionless cart moving back and forth to maintain a pole, which is attached to the cart as an un-actuated joint, in the upright position. The agent observes $x, \dot{x}, \theta, \dot{\theta}$ and takes actions in $\{\text{left}, \text{right}\}$, which apply a small force moving either left or right. The agent receives a reward of $+1$ until either the cart goes out of the $[-2.4, 2.4]$m bounds or the pole has an angle outside the range $[-12°, 12°]$. In contrast to the traditional problem, the mechanics are not deterministic, and following Wang & Petrik (2024), transitions dynamics models are multi-variate Gaussians of the form

$$p(s'|s,a) = \frac{1}{(2\pi)^2|\Sigma|^{1/2}} e^{-\frac{1}{2}(s'-\mu_c(s,a))^\intercal \Sigma^{-1}(s'-\mu_c(s,a))}, \tag{49}$$

where $\mu_c(s,a)$ is the deterministic next state given $(s,a) \in \mathcal{S} \times \mathcal{A}$ based on the mechanics of the original Cartpole problem. We first train and evaluate algorithms in a CMDP problem for their ability to adhere to safety constraints. We then also show the benefits of robustness training and evaluate algorithms in an RCMDP problem.

To formulate safety constraints into the above cartpole problem, we define the instantanous constraint-cost as $c_t = |\dot{x}_t| - d$, where d is a constant, set to $d = 0.15$ in the experiments, and $\dot{x}$ is the velocity of the cart. The safety constraint of the agent is to maintain constraint $C = \mathbb{E}[\sum_t \gamma^t c_t] \leq 0$.

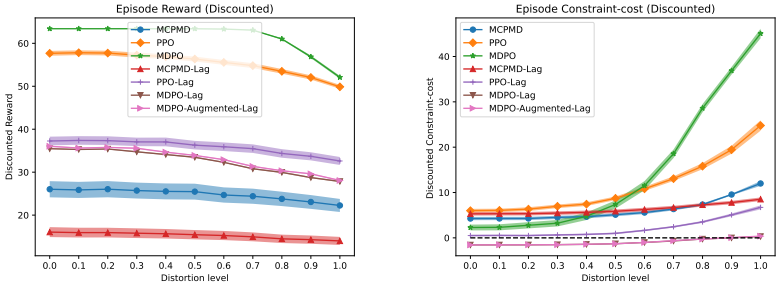
To incorporate robustness into our setting, we introduce delta-perturbations as in Wang & Petrik (2024), using an uncertainty set with parametrisation

$$p(s'|s,a) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(s'-(1+\delta)\mu_c(s,a))^\intercal \Sigma^{-1}(s'-(1+\delta)\mu_c(s,a))}, \tag{50}$$

where $\delta \in [-\kappa_i, \kappa_i]_{i=1}^n$. Our parameter settings differ in that we use a larger uncertainty set, with κ_i being increased fivefold (making the problem more challenging) while the number of time steps is reduced to make the constraint satisfiable. The robust algorithms are trained on this uncertainty set, where the transition dynamics are adjusted based on the transition mirror ascent algorithm of Wang & Petrik (2024), i.e. by using Monte Carlo and mirror ascent. In the experiment, this amounts to a projected update, where the projection restricts the update to lie within the uncertainty set.

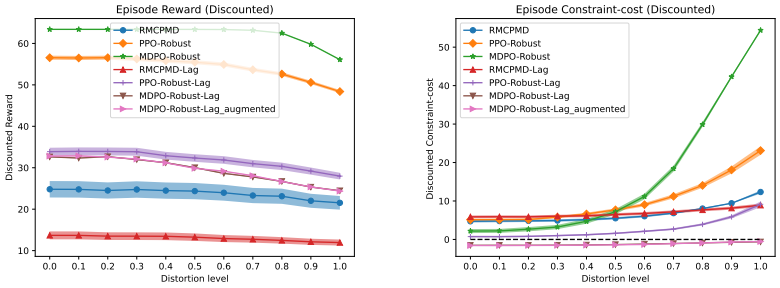
Training performance The training performance of non-robust algorithms can be found in Appendix E.1 (Figure 7). The training performance of robust methods, which use LTMA, can be found in Figure 8. A trade-off in reward vs constraint-cost can be observed; that is, the constrained algorithms put emphasis on reducing the constraint-cost while the unconstrained only maximise the reward. The plots demonstrate the rapid convergence of the MDPO-based algorithms.

Test performance Figure 1 and 2 show the test performance after 20,000 time steps depending on the distortion level. MDPO-Robust and PPO achieve the highest performance on the reward. However, since they



(a) Reward (b) Constraint-cost

Figure 1: Test performance of MDP and CMDP algorithms obtained by applying the learned determinstic policy from the Cartpole domain after 20,000 time steps of training. The line and shaded area represent the mean and standard error across the perturbations for the particular distortion level.



(a) Reward (b) Constraint-cost

Figure 2: Test performance of RMDP and RCMDP algorithms obtained by applying the learned determinstic policy from the Cartpole domain after 20,000 time steps of training. The line and shaded area represent the mean and standard error across the perturbations for the particular distortion level.

do not optimise the Lagrangian, they perform poorly on the constraint-cost. MDPO-Robust-Augmented-Lag achieves the best overall performance by providing the lowest constraint-cost by far across all distortion levels. In the test performance after 200,000 time steps (see Figure 13 and 14 of Appendix F.1), the MCPMD algorithms are able to get to similar performance levels, and all three baselearners are competitive.

To summarise the overall performance quantitatively, Table 2a shows that MDPO-Robust is the top performer on the return after 20,000 time steps. MDPO-Robust-Lag and MDPO-Lag algorithms obtain similar levels of performance on the mean penalised return statistics, although the former obtain an improved minimum score, and both are far ahead of PPO-Lag and MCPMD-Lag algorithms. In Table 2b, the performance after 200,000 time steps can be observed. MDPO-Robust-Lag, with or without augmentation, is the top performer on the mean penalised return metrics, followed by non-robust MDPO-Lag. MCPMD-Lag and RMCPMD-Lag perform highest on the minimum performance, and are followed by MDPO-Robust-Lag algorithms. In summary, the solutions converge to relatively similar test performance levels but MDPO-based algorithms are superior in terms of sample efficiency.

Table 2: Test performance on the return and penalised return statistics in the Cartpole domain based on 11 distortion levels, 16 perturbations per level, 10 seeds, and 100 evaluations per test. The mean score is the grand average over distortion levels, perturbations, seeds, and evaluations. The standard error and minimum are computed across the different environments (i.e. distortion levels and perturbations), indicating the robustness of the solution to changes in the environment. Bold indicates the top performance and any additional algorithms that are within one pooled standard error.

(a) After 20,000 time steps of training

	Return		$R^{\pm}_{\text{pen}}$ (signed)		R_{pen} (positive)	
	Mean ± SE	Min	Mean ± SE	Min	Mean ± SE	Min
MCPMD	26.7 ± 0.1	23.3	-196.6 ± 22.3	-523.9	-209.6 ± 20.9	-525.9
PPO	58.1 ± 0.1	51.3	-262.1 ± 76.8	-1121.3	-290.4 ± 72.7	-1122.3
MDPO	62.9 ± 0.1	52.4	-210.6 ± 152.3	-2015.7	-270.1 ± 144.3	-2016.0
RMCPMD	25.4 ± 0.1	22.3	-216.8 ± 22.0	-538.5	-228.2 ± 20.4	-539.0
PPO-Robust	57.1 ± 0.2	50.3	-225.3 ± 70.5	-985.3	-250.8 ± 66.9	-987.0
MDPO-Robust	**63.1** ± 0.1	**56.4**	-229.7 ± 179.6	-2610.6	-294.1 ± 171.4	-2610.8
MCPMD-Lag	16.6 ± 0.1	14.0	-247.3 ± 11.2	-383.7	-259.3 ± 10.9	-392.5
PPO-Lag	39.1 ± 0.1	33.9	2.4 ± 17.2	-217.4	-23.6 ± 15.1	-229.3
MDPO-Lag	37.0 ± 0.2	28.0	**106.1** ± 6.9	23.2	**35.7** ± 1.6	11.9
MDPO-Augmented-Lag	37.5 ± 0.2	28.3	**105.6** ± 7.2	18.4	**35.9** ± 1.7	7.2
RMCPMD-Lag	14.2 ± 0.1	12.0	-276.7 ± 10.8	-408.6	-287.7 ± 10.9	-419.5
PPO-Robust-Lag	35.5 ± 0.2	28.8	-12.2 ± 22.0	-376.9	-39.4 ± 19.8	-387.5
MDPO-Robust-Lag	34.4 ± 0.3	24.9	**108.4** ± 4.4	**65.9**	34.2 ± 1.2	**24.9**
MDPO-Robust-Augmented-Lag	34.6 ± 0.3	24.7	**110.8** ± 4.9	**61.6**	**34.6** ± 1.2	**24.7**

(b) After 200,000 time steps of training

	Return		$R^{\pm}_{\text{pen}}$ (signed)		R_{pen} (positive)	
	Mean ± SE	Min	Mean ± SE	Min	Mean ± SE	Min
MCPMD	63.0 ± 0.1	54.1	-316.3 ± 167.8	-2372.3	-364.0 ± 161.2	-2372.5
PPO	**63.1** ± 0.1	**56.4**	-252.2 ± 178.6	-2474.0	-307.2 ± 171.1	-2474.2
MDPO	62.7 ± 0.2	46.3	-96.2 ± 120.7	-1594.5	-159.5 ± 112.4	-1594.5
RMCPMD	**63.1** ± 0.1	**56.5**	-175.2 ± 181.9	-2672.5	-241.0 ± 173.6	-2672.9
PPO-Robust	62.8 ± 0.1	55.2	-191.0 ± 148.9	-2036.3	-251.0 ± 140.9	-2037.4
MDPO-Robust	**63.2** ± 0.1	56.1	-109.7 ± 176.0	-2647.1	-185.5 ± 167.1	-2648.3
MCPMD-Lag	29.7 ± 0.3	21.6	97.4 ± 3.9	**72.0**	29.7 ± 1.2	**21.6**
PPO-Lag	41.0 ± 0.2	33.8	78.4 ± 10.3	-55.5	32.0 ± 5.8	-58.2
MDPO-Lag	44.5 ± 0.2	37.8	90.9 ± 11.7	-38.9	**36.0** ± 5.5	-42.4
MDPO-Augmented-Lag	45.2 ± 0.2	38.4	86.0 ± 12.5	-54.4	**35.1** ± 6.6	-56.3
RMCPMD-Lag	29.4 ± 0.3	22.0	89.4 ± 3.7	**70.8**	29.1 ± 1.1	**21.9**
PPO-Robust-Lag	39.4 ± 0.2	32.1	84.2 ± 9.0	-28.9	32.1 ± 4.5	-35.6
MDPO-Robust-Lag	41.6 ± 0.2	32.1	**105.1** ± 9.5	9.6	**38.6** ± 2.5	2.6
MDPO-Robust-Augmented-Lag	41.2 ± 0.2	31.4	**106.2** ± 9.1	13.7	**38.6** ± 2.2	5.0

5.2 Inventory Management

A second domain is the Inventory Management domain from Wang & Petrik (2024), which involves maintaining an inventory of resources. The resources induce a period-wise cost and the goal is to purchase and sell resources such that minimal cost is incurred over time. To form a constrained variant of this benchmark, we introduce the constraint that the discounted sum of actions should not average to higher than zero (i.e. the rate of selling should not exceed that of purchasing). We use the same radial features and clipped uncertainty set as in the implementation on github `https://github.com/JerrisonWang/JMLR-DRPMD`. The implementation uses Gaussian parametric transition dynamics

$$p(s'|s,a) = \frac{1}{(2\pi)^{1/2}\sigma} e^{-\frac{1}{2\sigma}(s'-\eta(s,a)^{\intercal}\zeta(s,a))^2}, \tag{51}$$

with $\sigma = 1$, and the feature vector for $i = 1, 2$ is given by

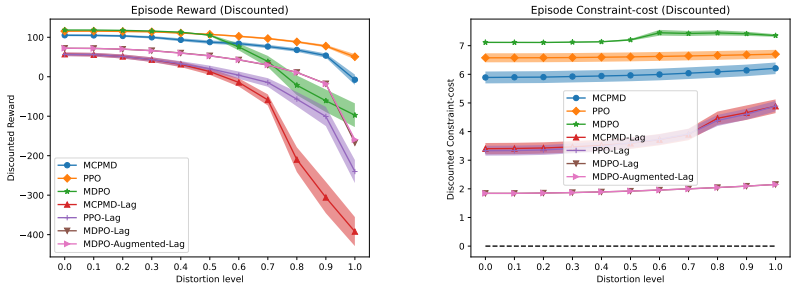
$$\zeta_i(s,a) = e^{-\frac{\left\|s-\mu_{\zeta_i,s}\right\|^2+\left\|a-\mu_{\zeta_i,a}\right\|^2}{2\sigma^2_{\zeta_i}}}, \tag{52}$$

where $\mu_{\zeta_1} = (-4, 5)$, $\mu_{\zeta_2} = (-2, 8)$. The uncertainty set is given by $\{\eta : \|\eta - \eta_c\|_\infty \leq \kappa\}$ where $\eta_c = (-2, 3.5)$. With the exception of the constraint, and the number of steps per episode for the adversary, the setting matches Wang & Petrik (2024). The constraint at time t is given by

$$c(s_t, a_t) = K_{s,a}(a_t^2 - s_t) - d, \tag{53}$$

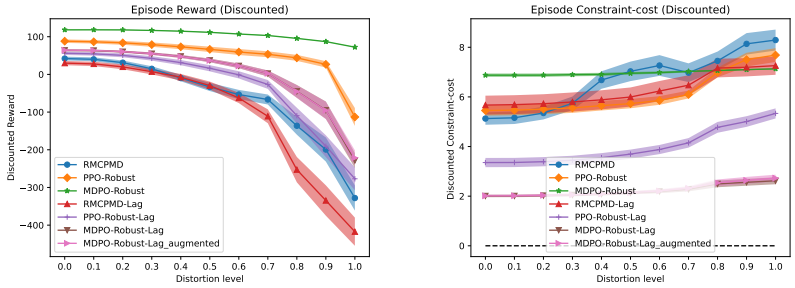
with constants set as $K_{s,a} = 0.01$ and $d = 0.0$ in the experiments. The constraint indicates a preference for actions with modest squared transaction magnitudes and a preference for having a positive inventory.

Training performance The training performance can be found in Figure 9 of Appendix E.2 for non-robust algorithms and in Figure 10 of Appendix E.2 for robust algorithms. The plots confirm the effectiveness of the constrained optimisation algorithms and the robust algorithms can also find a good solution despite the environment being adversarial. It is also clear that the MDPO based algorithms converge more rapidly to solutions of the highest quality in constrained and robust-constrained optimisation, although there appears to be a small benefit of PPO in non-robust unconstrained optimisation.



(a) Reward (b) Constraint-cost

Figure 3: Test performance of MDP and CMDP algorithms in the Inventory Management domain using the deterministic policy on the test distortions.



(a) Reward (b) Constraint-cost

Figure 4: Test performance of RMDP and RCMDP algorithms in the Inventory Management domain using the deterministic policy on the test distortions.

Test performance Figure 5 and Figure 6 visualise the results for the test performance after 16,000 time steps. MDPO-Robust obtains the highest performance on the test reward, followed closely by MDPO and PPO. The constraint is challenging to satisfy for all algorithms, but MDPO-Lag based algorithms obtain the best solutions with cost between 2 and 3 across the distortion levels. Algorithms based on PPO-Lag and MCPMD-Lag perform poorly with worst starting points and stronger performance degrading across distortion levels. After 400,000 time steps, the differences between the algorithms is relatively small in the test cases (see Figure 15 and 16 of Appendix F.2)

As can be observed in Table 3a, MDPO-Robust is the top performer on the return after 16,000 time steps. All other statistics, including the mean and minimum of the signed and positive penalised return are maximised by MDPO-Lag based algorithms. In Table 3b, the performance after the full 400,000 time steps can be observed. With the exception of PPO-Robust-Lag, all constrained algorithms converge to a similar optimum, indicating that the nominal environment and the other environments in the uncertainty set have large overlap. Similarly,

the unconstrained algorithms reach a similar optimum. In summary, the solutions converge to relatively similar test performance levels but MDPO-based algorithms are superior in terms of sample efficiency.

Table 3: Test performance on the return and penalised return statistics in the Inventory Management domain based on 11 distortion levels, 4 perturbations per level, 10 seeds, and 50 evaluations per test. The mean score is the grand average over distortion levels, perturbations, seeds, and evaluations. The standard error and minimum are computed across the different environments (i.e. distortion levels and perturbations), indicating the robustness of the solution to changes in the environment. Bold indicates the top performance and any additional algorithms that are within one pooled standard error.

(a) After 16,000 time steps of training

	Return		$R^{\pm}_{\text{pen}}$ (signed)		R_{pen} (positive)	
	Mean ± SE	Min	Mean ± SE	Min	Mean ± SE	Min
MCPMD	103.6 ± 4.0	-7.5	-2838.3 ± 12.7	-2946.2	-2838.4 ± 12.6	-2946.2
PPO	114.7 ± 2.3	50.7	-3166.9 ± 6.5	-3208.0	-3166.9 ± 6.5	-3208.0
MDPO	102.6 ± 8.0	-97.4	-3430.3 ± 21.9	-3557.6	-3430.3 ± 21.9	-3557.6
RMCPMD	63.6 ± 23.1	-328.0	-2646.9 ± 124.6	-3548.1	-2648.0 ± 123.9	-3548.1
PPO-Robust	100.7 ± 9.9	-113.0	-2673.0 ± 74.2	-3397.5	-2674.4 ± 73.4	-3397.5
MDPO-Robust	**118.0** ± 1.8	**72.3**	-3306.4 ± 9.7	-3350.0	-3306.4 ± 9.7	-3350.0
MCPMD-Lag	50.1 ± 20.2	-392.3	-1684.0 ± 35.8	-2003.1	-1686.4 ± 36.0	-2003.1
PPO-Lag	64.7 ± 13.9	-240.1	-1593.3 ± 43.0	-1878.9	-1597.2 ± 39.5	-1878.9
MDPO-Lag	66.8 ± 7.1	-167.2	**-837.3** ± 21.4	**-939.0**	**-837.9** ± 20.4	**-939.0**
MDPO-Augmented-Lag	66.9 ± 7.0	-159.4	**-837.5** ± 21.1	**-938.9**	**-838.0** ± 20.4	**-938.9**
RMCPMD-Lag	50.3 ± 27.4	-417.4	-2635.6 ± 113.0	-3159.5	-2638.3 ± 111.1	-3159.5
PPO-Robust-Lag	63.2 ± 17.1	-277.0	-1655.0 ± 67.5	-2190.2	-1660.0 ± 65.2	-2190.2
MDPO-Robust-Lag	60.7 ± 10.7	-228.5	-930.9 ± 32.3	-1118.2	-932.2 ± 30.8	-1118.2
MDPO-Robust-Augmented-Lag	60.9 ± 10.6	-221.4	-938.6 ± 34.4	-1172.2	-940.1 ± 32.9	-1172.2

(b) After 400,000 time steps of training

	Return		$R^{\pm}_{\text{pen}}$ (signed)		R_{pen} (positive)	
	Mean ± SE	Min	Mean ± SE	Min	Mean ± SE	Min
MCPMD	**119.9** ± 1.8	**72.8**	-3425.3 ± 5.5	-3460.1	-3425.3 ± 5.5	-3460.1
PPO	**120.0** ± 1.8	**73.6**	-3425.4 ± 5.4	-3459.7	-3425.4 ± 5.4	-3459.7
MDPO	**118.8** ± 2.5	32.2	-3417.0 ± 8.8	-3459.7	-3417.0 ± 8.8	-3459.7
RMCPMD	109.3 ± 2.8	26.0	-2907.7 ± 8.5	-2956.1	-2907.9 ± 8.3	-2956.1
PPO-Robust	**119.9** ± 1.8	**73.2**	-3425.3 ± 5.4	-3460.3	-3425.3 ± 5.4	-3460.3
MDPO-Robust	**119.7** ± 1.8	72.4	-3416.7 ± 4.1	-3429.7	-3416.7 ± 4.1	-3429.7
MCPMD-Lag	67.1 ± 7.0	-156.5	**-837.7** ± 20.9	**-939.0**	**-838.2** ± 20.2	**-939.0**
PPO-Lag	66.7 ± 7.1	-166.7	**-837.2** ± 21.4	**-939.0**	**-837.8** ± 20.5	**-939.0**
MDPO-Lag	66.8 ± 7.2	-168.5	**-837.3** ± 21.5	**-939.3**	**-838.0** ± 20.4	**-939.3**
MDPO-Augmented-Lag	66.9 ± 7.1	-162.5	-837.5 ± 21.2	**-939.2**	**-838.0** ± 20.4	**-939.2**
RMCPMD-Lag	66.8 ± 7.1	-165.8	**-837.4** ± 21.4	**-939.0**	**-838.1** ± 20.3	**-939.0**
PPO-Robust-Lag	60.8 ± 10.8	-234.0	-931.0 ± 32.5	-1118.3	-932.4 ± 30.9	-1118.3
MDPO-Robust-Lag	66.9 ± 7.1	-166.5	**-837.4** ± 21.4	**-939.3**	**-838.0** ± 20.4	**-939.3**
MDPO-Robust-Augmented-Lag	66.8 ± 7.1	-167.7	**-837.2** ± 21.5	**-939.1**	**-837.9** ± 20.5	**-939.1**

5.3 3-D Inventory Management

The third (and last) domain in the experiments introduces a multi-dimensional variant of the above Inventory Management problem, with dynamics

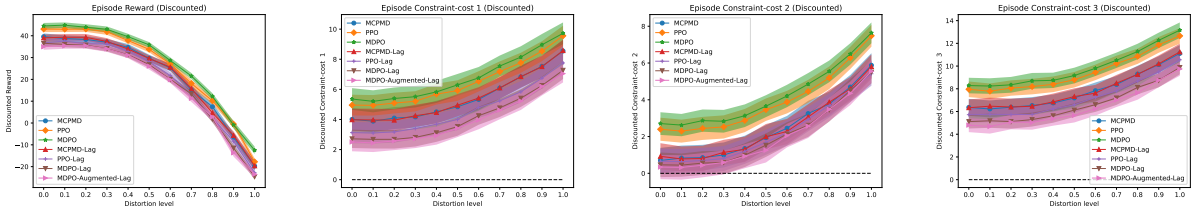
$$p(s'|s,a) = \frac{1}{(2\pi)^{n/2}\sigma} e^{-\frac{1}{2\sigma}(s'-\eta(s,a)^\intercal \zeta(s,a))^2}, \tag{54}$$

where the feature vectors of Eq. 52 are now given by $\mu_{\zeta_1} = (-3, -2.5, -3.5, 5.0, 2.0, 4.5)$, $\mu_{\zeta_2} = (-6.0, -2.8, -4.0, 8.0, 2.0, 2.5)$, $\sigma_\zeta = (4, 4.5)$, and the uncertainty set is given by $\{\eta : \|\eta - \eta_c\|_\infty \leq \kappa\}$ where $\eta_c = [[-2, 2.5], [-1.8, 1.5], [-1.5, 2.0]]$ and $\kappa = 0.5$. The constraint $j \in [m]$ at time t is given by

$$c_j(s_t, a_t) = a_{t,j} - K_s s_{t,j} - d\,, \tag{55}$$

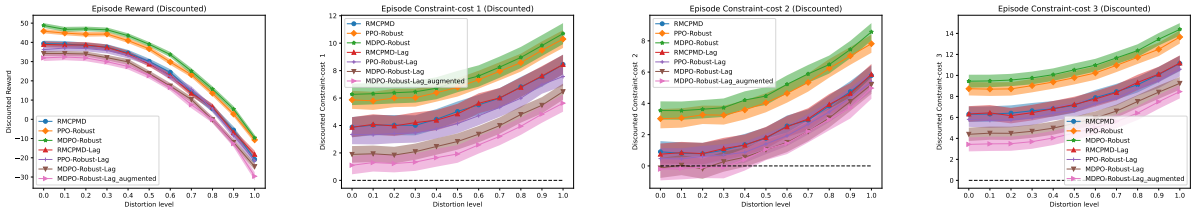
where $a_{t,j}$ and $s_{t,j}$ denote the j'th dimension of the state and action, respectively, at time t, and constants are set as $K_s = 0.5$ and $d = 0.0$ in the experiments. The constraint indicates that the agent should not get a negative inventory and if the inventory is positive, it should not sell more than 50% of the current inventory.

Training performance The training performance can be found in Figure 11 and Figure 12 of Appendix E.3. It is clear that the MDPO-based algorithm have improved sample-efficiency and converge to the highest unconstrained and constrained performances.



(a) Reward (b) Constraint-cost 1 (c) Constraint-cost 2 (d) Constraint-cost 3

Figure 5: Test performance of MDP and CMDP algorithms in the 3-D Inventory Management domain using the deterministic policy on the test distortions.



(a) Reward (b) Constraint-cost 1 (c) Constraint-cost 2 (d) Constraint-cost 3

Figure 6: Test performance of RMDP and RCMDP algorithms in the 3-D Inventory Management domain using the deterministic policy on the test distortions.

Test performance Figure 5 and Figure 6 visualise the results for the test performance after 40,000 time steps. While all algorithms are sensitive to the distortion level, it can be seen that the robust algorithms have are generally shifted up for the rewards, and robust-constrained algorithms have been shifted down for the constraint-costs, indicating the effectiveness of robustness training. The MDPO-Robust algorithm is superior in the return and the the MDPO-Robust-Lag algorithms are found to be superior in the constraint-cost. While it is challenging to satisfy the constraint with limited training, after 400,000 time all algorithms have further improved, but remarkably MDPO-Robust-Augmented-Lag can satisfy all the constraints for all but the highest distortion levels (see Figure 17 and Figure 18 of Appendix F.3).

Table 4: Test performance on the return and penalised return statistics in the 3-D Inventory Management domain based on 11 distortion levels, 64 perturbations per level, 10 seeds, and 10 evaluations per test. The standard error and minimum are computed across the different environments (i.e. distortion levels and perturbations), indicating the robustness of the solution to changes in the environment. Bold indicates the top performance and any additional algorithms that are within one pooled standard error.

(a) After 40,000 time steps

	Return		$R^{\pm}_{\text{pen}}$ (signed)		R_{pen} (positive)	
	Mean ± SE	Min	Mean ± SE	Min	Mean ± SE	Min
MCPMD	46.2 ± 0.5	-19.7	-3810.9 ± 448.7	-10322.4	-7143.7 ± 250.0	-11158.3
PPO	50.6 ± 0.4	-17.8	-5909.8 ± 458.0	-12737.1	-7957.4 ± 286.0	-12855.0
MDPO	51.6 ± 0.4	-12.5	-6415.1 ± 463.7	-13210.2	-8389.6 ± 300.0	-13384.9
RMCPMD	46.3 ± 0.5	-20.8	-3817.4 ± 449.2	-10449.0	-7146.4 ± 253.1	-11228.8
PPO-Robust	53.2 ± 0.4	-10.7	-7164.3 ± 453.6	-13926.4	-8652.0 ± 316.2	-14027.1
MDPO-Robust	**54.9** ± 0.4	**-9.6**	-7956.9 ± 457.5	-14747.8	-9376.8 ± 330.2	-14836.3
MCPMD-Lag	46.3 ± 0.5	-19.6	-3822.8 ± 452.0	-10390.2	-7149.5 ± 253.6	-11185.8
PPO-Lag	44.7 ± 0.5	-22.8	-3038.1 ± 448.7	-9467.8	-6515.6 ± 238.5	-10277.8
MDPO-Lag	43.4 ± 0.5	-24.6	-2391.4 ± 447.4	-8762.9	-5530.5 ± 237.8	-9481.2
MDPO-Augmented-Lag	42.8 ± 0.5	-23.5	-2126.5 ± 446.1	-8385.7	-5460.9 ± 229.5	-9142.2
RMCPMD-Lag	46.2 ± 0.5	-18.2	-3811.7 ± 451.5	-10399.7	-7144.5 ± 253.5	-11239.9
PPO-Robust-Lag	44.7 ± 0.5	-21.5	-3014.6 ± 445.4	-9428.9	-6494.1 ± 238.7	-10268.1
MDPO-Robust-Lag	41.4 ± 0.5	-24.6	-1416.9 ± 446.1	-7726.9	**-5088.9** ± 221.7	-8560.6
MDPO-Robust-Augmented-Lag	39.5 ± 0.5	-29.6	**-525.0** ± 441.2	**-6861.9**	**-4957.0** ± 211.2	**-8289.3**

(b) After 400,000 time steps

	Return		$R^{\pm}_{\text{pen}}$ (signed)		R_{pen} (positive)	
	Mean ± SE	Min	Mean ± SE	Min	Mean ± SE	Min
MCPMD	46.2 ± 0.5	-21.3	-3810.9 ± 448.9	-10358.5	-7147.0 ± 251.9	-11105.7
PPO	59.5 ± 0.4	-5.9	-10122.1 ± 462.4	-16777.6	-10886.2 ± 370.4	-16823.6
MDPO	81.2 ± 0.4	18.6	-20577.2 ± 535.4	-28669.4	-20769.1 ± 504.0	-28670.9
RMCPMD	46.3 ± 0.5	-18.9	-3817.2 ± 450.6	-10514.2	-7148.9 ± 252.9	-11353.5
PPO-Robust	60.9 ± 0.4	-2.4	-10802.7 ± 443.3	-17484.1	-11429.2 ± 367.4	-17506.1
MDPO-Robust	**87.0** ± 0.4	**24.4**	-23312.6 ± 525.6	-31314.1	-23459.7 ± 499.3	-31314.1
MCPMD-Lag	46.2 ± 0.5	-22.0	-3813.8 ± 453.1	-10509.0	-7147.1 ± 254.3	-11366.5
PPO-Lag	34.2 ± 0.5	-36.5	1965.9 ± 416.5	-3963.9	-3126.5 ± 214.3	-7007.9
MDPO-Lag	34.9 ± 0.5	-32.5	1566.5 ± 420.9	-4406.7	**-1180.5** ± 229.6	-5420.4
MDPO-Augmented-Lag	24.6 ± 0.5	-44.1	**6421.1** ± 369.2	**1517.0**	**-1139.5** ± 175.7	-4564.8
RMCPMD-Lag	46.2 ± 0.5	-20.8	-3813.1 ± 451.5	-10594.7	-7146.9 ± 252.7	-11310.6
PPO-Robust-Lag	31.0 ± 0.5	-34.8	3460.0 ± 419.6	-2398.7	-2285.9 ± 167.0	-5397.1
MDPO-Robust-Lag	32.8 ± 0.5	-32.4	2617.9 ± 378.6	-2601.7	**-1011.4** ± 169.9	**-3767.4**
MDPO-Robust-Augmented-Lag	26.8 ± 0.4	-38.6	5312.6 ± 340.2	758.7	-1219.2 ± 150.4	**-3867.7**

The return and penalised return statistics after 40,000 time steps can be observed in Table 4a. MDPO-Robust has the highest score on the mean and minimum of the test return, which indicates the robustness training was succesful in guaranteeing high levels of performance across the uncertainty set. MDPO-Robust-Augmented-Lag outperforms all other algorithms on the signed and positive penalised return statistics. MDPO-Robust-Lag follows closely in mean and minimum performance on the positive penalised return. After 400,000 time steps, MDPO-Robust-Lag algorithms have by far the highest minimum penalised return while MDPO algorithms with augmented Lagrangian score remarkably well on the signed penalised return (see Table 4b). Overall, the data indicate the effectiveness for MDPO-based algorithms, and particularly the effectiveness of MDPO-Robust-Lag algorithms for robust constrained optimisation.

6 Conclusion

This paper presents mirror descent policy optimisation for robust constrained MDPs (RCMDPs), making use of policy gradient techniques to optimise both the policy and the transition kernel (as an adversary) on the Lagrangian representing a constrained MDP. In the oracle-based RCMDP setting, we confirm it is indeed possible to obtain guarantees similar to those of traditional MDPs, with an $\mathcal{O}\left(\frac{1}{T}\right)$ convergence rate

for the squared distance as a Bregman divergence, and an $\mathcal{O}\left(e^{-T}\right)$ convergence rate for entropy-regularised objectives. In the sample-based setting, we require $\tilde{\mathcal{O}}(\epsilon^{-3})$ samples for an average regret of at most ϵ, confirming an $\tilde{\mathcal{O}}\left(\frac{1}{T^{1/3}}\right)$ convergence rate. Experiments confirm the performance benefits of mirror descent policy optimisation in practice, obtaining significant improvements in test performance.

References

Mohammed Amin Abdullah, Hang Ren, Haitham Bou Ammar, Vladimir Milenkovic, Rui Luo, Mingtian Zhang, and Jun Wang. Wasserstein Robust Reinforcement Learning. *arXiv preprint arXiv:1907.13196*, 2019.

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of the International Conference on Machine Learning (ICML 2017)*, pp. 30–47, 2017. ISBN 9781510855144.

Alekh Agarwal, Sham M. Kakade, Jason D. Lee, and Gaurav Mahajan. On the theory of policy gradient methods: Optimality, approximation, and distribution shift. *Journal of Machine Learning Research*, 22, 2021.

Eitan Altman. *Constrained Markov Decision Processses*. CRC Press, 1999.

Andrew G. Barto, Richard S. Sutton, and Charles W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13(5): 834–846, 1983. doi: 10.1109/TSMC.1983.6313077.

Amir Beck and Marc Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31:167–175, 2003.

David M. Bossens. Robust Lagrangian and Adversarial Policy Gradient for Robust Constrained Markov Decision Processes. In *IEEE Conference on Artificial intelligence (CAI 2024)*, 2024.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym. *arXiv preprint arXiv:1606.01540*, pp. 1–4, 2016.

Shicong Cen, Chen Cheng, Yuxin Chen, Yuting Wei, and Yuejie Chi. Fast Global Convergence of Natural Policy Gradient Methods with Entropy Regularization. *Operations Research*, 70(4):2563–2578, 2022. ISSN 15265463. doi: 10.1287/opre.2021.2151.

Dongsheng Ding, Kaiqing Zhang, Tamer Basar, and Mihailo R. Jovanovic. Natural policy gradient primal-dual method for constrained Markov decision processes. In *Advances in Neural Information Processing Systems (NeurIPS 2020)*, 2020.

Chin Pang Ho, Marek Petrik, and Wolfram Wiesemann. Partial policy iteration for L1-Robust Markov decision processes. *Journal of Machine Learning Research*, 22:1–46, 2021.

Linfang Hou, Liang Pang, Xin Hong, Yanyan Lan, Zhiming Ma, and Dawei Yin. Robust Reinforcement Learning with Wasserstein Constraint. *arXiv preprint arXiv:2006.00945v1*, 2020.

Garud N. Iyengar. Robust dynamic programming. *Mathematics of Operations Research*, 30(2):257–280, 2005. ISSN 0364765X. doi: 10.1287/moor.1040.0129.

Yufei Kuang, Miao Lu, Jie Wang, Qi Zhou, Bin Li, and Houqiang Li. Learning Robust Policy against Disturbance in Transition Dynamics via State-Conservative Policy Optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence, AAAI 2022*, pp. 7247–7254, 2022. doi: 10.1609/aaai.v36i7.20686.

Navdeep Kumar, Matthieu Geist, Kfir Levy, Esther Derman, and Shie Mannor. Policy Gradient for Rectangular Robust Markov Decision Processes. In *Advances in Neural Information Processing Systems (NeurIPS 2023)*, 2023.

Mengmeng Li, Tobias Sutter, and Daniel Kuhn. Policy Gradient Algorithms for Robust MDPs with Non-Rectangular Uncertainty Sets. In *Advances in Neural Information Processing Systems (NeurIPS 2023)*, pp. 1–31, 2023.

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. *arXiv preprint arXiv:1509.02971*, pp. 1–127, 2015.

Tao Liu, Ruida Zhou, Dileep Kalathil, P. R. Kumar, and Chao Tian. Policy Optimization for Constrained MDPs with Provable Fast Global Convergence. *arXiv preprint arXiv:2111.00552*, 2021.

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. *arXiv preprint arXiv:2010.10644*, pp. 1–23, 2020.

Jincheng Mei, Chenjun Xiao, Csaba Szepesvari, and Dale Schuurmans. On the global convergence rates of softmax policy gradient methods. In *International Conference on Machine Learning (ICML 2020)*, pp. 6776–6785, 2020.

Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous Methods for Deep Reinforcement Learning. In *Proceedings of the International Conference on Machine Learning (ICML 2016)*, New York, NY, USA, 2016. doi: 10.1177/0956797613514093.

Arnab Nilim and Laurent El Ghaoui. Robust control of Markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005. doi: 10.1287/opre.1050.0216.

Santiago Paternain, Miguel Calvo-Fullana, Luiz F.O. Chamon, and Alejandro Ribeiro. Safe Policies for Reinforcement Learning via Primal-Dual Methods. *IEEE Transactions on Automatic Control*, 68(3): 1321–1336, 2023. doi: 10.1109/TAC.2022.3152724.

Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking Safe Exploration in Deep Reinforcement Learning. *OpenAI*, pp. 1–6, 2019. URL `https://cdn.openai.com/safexp-short.pdf`.

Reazul Hasan Russel, Mouhacine Benosman, and Jeroen Van Baar. Robust Constrained-MDPs: Soft-Constrained Robust Policy Optimization under Model Uncertainty. *arXiv preprint arXiv:2010.04870*, 2020.

Reazul Hasan Russel, Mouhacine Benosman, Jeroen Van Baar, and Radu Corcodel. *Lyapunov Robust Constrained-MDPs: Soft-Constrained Robustly Stable Policy Optimization under Model Uncertainty*, pp. 307–328. Springer International Publishing, 2023.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *Proceedings of the International Conference on Learning Representations (ICLR 2018)*, pp. 1–14, 2018.

Zhongchang Sun, Sihong He, Fei Miao, and Shaofeng Zou. Constrained Reinforcement Learning Under Model Mismatch. In *Proceedings of the International Conference on Machine Learning (ICML 2024)*, pp. 47017–47032, 2024.

Chen Tessler, Daniel J. Mankowitz, and Shie Mannor. Reward constrained policy optimization. In *Proceedings of the International Conference on Learning Representations (ICLR 2019)*, pp. 1–15, 2019.

Manan Tomar, Lior Shani, Yonathan Efroni, and Mohammad Ghavamzadeh. Mirror Descent Policy Optimization. In *International Conference on Learning Representations (ICLR 2022)*, pp. 1–24, 2022.

Qiuhao Wang and Marek Petrik. Policy Gradient for Robust Markov Decision Processes. *arXiv preprint arXiv:2410.22114v2*, pp. 1–59, 2024.

Qiuhao Wang, Chin Pang Ho, and Marek Petrik. Policy Gradient in Robust MDPs with Global Convergence Guarantee. In *Proceedings of the International Conference on Machine Learning (ICML 2023)*, pp. 35763–35797, 2023.

Yue Wang, Fei Miao, and Shaofeng Zou. Robust Constrained Reinforcement Learning. *arXiv preprint arXiv:2209.06866*, 2022.

Zifan Wu, Bo Tang, Qian Lin, Chao Yu, Shangqin Mao, Qianlong Xie, Xingxing Wang, and Dong Wang. Off-Policy Primal-Dual Safe Reinforcement Learning. In *Proceedings of the International Conference on Learning Representations (ICLR 2024)*, pp. 1–20, 2024.

Lin Xiao. On the Convergence Rates of Policy Gradient Methods. *Journal of Machine Learning Research*, 23: 1–36, 2022.

Tengyu Xu, Yingbin Lang, and Guanghui Lan. CRPO: A New Approach for Safe Reinforcement Learning with Convergence Guarantee. In *Proceedings of the International Conference on Machine Learning (ICML 2021)*, pp. 11480–11491, 2021.

Tsung Yen Yang, Justinian Rosca, Karthik Narasimhan, and Peter J. Ramadge. Projection-Based Constrained Policy Optimization. In *8th International Conference on Learning Representations, ICLR 2020*, pp. 1–24, 2020.

Donghao Ying, Mengzi Amy Guo, Yuhao Ding, Javad Lavaei, and Zuo Jun Shen. Policy-Based Primal-Dual Methods for Convex Constrained Markov Decision Processes. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI 2023)*, pp. 10963–10971, 2023. doi: 10.1609/aaai.v37i9.26299.

Wenhao Zhan, Shicong Cen, Baihe Huang, Yuxin Chen, Jason D. Lee, and Yuejie Chi. Policy Mirror Descent for Regularized Reinforcement Learning: a Generalized Framework With Linear Convergence. *SIAM Journal on Optimization*, 33(2):1061–1091, 2023. doi: 10.1137/21M1456789.

Xuezhou Zhang, Yiding Chen, Jerry Zhu, and Wen Sun. Robust Policy Gradient against Strong Data Corruption. In *Proceedings of the International Conference on Machine Learning (ICML 2021)*, pp. 12391–12401, 2021.

Zhengfei Zhang, Kishan Panaganti, Laixi Shi, Yanan Sui, Adam Wierman, and Yisong Yue. Distributionally Robust Constrained Reinforcement Learning under Strong Duality. *arXiv preprint arXiv:2406.15788*, 2024.

Ruida Zhou, Tao Liu, Min Cheng, Dileep Kalathil, P. R. Kumar, and Chao Tian. Natural Actor-Critic for Robust Reinforcement Learning with Function Approximation. In *Advances in Neural Information Processing Systems (NeurIPS 2023)*, 2023.

A Bregman divergence and associated policy definitions

Definition 3. ***Bregman divergence.*** *Let $h : \Delta(A) \to \mathbb{R}$ be a convex and differentiable function. We define*

$$B(x, y; h) := h(x) - h(y) - \langle \nabla h(y), x - y \rangle \tag{56}$$

as the Bregman divergence with distance-generating function h.

The Bregman divergence represents the distance between the first-order Taylor expansion and the actual function value, intuitively representing the strength of the convexity. A list of common distance-generating functions and their associated Bregman divergences is given in Table 5.

Table 5: Bregman divergence for common distance-generating functions.

Distance-generating function (h)	**Bregman divergence** $(B(\cdot, \cdot; h))$
ℓ_1-norm $\|p\|_1$	0 for $x, y \in \mathbb{R}^d_+$.
Squared ℓ_2-norm $\frac{1}{2}\|x\|_2^2$	Squared Euclidian distance $D_{\mathrm{SE}}(x, y) = \frac{1}{2}\|x - y\|_2^2$ for $x, y \in \mathbb{R}^d$
Negative entropy $\sum_i p(i) \log(p(i))$	Kullbach-Leibler divergence $D_{\mathrm{KL}}(p, q) = \sum_i p(i) \log(p(i)/q(i))$ for $p, q \in \Delta$

Table 6 and Table 7 show the update rules for different policy and transition kernel parametrisations. They hold true for any value function as well as Lagrangian value functions, so the tables omit the bold for generality purposes.

Table 6: Update rules for different policy parametrisations and Bregman divergences.

Parametrisation	**Bregman divergence**	**Update rule**
Direct: $\pi = \theta$	$D_{\mathrm{SE}}(\theta, \theta_t)$	$\theta_{t+1} \leftarrow \mathrm{proj}_{\Delta(\mathcal{A})}(\theta_t - \eta_t \nabla_\theta V_{\pi,p}(d_0))$
Softmax: $\pi(a\|s) = \frac{\exp(\theta_{s,a})}{\sum_{a'} \exp(\theta_{s,a'})}$	$D_{\mathrm{SE}}(\theta, \theta_t)$	$\theta_{t+1} \leftarrow \theta_t - \eta_t \nabla_\theta V_{\pi,p}(d_0)$
Softmax: $\pi(a\|s) = \frac{\exp(\theta_{s,a})}{\sum_{a'} \exp(\theta_{s,a'})}$	Occupancy-weighted KL-divergence (Eq. 13)	$\pi_k^{t+1}(a\|s) = \frac{1}{Z_k^t(s)}(\pi_k^t(a\|s))^{1-\frac{\eta\alpha}{1-\gamma}} e^{\frac{-\eta \mathbf{Q}^{\alpha}_{\pi_k^t, p}(s,a)}{1-\gamma}}$

Table 7: Update rules and uncertainty sets for different parametric transition kernels (PTKs). The notation $D(\cdot, \cdot)$ indicates a particular distance function such as ℓ_1 or ℓ_∞-norm and the $\bar{x}$ notation indicates the nominal model for any parameter x (typically obtained from parameter estimates).

Parametrisation	**Update rule**	**Uncertainty sets**
Entropy PTK: $p(s'\|s,a) = \frac{\bar{p}(s'\|s,a) \exp\left(\frac{\zeta^\intercal \phi(s')}{\lambda^{\prime\intercal} \varphi(s,a)}\right)}{\sum_{s''} \bar{p}(s''\|s,a) \exp\left(\frac{\zeta^\intercal \phi(s'')}{\lambda^{\prime\intercal} \varphi(s,a)}\right)}$	$\xi_{t+1} \leftarrow \arg\max_\xi \{\langle \eta_t \nabla_\xi V_{\pi,p}(d_0), \xi \rangle - D(\xi, \xi_t)\}$	$\mathcal{U}_\xi = \{\xi : D(\xi, \bar{\xi}) \le \kappa_\xi\}$.
Gaussian mixture PTK: $p(s'\|s,a) = \sum_{m=1}^M \omega_m \mathcal{N}(\eta^\intercal \zeta(s,a))$	$\xi_{t+1} \leftarrow \arg\max_\xi \{\langle \eta_t \nabla_\xi V_{\pi,p}(d_0), \xi \rangle - D(\xi, \xi_t)\}$	$\mathcal{U}_\eta = \{\eta : \forall m \in [M]\, D(\eta, \eta_m^-) \le \kappa_\eta\}$ $\mathcal{U}_\omega = \{\omega \in \Delta^M : \forall m \in [M]\, \omega_m \in [F_{\omega_m}^{-1}(\delta/2), F_{\omega_m}^{-1}(1-\delta/2)]\}$

B Supporting lemmas for traditional MDPs

The proof relies on the following continuity and smoothness conditions.

Definition 4 (Continuity and smoothness)**.** *A function $f : \Theta \to \mathbb{R}$ is L_θ-**Lipschitz continuous** if $\|f(\theta) - f(\theta')\| \le L_\theta \|\theta - \theta'\|$. f is l_θ-**smooth** if its gradients are l_θ-Lipschitz continuous, i.e. $\|\nabla f(\theta) - \nabla f(\theta')\| \le l_\theta \|\theta - \theta'\|$ or, equivalently,*

$$|f(\theta') - f(\theta) - \langle \nabla f(\theta), \theta' - \theta \rangle| \le \frac{l_\theta}{2} \|\theta - \theta'\|_2^2 .$$

B.1 Softmax policy gradient with unregularised objective

Mei et al. (2020) previously used above smoothness and continuity to prove the convergence rate for softmax policies on the unregularised objective. In particular, Theorem 4 in Mei et al. (2020) provides convergence rate results for softmax policies. We rephrase the theorem in a slightly more general manner based on the relation between the learning rate η and the smoothness coefficient l_θ, allowing easy reuse for our setting.

Theorem 5 (Convergence rate for softmax policies, Theorem 4 in Mei et al. (2020) rephrased)**.** *Let p be the transition dynamics, let costs range in $[0,1]$, let $\{\theta_t\}_{t\geq 1}$ be a sequence generated by the gradient descent update over logits according to*

$$\theta_{t+1} = \theta_t - \eta \nabla V_{\pi_{\theta_t},p}(\rho)\,, \tag{57}$$

with $\eta = 1/l_\theta$, and let $U_p = \inf_{s\in\mathcal{S},t\geq 1} \pi_\theta(a^*(s|p)|s) > 0$. Then, for all $t \geq 1$, the regret is bounded by

$$V_{\pi_{\theta_t},p}(\rho) - V_{\pi_{\theta^*},p}(\rho) \leq \frac{2SM_p(\pi^*)^2}{U_p\eta(1-\gamma)^3 t} \left\| \frac{1}{\mu} \right\|_\infty , \tag{58}$$

where $M_p(\pi^*) = \left\| \frac{d_\mu^{\pi_\tau^*,p}}{\mu} \right\|_\infty$.

B.2 Softmax policy gradient with regularised objective

Mei et al. (2020) previously used the above smoothness and continuity to prove the convergence rate for softmax policies on an entropy-regularised objective. Below we summarise their key results that factor into our analysis.

Theorem 6 in Mei et al. (2020) provides convergence rate results for softmax policies with entropy regularisation, a result which can be reused in our setting with a few parameter changes. We rephrase the theorem in slightly more general manner based on the relation between the learning rate η and the smoothness coefficient l_θ.

Theorem 6 (Convergence rate for softmax policies with entropy regularisation, Theorem 6 in Mei et al. (2020) rephrased)**.** *Let p be the transition dynamics, let $\{\theta_t\}_{t\geq 1}$ be a sequence generated by regularised policy gradient over logits according to*

$$\theta_{t+1} = \theta_t - \eta \nabla V^\tau_{\pi_{\theta_t},p}(\rho)\,, \tag{59}$$

with $\eta = 1/l_\theta$, let the initial state be sampled from μ, and let $U_p = \inf_{s\in\mathcal{S},t\geq 1} \pi_{\theta_t}(a^*(s|p)|s) > 0$. Then, for all $t \geq 1$, the regret on the regularised objective is bounded by

$$V^\tau_{\pi_{\theta_t},p}(\rho) - V^\tau_{\pi_\tau^*,p}(\rho) \leq \left\| \frac{1}{\mu} \right\|_\infty \frac{1 + \tau\log(A)}{(1-\gamma)^2} e^{-C\,(t-1)}\,, \tag{60}$$

where $C = \frac{\eta}{S}\min_s \mu(s) U_p^2 (M_p(\pi_\tau^*))^{-1}$ is independent of the iteration t.

C Supporting lemmas for sample-based analysis

C.1 Analysis of the multipliers

To analyse the regret in terms of the value directly, the lemma below provides an equivalence between the unconstrained value and the Lagrangian value at the optimum.

Lemma 11 (Complementary slackness)**.** *For any CMDP and any $j \in [m]$, the optimal constrained solution (π^*, λ^*) has either $\lambda_j^* = 0$ or $V_{\pi^*}^j(\rho) = 0$, such that its Lagrangian value is equal to its unconstrained value:*

$$\mathbf{V}_{\pi^*}(\rho;\lambda^*) = V_{\pi^*}(\rho)\,. \tag{61}$$

Lemma 12 (Bounds on the multipliers)**.** *The sequence of multipliers produced by Robust PMD-PD $\{\lambda_{k,j}\}_{k\geq 0}$, $j \in [m]$ satisfy the following properties:*

1. *Non-negativity: for any macro step $k \geq 0$, $\lambda_{k,j} \geq 0$ for all $j \in [m]$.*

2. *Positive modified multiplier: for any macro step* $k \geq 0$, $\lambda_{k,j} + \eta_\lambda \hat{V}^j_{\pi_k,p_k}(\rho) \geq 0$ *for all* $j \in [m]$.

3. *Bounded initial multiplier: for macro step* 0, $|\lambda_{0,j}|^2 \leq |\eta_\lambda \hat{V}^j_{\pi_k,p_k}(\rho)|^2$ *for all* $j \in [m]$

4. *Bounded value: for macro step* $k > 0$, $|\lambda_{k,j}|^2 \geq |\eta_\lambda \hat{V}^j_{\pi_k,p_k}(\rho)|^2$ *for all* $j \in [m]$

Proof. 1) Note that $\lambda_{0,j} \geq 0$ is trivially satisfied by the initialisation in Algorithm 1. Via induction, if $\lambda_{k,j} \geq 0$ note that if $\hat{V}_{\pi_{k+1},p_{k+1}}(\rho) \geq 0$ then $\lambda_k, j + \eta_\lambda \hat{V}_{\pi_{k+1},p_{k+1}}(\rho) \geq 0$; if it is negative, then $-\eta_\lambda \hat{V}_{\pi_{k+1},p_{k+1}}(\rho) \geq 0$.
2) Note that $0 \leq \lambda_{k,j} = \max\{\lambda_{k-1,j} + \eta_\lambda \hat{V}_{\pi_k,p_k}(\rho), -\eta_\lambda \hat{V}_{\pi_k,p_k}(\rho)\}$. This implies either a) $\eta_\lambda \hat{V}_{\pi_k,p_k}(\rho) \geq \lambda_{k-1,j} \geq 0$ or b) $\hat{V}_{\pi_k,p_k}(\rho) \leq 0$. In case a),

$$\lambda_{k,j} + \eta_\lambda \hat{V}^j_{\pi_k,p_k}(\rho) \geq 0\,.$$

In case b),

$$\lambda_{k,j} = -\eta_\lambda \hat{V}^j_{\pi_k,p_k} \geq 0\,.$$

3) This follows directly from the initialisation to $\lambda_{0,j} = \max\{0, -\eta_\lambda \hat{V}^j_{\pi_0,p_0}(\rho)\}$.
4) Note that

$$\begin{aligned}
|\lambda_{k,j}|^2 &= |\max\{\lambda_{k-1,j} + \eta_\lambda \hat{V}^j_{\pi_k,p_k}(\rho), -\eta_\lambda \hat{V}^j_{\pi_k,p_k}(\rho)\}|^2 \\
&= \max\{|\lambda_{k-1,j} + \eta_\lambda \hat{V}^j_{\pi_k,p_k}|^2, |\eta_\lambda \hat{V}^j_{\pi_k,p_k}|^2\} \\
&\geq |\eta_\lambda \hat{V}^j_{\pi_k,p_k}(\rho)|^2\,.
\end{aligned}$$

□

The analysis of the dual variables focuses on the inner product between the modified Lagrangian multiplier and the constraint-costs, which represents the total constraint-penalty at the end of an iteration. Due to the approximation errors $\epsilon_k = \hat{V}^{1:m}_{\pi_k,p_k}(\rho) - V^{1:m}_{\pi_k,p_k}(\rho)$, the inner product can be written as

$$\left\langle \lambda_k + \eta_\lambda \hat{V}^{1:m}_{\pi_k,p_k}(\rho), V^{1:m}_{\pi^{t_k}_k,p_k}(\rho) \right\rangle = \left\langle \lambda_k, \hat{V}^{1:m}_{\pi^{t_k}_k,p_{k+1}}(\rho) \right\rangle + \langle \lambda_k, -\epsilon_k \rangle + \left\langle \eta_\lambda V^{1:m}_{\pi_k,p_k}(\rho), V^{1:m}_{\pi^{t_k}_k,p_{k+1}}(\rho) \right\rangle + \left\langle \eta_\lambda \epsilon_k, V^{1:m}_{\pi^{t_k}_k,p_{k+1}}(\rho) \right\rangle . \tag{62}$$

The inner product can be lower bounded, which leads to a somewhat complex summation but a useful one that can be telescoped in the average regret analysis.

Lemma 13 (Lower bound on the inner product, Eq. 41 in Liu et al. (2021))**.** *For any* $k = 0, 1, \ldots, K-1$,

$$\begin{aligned}
&\left\langle \lambda_k + \eta_\lambda \hat{V}^{1:m}_{\pi_k,p_k}(\rho), \hat{V}^{1:m}_{\pi_{k+1},p_{k+1}}(\rho) \right\rangle \\
&\geq \frac{1}{2\eta_\lambda}\left(\|\lambda_{k+1}\|^2 - \|\lambda_k\|^2\right) + \frac{\eta}{2}\left(\left\|V^{1:m}_{\pi_k,p_k}(\rho)\right\|^2 - \left\|V^{1:m}_{\pi_{k+1},p_{k+1}}(\rho)\right\|^2\right) \\
&+ \langle \lambda_k, -\epsilon_{k-1} \rangle + \|\epsilon_{k+1}\|^2 + \eta_\lambda \left\langle \epsilon_k, V^{i:m}_{\pi_k,p_k}(\rho) \right\rangle - \eta_\lambda \|\epsilon_k\| - 2\eta_\lambda \left\langle V^{i:m}_{\pi_{k+1},p_{k+1}}(\rho), \epsilon_{k+1} \right\rangle - \frac{\gamma^2 \eta_\lambda}{(1-\gamma)^4} B_{d_\rho^{\pi_{k+1},p_{k+1}}}(\pi_{k+1}, \pi_k)\,.
\end{aligned}$$

C.2 Analysis of the Bregman divergence

Another essential part of the regret analysis is the pushback property, which allows a telescoping sum.

Lemma 14 (Pushback property, Lemma 2 in Liu et al. (2021))**.** *If* $x^* = \arg\min_{x \in \Delta} f(x) + B(x, y; h)$ *for a fixed* $y \in$ *Int*(Δ)*, then for* $\alpha > 0$ *and any* $z \in \Delta$,

$$f(x^*) + B(x^*, y; h) \leq f(x) + \alpha\left(B(z, y; h) - B(z, x^*; h)\right).$$

By selecting $z = \pi$, $y = \pi^*_k$, and $x^* = \pi^*$, and using the weighted Bregman divergence, it is possible to use the property in the context of softmax policies, which are always in the interior of the probability simplex.

Lemma 15 (Pushback property for softmax policies, Lemma 10 in Liu et al. (2021)). *For any softmax policy π and any p,*

$$\mathbf{V}_{\pi_k^*,p}(\rho) + \frac{\alpha}{1-\gamma} B_{d_\rho^{\pi_k^*,p}}(x^*, y) \leq \mathbf{V}_{\pi,p}(\rho) + \alpha \left(B_{d_\rho^{\pi,p}}(\pi, \pi_k) - B_{d_\rho^{\pi,p}}(\pi, \pi_k^*) \right).$$

The weighted Bregman divergence under the generating function $\sum_i p(i)\log(p(i))$ is equivalent to the KL-divergence, which is bounded by $\log(A)$ as shown below by noting that the uniform policy is the maximum entropy policy.

Lemma 16 (Bound on Bregman divergence). *Let π_0 be a uniform policy, then for any policy $\pi \in \Pi$ and any $p \in \mathcal{P}$,*

$$B_{d_\rho^{\pi,p}}(\pi, \pi_0) = \sum_{s\in\mathcal{S}} d_\rho^{\pi,p}(s) \sum_{a\in\mathcal{A}} \pi(a|s)\log(A\pi(a|s)) \leq \log(A).$$

C.3 Convergence of approximate entropy-regularised NPG

Below is the supporting convergence result for approximate entropy-regularised NPG.

Lemma 17 (Convergence of approximate entropy-regularised NPG, Theorem 2 of Cen et al. (2022)). *Let $\epsilon > 0$. Then if $\left\| \hat{\mathbf{Q}}^\alpha_{\pi_k^t,p} - \mathbf{Q}^\alpha_{\pi_k^t,p} \right\|_\infty \leq \epsilon$,*

$$C_k \geq \left\| \mathbf{Q}^\alpha_{\pi_k^*,p} - \mathbf{Q}^\alpha_{\pi_k^0,p} \right\|_\infty + 2\alpha \left(1 - \frac{\eta\alpha}{1-\gamma} \left\| \log(\pi_k^*) - \log(\pi_k^0) \right\|_\infty \right)$$

and

$$C' \geq \frac{2\epsilon}{1-\gamma}(1 + \frac{\gamma}{\eta\alpha}),$$

it follows that for all $t \geq 0$

$$\left\| \mathbf{Q}^\alpha_{\pi_k^*,p} - \mathbf{Q}^\alpha_{\pi_k^{t+1},p} \right\|_\infty \leq C_k\gamma(1-\eta\alpha)^t + \gamma C' \tag{63}$$

$$\left\| \log(\pi_k^*) - \log(\pi_k^{t+1}) \right\|_\infty \leq 2C_k\alpha^{-1}(1-\eta\alpha)^t + 2\alpha^{-1}C' \tag{64}$$

$$\left\| \mathbf{V}^\alpha_{\pi_k^*,p} - \mathbf{V}^\alpha_{\pi_k^{t+1},p} \right\|_\infty \leq 3C_k(1-\eta\alpha)^t + 3C'. \tag{65}$$

The following lemma provides settings for the number, t_k, of iterations to optimise the policy such that it has negligible error on the regularised objective (i.e. with KL-divergence and modified Lagrangian multiplier).

Lemma 18 (Number of inner loop iterations, Lemma 7 in Liu et al. (2021)). *Let $\epsilon > 0$, $\left\| \hat{\mathbf{Q}}^\alpha_{\pi_k^t,p} - \mathbf{Q}^\alpha_{\pi_k^t,p} \right\|_\infty \leq \epsilon$, $\eta \leq (1-\gamma)/\alpha$, $t_k = \frac{1}{\eta\alpha}\log(3C_kK)$, and*

$$C_k = 2\gamma \left(\frac{1 + \sum_{j=1}^m \lambda_j}{1-\gamma} + \frac{m\eta_\lambda}{(1-\gamma)^2} \right).$$

It follows that

$$\left\| \mathbf{V}^\alpha_{\pi_k^*,p} - \mathbf{V}_{\pi_{k+1}}\alpha \right\|_\infty \leq \frac{1}{K} + \frac{6\epsilon}{(1-\gamma)^2} \tag{66}$$

and

$$\left\| \log(\pi_k^*) - \log(\pi_{k+1}) \right\|_\infty \leq \frac{2}{3\alpha K} + \frac{4\epsilon}{\alpha(1-\gamma)^2}. \tag{67}$$

The following lemma bounds the performance difference of entropy-regularised NPG based on the approximation error.

Lemma 19 (Performance difference lemma, Lemma 4 in Cen et al. (2022) and Lemma 6 in Liu et al. (2021))**.** *For learning rate $\eta \leq (1-\gamma)/\alpha$ and any $t \geq 0$, it holds for any starting distribution ρ that under entropy-regularised NPG,*

$$\mathbf{V}^{\alpha}_{\pi_k^{t+1},p}(\rho) - \mathbf{V}^{\alpha}_{\pi_k^t,p}(\rho) \leq \frac{2}{1-\gamma} \left\| \hat{\mathbf{Q}}^{\alpha}_{\pi_k^t,p} - \mathbf{Q}^{\alpha}_{\pi_k^t,p} \right\|_{\infty} . \tag{68}$$

Corollary 1. *It follows from Lemma 19 that*

$$Q_{\pi_k^{t+1},p}(s,a) - Q_{\pi_k^t,p}(s,a) = \gamma(V_{\pi_k^{t+1},p}(p(\cdot|s,a)) - V_{\pi_k^t,p}(p(\cdot|s,a))) \leq \frac{2\gamma}{1-\gamma} \left\| \hat{Q}_{\pi_k^t} - Q_{\pi_k^t} \right\|_{\infty} . \tag{69}$$

C.4 Performance difference across transition kernels

In the context of robust MDPs, the performance difference lemmas below have been formulated across transition kernels, which help the analysis of changing transition dynamics in RCMDPs.

Lemma 20 (First performance difference lemma across transition kernels, Lemma 5.2 in Wang et al. (2023))**.** *For any pair of transition kernels $p, p' \in \mathcal{P}$ and any policy $\pi \in \Pi$, we have for any value function that*

$$V_{\pi,p}(\rho) - V_{\pi,p'}(\rho) = \frac{1}{1-\gamma} \sum_{s\in\mathcal{S}} d_{\rho}^{\pi,p'}(s) \left(\sum_{a\in\mathcal{A}} \pi(a|s) \sum_{s'\in\mathcal{S}} (p(s'|s,a) - p'(s'|s,a)) \left[c(s,a,s') + \gamma V_{\pi,p}(s') \right] \right) . \tag{70}$$

Lemma 21 (Second performance difference lemma across transition kernels, Lemma 5.3 in Wang et al. (2023))**.** *For any pair of transition kernels $p, p' \in \mathcal{P}$ and any policy $\pi \in \Pi$, we have for any value function that*

$$V_{\pi,p}(\rho) - V_{\pi,p'}(\rho) = \frac{1}{1-\gamma} \sum_{s\in\mathcal{S}} d_{\rho}^{\pi,p}(s) \left(\sum_{a\in\mathcal{A}} \pi(a|s) \sum_{s'\in\mathcal{S}} (p(s'|s,a) - p'(s'|s,a)) \left[c(s,a,s') + \gamma V_{\pi,p'}(s') \right] \right) . \tag{71}$$

C.5 Value function approximation

A proof of Lemma 9 is provided below to demonstrate its applicability to the robust Sample-based PMD algorithm.

Proof. **a): approximation of $V^i_{\pi_k^t,p_k}(\rho)$ for all $i \in [m]$.**
Pick $i \in [m]$ and $k \in [K]$ arbitrarily. Note that since costs and constraint-costs are in $[-1,1]$, the cumulative discounted constraint-cost is bounded by $\sum_{l=0}^{\infty} \gamma^t c_i(s_l,a_l) \in [-\frac{1}{1-\gamma}, \frac{1}{1-\gamma}]$. Denoting the $N_{V,k}$-step cumulative discounted constraint-cost as a random variable $X = \sum_{l=0}^{N_{V,k}} \gamma^l c_i(s_l,a_l)$, and $\bar{X} = \frac{1}{M_{V,k}} \sum_{n=1}^{M_{V,k}} X_n$, we have by Hoeffding's inequality that the number of samples required is derived as

$$\begin{aligned}
&P(|\bar{X} - \mathbb{E}[\bar{X}]| \geq \epsilon) \\
&\leq 2\exp\left(-2 \frac{\epsilon^2}{\sum_{i=1}^{M_{V,k}} \left(\frac{2}{M_{V,k}(1-\gamma)} \right)^2} \right) \\
&= 2\exp\left(-\frac{(1-\gamma)^2 M_{V,k} \epsilon^2}{2} \right) = \delta_k \\
&M_{V,k} = \log\left(\frac{2}{\delta_k} \right) \frac{2}{(1-\gamma)^2\epsilon^2} = \Theta\left(\log\left(\delta_k^{-1}\right) \frac{1}{\epsilon^2} \right) .
\end{aligned}$$

Further, note that since constraint-costs are in $[-1, 1]$, it follows that the number of time steps is derived as

$$
\begin{aligned}
&|\mathbb{E}[X] - V^i_{\pi_k}(\rho)| \leq 2 \sum_{t=N_{V,k}}^{\infty} \gamma^t = 2\frac{\gamma^{N_{V,k}}}{1-\gamma} = \epsilon \\
&N_{V,k} \log(\gamma) = \log((1-\gamma)\epsilon/2) \\
&N_{V,k} = \log_\gamma((1-\gamma)\epsilon/2) = \Theta\left(\log_{1/\gamma}\left(\frac{1}{\epsilon}\right)\right), \qquad (72)
\end{aligned}
$$

thereby obtaining an ϵ-precise estimate of $V^i_{\pi_k}(\rho)$ with the chosen settings of $M_{V,k}$ and $N_{V,k}$.

b): approximation of $\tilde{\mathbf{Q}}^{\alpha}_{\pi^t_k, p_k}(s,a)$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$. Select $(s,a) \in \mathcal{S} \times \mathcal{A}$ arbitrarily. First note that $\tilde{\mathbf{c}}_k(s,a) = \mathcal{O}(\max\{1, \|\lambda_k\|_1\})$ and similarly $\tilde{V}_{\pi^t_k, p_k}(s) = \mathcal{O}(\max\{1, \|\lambda_k\|_1\})$ (again omitting the division by $1-\gamma$ from the notation).

Define the random variable

$$
X(s,a) = \tilde{\mathbf{c}}(s,a) + \alpha \log\left(\frac{1}{\pi_k(a|s)}\right) + \sum_{l=1}^{N_{Q,k}} \gamma^l \left(\tilde{\mathbf{c}}(s_l, a_l) + \alpha \sum_{a'} \pi^t_k(a'|s_l) \log\left(\frac{\pi^t_k(a'|s_l)}{\pi_k(a'|s_l)}\right)\right).
$$

For $t = 0$, the regularisation term drops. Defining $\bar{X} = \frac{1}{M_{Q,k}} \sum_{n=1}^{M_{Q,k}} X(s,a)_n$, we obtain

$$
\begin{aligned}
&P(|\bar{X} - \mathbb{E}[\bar{X}]| \geq \epsilon) \\
&\leq 2\exp\left(-2\frac{\epsilon^2}{M_{Q,k}\left(\frac{\max\{1, \|\lambda_k\|_1\}}{M_{Q,k}}\right)^2}\right) = \delta_k \\
&M_{Q,k} = \log\left(\frac{2}{\delta_k}\right)\frac{\max\{1, \|\lambda_k\|_1\}^2}{\epsilon^2} = \Theta\left(\log\left(\delta_k^{-1}\right)\frac{\max\{1, \|\lambda_k\|_1\}^2}{\epsilon^2}\right).
\end{aligned}
$$

Therefore, setting $M_{Q,K} = \Theta\left(\log(\delta_k^{-1})\frac{\max\{1, \|\lambda_k\|_1\}^2 + \epsilon t_k}{\epsilon^2}\right)$ is sufficient for approximating $\tilde{\mathbf{Q}}^{\alpha}_{\pi^0_k, p_k}(s,a)$.

For any $t > 0$, note that the regularisation term is bounded by

$$
\begin{aligned}
&\alpha \sum_{a'} \pi^t_k(a'|s_l) \log\left(\frac{\pi^t_k(a'|s_l)}{\pi_k(a'|s_l)}\right) \\
&\leq \tilde{\mathbf{V}}^{\alpha}_{\pi^t_k, p_k}(s) - \mathbf{V}_{\pi^t_k, p_k}(s) \\
&\leq \tilde{\mathbf{V}}^{\alpha}_{\pi^t_k, p_k}(s) + \mathcal{O}(\max\{1, \|\lambda_k\|_1\}) \quad \text{(from range)} \\
&\leq \tilde{\mathbf{V}}^{\alpha}_{\pi_k, p_k}(s) + \mathcal{O}(\max\{1, \|\lambda_k\|_1\}) + \sum_{t=0}^{t-1} \frac{2}{1-\gamma} \left\| \hat{\mathbf{Q}}^{\alpha}_{\pi^t_k, p_k} - \mathbf{Q}^{\alpha}_{\pi^t_k, p_k} \right\|_\infty \quad \text{(iteratively applying Lemma 19)} \\
&\leq \tilde{\mathbf{V}}_{\pi_k, p_k}(s) + \frac{\alpha}{1-\gamma}\log(A) + \mathcal{O}(\max\{1, \|\lambda_k\|_1\}) + \sum_{t=0}^{t-1} \frac{2}{1-\gamma} \left\| \hat{\mathbf{Q}}^{\alpha}_{\pi^t_k, p_k} - \mathbf{Q}^{\alpha}_{\pi^t_k, p_k} \right\|_\infty \quad \text{(via bound on KL-divergence)} \\
&\leq \mathcal{O}(\max\{1, \|\lambda_k\|_1\} + 2\epsilon t_k).
\end{aligned}
$$

Since the unregularised Lagrangian is bounded by $\mathcal{O}(\max\{1, \|\lambda_k\|_1\})$, and again applying Hoeffding's inequality, it follows that $M_{Q,K} = \Theta\left(\log(\delta_k^{-1})\frac{\left(\max\{1, \|\lambda_k\|_1\} + \epsilon t_k\right)^2}{\epsilon^2}\right)$ is sufficient for approximating $\tilde{\mathbf{Q}}^{\alpha}_{\pi^t_k, p_k}(s,a)$.

With similar reasoning as in Eq. 72, but applied to the augmented regularised Lagrangian, we have

$$
\begin{aligned}
|\mathbb{E}[X] - \mathbf{Q}_{\pi_k^t}(s,a)| &\leq \sum_{l=N_{Q,k}}^{\infty} \gamma^l \left(\mathcal{O}(\max\{1, \|\lambda_k\|_1\}) + \epsilon t_k\right) \\
&= \frac{\gamma^{N_{Q,k}} \mathcal{O}(\max\{1, \|\lambda_k\|_1\}) + \epsilon t_k}{1-\gamma} = \epsilon \\
N_{Q,k} &= \Theta\left(\log_{1/\gamma}\left(\frac{\mathcal{O}(\max\{1, \|\lambda_k\|_1\})}{\epsilon} + t_k\right)\right)
\end{aligned}
$$

Note that $t_k = \Theta\left(\log(\max\{1, \|\lambda_k\|_1\}/\epsilon)\right) = \mathcal{O}\left(\frac{\max\{1, \|\lambda_k\|_1\})}{\epsilon}\right)$. Therefore $N_{Q,k} = \mathcal{O}\left(\log_{1/\gamma}\left(\frac{\max\{1, \|\lambda_k\|_1\}}{\epsilon}\right)\right)$.

By union bound, with probability at least $1 - \sum_{k=0}^{K-1} t_k \delta_k = 1 - \delta$, the statement holds for iteration $K-1$. □

As we seek to derive a bound based on telescoping, the term $\|\lambda_K\|_1$ is of particular interest.

Lemma 22. *Let $K' \geq 0$ be a macro-iteration. Under the event that $|V^i_{\pi^t_{K'-1}, p_{K'-1}}(\rho) - V^i_{\pi^t_{K'-1}, p_{K'-1}}(\rho)| \leq \epsilon$ for all $i \in [m]$ and $\Delta_p = \Theta\left(\frac{(1-\gamma)}{\alpha \log(A)}\right)$, it follows that $\|\lambda_{K'}\|_1 = \mathcal{O}(1)$.*

Proof. The proof makes use of complementary slackness, telescoping, the inner product in Lemma 13, and the basic upper bound via property 3 of Lemma 12 that $\|\lambda_0\| \leq \|\lambda_0\|_1 \leq \eta_\lambda^2 \frac{m}{1-\gamma}$. Additionally, due to the slack variable $\zeta > 0$ in Assumption 3 for any $p \in \mathcal{P}$ and Lemma 16 in Liu et al. (2021), the optimal dual variable can be bounded based on $\lambda^* \leq \frac{2}{(1-\gamma)\zeta} = \mathcal{O}(1)$. The full derivation is rather lengthy so we refer to Lemma 21 of Liu et al. (2021). Our analysis adds a correction for the transition dynamic changes within the uncertainty set $\Delta_p = \mathcal{O}(1)$ as derived in Eq. 43. Since the additional term is $\alpha \Delta_p \mathcal{O}(\frac{\log(A)}{1-\gamma}) = \mathcal{O}(1)$, we obtain the desired result. □

D Algorithm implementation details

All the algorithms were implemented in Pytorch. The code for RMCPMD was taken from its original implementation `https://github.com/JerrisonWang/JMLR-DRPMD` and the code for PPO was taken from the StableBaselines3 class. Both were modified to fit our purposes by allowing to turn off and on robust training and constraint-satisfaction. Similar to PPO, MDPO was also implemented following StableBaselines3 class conventions. All the environments are implemented in Gymnasium. The updates with LTMA are based on modifying the TMA code in `https://github.com/JerrisonWang/JMLR-DRPMD`. For MDPO and PPO experiments, we use four parallel environments and with four CPUs. Our longest experiments typically take no more than two hours to complete. The remainder of the section describes domain-specific hyperparameter settings.

D.1 Hyperparameters for Cartpole experiments

Hyperparameters settings for the Carpole experiments can be found in 8. Settings for robust optimisation, including the policy architecture, transition kernel architecture, and TMA learning rate are taken from Wang et al. (2023) with the exception that we formulate a more challenging uncertainty set with a 5 times larger range. The policy learning rate and GAE lambda is typical for PPO based methods so we apply these for PPO and MDPO methods. The number of policy epochs and early stopping KL target is based for PPO methods on the standard repository for PPO-Lag (`https://github.com/openai/safety-starter-agents/`) and for MDPO it is based on the original paper's settings (Tomar et al., 2022). PPO obtained better results without entropy regularisation and value function clipping on initial experiments, so for simplicity we disabled these

for all algorithms. The batch size, multiplier initialisation, and multiplier learning rate and scaling are obtained via a limited tuning procedure. For the tuning, the batch size tried was in $\{100, 200, 400, 1000, 2000\}$ and the multiplier learning rate was set to $\{1e^{-1}, 1e^{-2}, 1e^{-3}, 1e^{-4}\}$ on partial runs. Then the use of a softplus transformation of the multiplier was compared to the direct (linear) multiplier, and for the linear multiplier we tested initialisations in $\{1, 5, 10\}$.

Table 8: Hyperparameter settings for the Cartpole experiments.

Hyperparameter	**Setting**
Policy architecture	MLP 4 Inputs – Linear(128) – Dropout(0.6) – Linear(128) – Softmax(2)
Critic architecture	MLP 4 Inputs – Linear(128) – Dropout(0.6) – ReLU – Linear(1+m)
Policy learning rate (η)	$3e^{-4}$
Policy optimiser	Adam
GAE lambda (λ_{GAE})	0.95
Discount factor (γ)	0.99
Batch and minibatch size	PPO/MDPO policy update: 400×4 time steps per batch, minibatch 32, episode steps at most 100 PPO/MDPO multiplier update: 100×4 time steps per batch, minibatch 32, episode steps at most 100 LTMA: $10 \times$ factor[1] Monte Carlo updates, episode steps at most 10
Policy epochs	MCPMD: 1 PPO: 50, early stopping with KL target 0.01 MDPO: 5
Transition kernel architecture	multi-variate Gaussian with parametrised mean in $(1+\delta)\mu_c(s)$ where $\delta \in (\pm 0.005, \pm 0.05, \pm 0.005, \pm 0.05)$ and covariance $\sigma\mathbb{I}$ where $\sigma = 1e^{-7}$
LTMA learning rate (η_ξ)	$1e^{-7}$
Dual learning rate (η_λ)	$1e^{-3}$
Dual epochs	MCPMD: 1 PPO: 50 with early stopping based on target-kl 0.01 MDPO: 5
Multiplier	initialise to 5, linear, clipping to $\lambda_{\max} = 50$ for non-augmented algorithms

D.2 Hyperparameters for Inventory Management experiments

Hyperparameters settings for the unidimensional Inventory Management (see Table 9) are similar to the Cartpole experiments. The key differences are the output layer, the discount factor, the transition kernel, and the TMA parameters, which follow settings of the domain in Wang & Petrik (2024). Based on initial tuning experiments with PPO and MDPO, the learning rates are set to $\eta = 1e^{-3}$ and $\eta_\lambda = 1e^{-2}$, a lower GAE lambda of 0.50 was chosen and PPO is implemented without early stopping and with the same number of (5) epochs as MDPO.

Table 9: Hyperparameter settings for Inventory Management experiments.

Hyperparameter	**Setting**
Policy architecture	MLP 1 Input – Linear(64) – Dropout(0.6) – ReLU – Softmax(4)
Critic architecture	MLP 3 Inputs – Linear(128) – Dropout(0.6) – ReLU – Linear(1+m)
Policy learning rate (η)	$1e^{-3}$
Policy optimiser	Adam
GAE lambda (λ_{GAE})	0.50
Discount factor (γ)	0.95
Batch and minibatch size	PPO/MDPO policy update: 400×4 time steps per batch, minibatch 32, episode steps at most 80 PPO/MDPO multiplier update: 100×4 time steps per batch, minibatch 32, episode steps at most 80 LTMA: $20 \times$ factor Monte Carlo updates, episode steps at most 40
Policy epochs	MCPMD: 1 PPO: 5 MDPO: 5
Transition kernel architecture	Radial features with Gaussian mixture parametrisation
LTMA learning rate (η_ξ)	$1e^{-1}$
Dual learning rate (η_λ)	$1e^{-2}$
Dual epochs	MCPMD: 1 PPO and MDPO: 5
Multiplier	initialise to 5, linear, clipping to $\lambda_{\max} = 500$ for non-augmented algorithms

D.3 Hyperparameters for 3-D Inventory Management experiments

Hyperparameters settings for the 3-D Inventory Management (see Table 9) are the same as in the IM domain with a few exceptions. The policy architecture is now a Gaussian MLP. The standard deviation of the policy is set to the default with Log std init equal to 0.0 albeit after some tuning effort. The optimiser is set to SGD instead of Adam based on improved preliminary results.

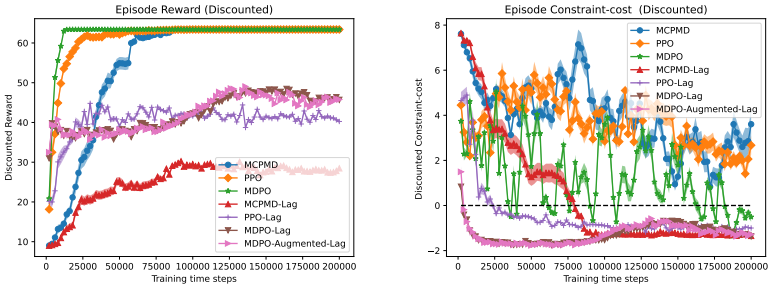
Table 10: Hyperparameter settings for Inventory Management experiments.

Hyperparameter	**Setting**
Policy architecture	Gaussian MLP 3 Inputs – Linear(128) – Dropout(0.6) – ReLU – Linear(128) – ReLU – Linear(3×2)
Critic architecture	MLP 3 Inputs – Linear(128) – Dropout(0.6) – ReLU – Linear(1+m)
Policy learning rate (η)	$1e^{-3}$
Policy optimiser	SGD
Log std init	0.0
GAE lambda (λ_{GAE})	0.50
Discount factor (γ)	0.95
Batch and minibatch size	PPO/MDPO policy update: 400×4 time steps per batch, minibatch 32, episode steps at most 100 PPO/MDPO multiplier update: 100×4 time steps per batch, minibatch 32, episode steps at most 100 LTMA: $20 \times$ factor Monte Carlo updates, episode steps at most 40
Policy epochs	MCPMD: 1 PPO: 5 MDPO: 5
Transition kernel architecture	Radial features with Gaussian mixture parametrisation
LTMA learning rate (η_ξ)	$1e^{-1}$
Dual learning rate (η_λ)	$1e^{-2}$
Dual epochs	MCPMD: 1 PPO and MDPO: 5
Multiplier	initialise to 5, linear, clipping to $\lambda_{\max} = 500$ for non-augmented algorithms

E Training performance plots

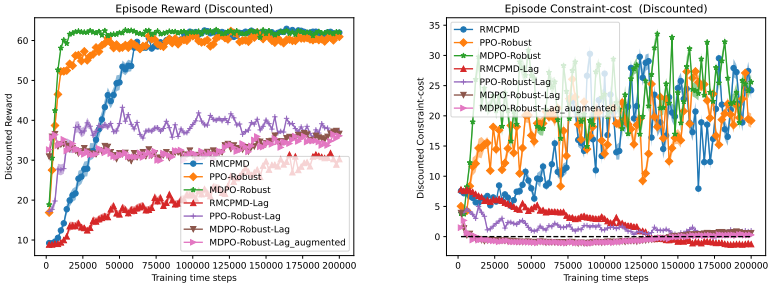
While the experiments with small and large training time steps were run independently, we report here the training development under the large training data regime (i.e. between 200,000 and 400,000 time steps) since this gives a view of both the early and late stages of training.

E.1 Cartpole



(a) Reward (b) Constraint-cost

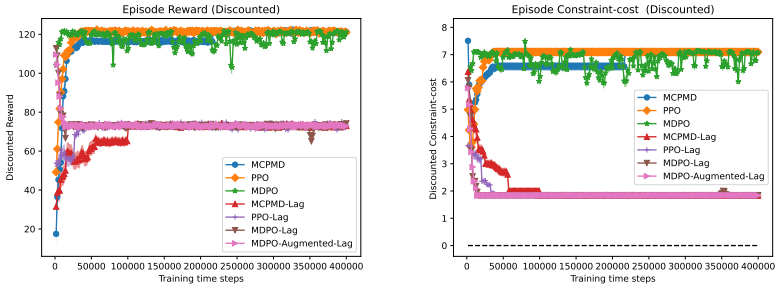
Figure 7: Constrained MDP training development plots in the Cartpole domain, where each sample in the plot is based on 20 evaluations of the deterministic policy. The line and shaded area represent the mean and standard error across 10 seeds.



(a) Reward (b) Constraint-cost

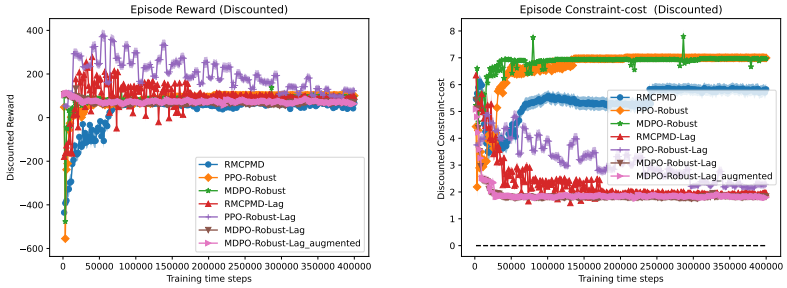
Figure 8: Robust constrained MDP training development plots in the Cartpole domain, where each sample in the plot is based on 20 evaluations of the deterministic policy as it interacts with the adversarial environment from that iteration. The line and shaded area represent the mean and standard error across 10 seeds.

E.2 Inventory Management



(a) Reward (b) Constraint-cost

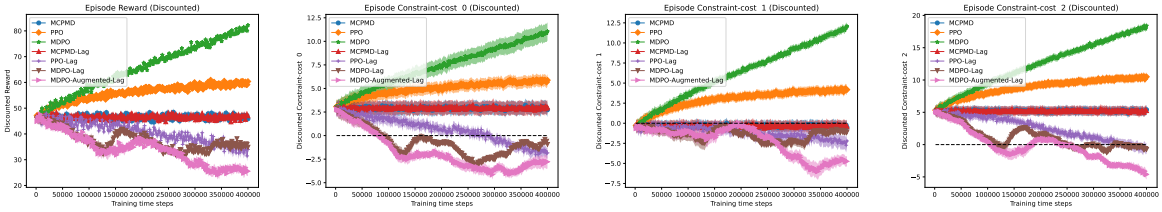
Figure 9: Constrained MDP training development plots in the Inventory Management domain, where each sample in the plot is based on 20 evaluations of the deterministic policy. The line and shaded area represent the mean and standard error across 10 seeds.



(a) Reward (b) Constraint-cost

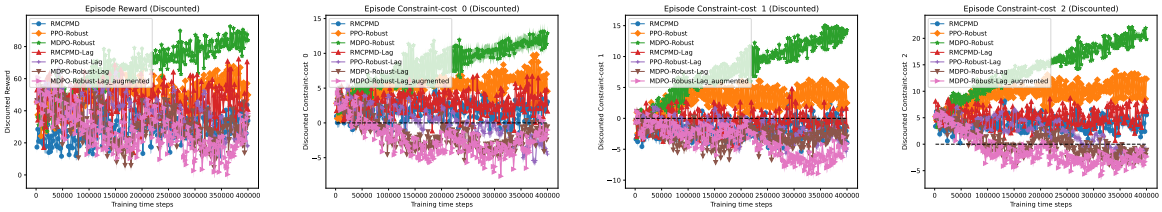
Figure 10: Robust constrained MDP training development plots in the Inventory Management domain, where each sample in the plot is based on 20 evaluations of the deterministic policy as it interacts with the adversarial environment from that iteration. The line and shaded area represent the mean and standard error across 10 seeds.

E.3 3-D Inventory Management



(a) Reward (b) Constraint-cost 1 (c) Constraint-cost 2 (d) Constraint-cost 3

Figure 11: Constrained MDP training development plots in the 3-D Inventory Management domain, where each sample in the plot is based on 20 evaluations of the deterministic policy. The line and shaded area represent the mean and standard error across 10 seeds.



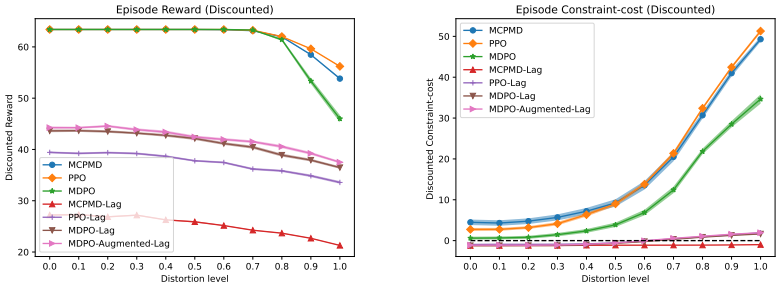
(a) Reward (b) Constraint-cost 1 (c) Constraint-cost 2 (d) Constraint-cost 3

Figure 12: Robust constrained MDP training development plots in the 3-D Inventory Management domain, where each sample in the plot is based on 20 evaluations of the deterministic policy as it interacts with the adversarial environment from that iteration. The line and shaded area represent the mean and standard error across 10 seeds.

F Test performance plots with large sample budget

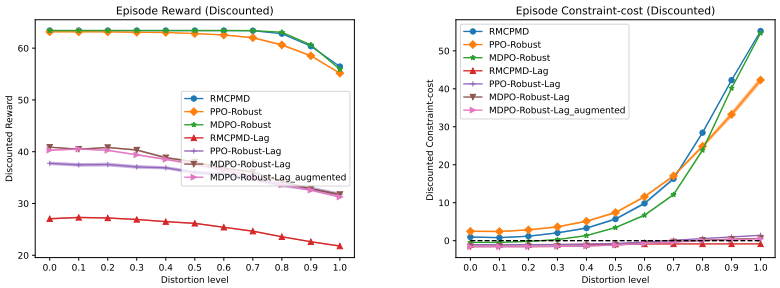
While the main text presents the test performance plots with small sample budgets, between 16,000 and 50,000 time steps, the section below presents the test performance plots after the larger sample budget of 200,000 to 500,000 time steps.

F.1 Cartpole



(a) Reward (b) Constraint-cost

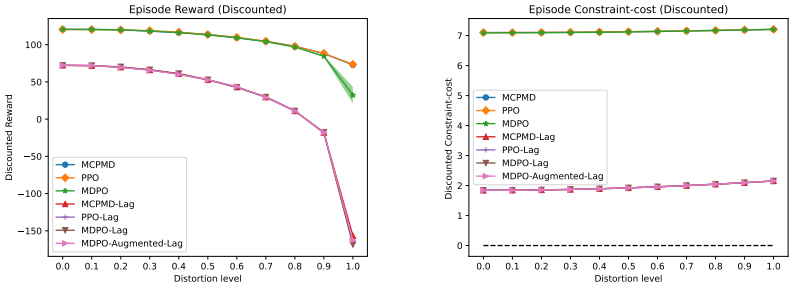
Figure 13: Test performance of MDP and CMDP algorithms obtained by applying the learned deterministic policy from the Cartpole domain after 200,000 time steps of training. The line and shaded area represent the mean and standard error across the perturbations for the particular distortion level.



(a) Reward (b) Constraint-cost

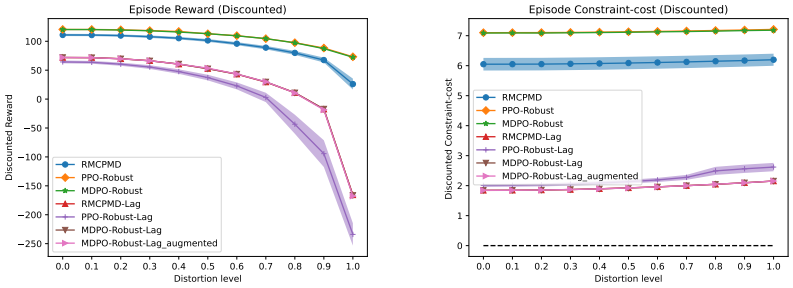
Figure 14: Test performance of RMDP and RCMDP algorithms obtained by applying the learned deterministic policy from the Cartpole domain after 200,000 time steps of training. The line and shaded area represent the mean and standard error across the perturbations for the particular distortion level.

F.2 Inventory Management



(a) Reward (b) Constraint-cost

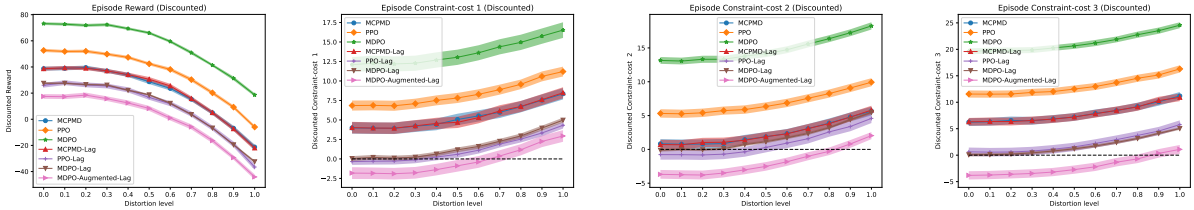
Figure 15: Test performance of MDP and CMDP algorithms obtained by applying the learned deterministic policy from the Inventory Management domain after 400,000 time steps of training. The line and shaded area represent the mean and standard error across the perturbations for the particular distortion level.



(a) Reward (b) Constraint-cost

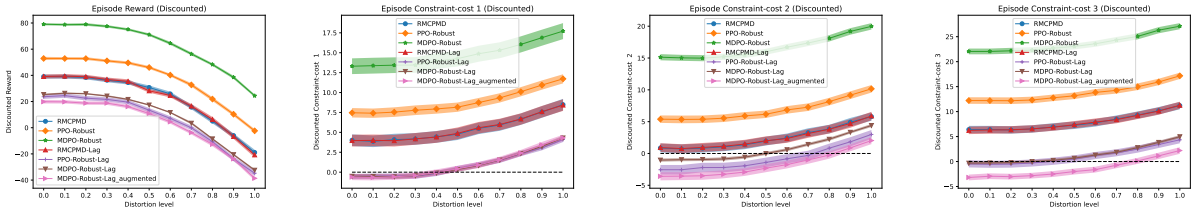
Figure 16: Test performance of RMDP and RCMDP algorithms obtained by applying the learned deterministic policy from the Inventory Management domain after 400,000 time steps of training. The line and shaded area represent the mean and standard error across the perturbations for the particular distortion level.

F.3 3-D Inventory Management



(a) Reward (b) Constraint-cost 1 (c) Constraint-cost 2 (d) Constraint-cost 3

Figure 17: Test performance of RMDP and RCMDP algorithms obtained by applying the learned deterministic policy from the 3-D Inventory Management domain after 400,000 time steps of training. The line and shaded area represent the mean and standard error across the perturbations for the particular distortion level.



(a) Reward (b) Constraint-cost 1 (c) Constraint-cost 2 (d) Constraint-cost 3

Figure 18: Test performance of RMDP and RCMDP algorithms obtained by applying the learned deterministic policy from the 3-D Inventory Management domain after 400,000 time steps of training. The line and shaded area represent the mean and standard error across the perturbations for the particular distortion level.

G Comparison of schedules

Tomar et al. (2022) propose either a fixed α_k or a linear increase across iterations to give a larger penalty at the end of the iterations (inspired by theoretical works, e.g. Beck & Teboulle (2003)). A potential problem with such a linear scheme is that if the LTMA update is strong and the policy cannot take sufficiently large learning steps, it may sometimes lead to a gradual loss of performance and premature convergence, especially in the context of contrained optimisation where the objective is often subject to large changes due to constraints becoming active or inactive. As an alternative, we also consider a schedule where after every time the LTMA update to ξ was larger than a particular threshold, such that $\alpha_k = \frac{1}{1-(k-k')/(K-k')}$, where k' indicates the restart time. In the experiments, the restart is done every time any dimension i of the update is larger than half the maximal distance from nominal, i.e. larger than $0.5 \max_{\xi_i \in \mathcal{U}_{\xi_i}} |\xi_i - \bar{\xi}_i|$. A last included schedule is the geometrically decreasing schedule $\alpha_k = \alpha_{k-1} * \gamma$ based on the discount factor γ as proposed in earlier works (Xiao, 2022; Wang & Petrik, 2024).

Table 11 summarises the experiments comparing fixed schedules, restart schedules, linear schedules, and the geometric schedules for MDPO-Robust and MDPO-Robust-Lag variants. For the fixed schedule, the parameter is set to $\alpha = 2$. For the geometric schedule, the starting parameter is set to $\alpha_0 = 5.0$. An overall conclusion is that a fixed setting works reasonably well overall, while in particular cases there may be benefits from time-varying schedules.

Table 11: Return and penalised return statistics comparing the schedule with restart schedules to the traditional linear schedule under the short runs (200-500 episodes × maximal number of time steps) and the long runs (2000-5000 episodes × maximal number of time steps).

Domain	Algorithm	Return		$R^{\pm}_{\text{pen}}$ (signed)		R_{pen} (positive)	
		Mean ± SE	Min	Mean ± SE	Min	Mean ± SE	Min
Cartpole (short runs)	MDPO-Robust (fixed)	63.1 ± 0.1	56.4	-229.7 ± 179.6	-2610.6	-294.1 ± 171.4	-2610.8
	MDPO-Robust (linear)	62.9 ± 0.1	55.0	-482.5 ± 182.3	-2487.1	-514.6 ± 177.4	-2487.1
	MDPO-Robust (restart)	63.1 ± 0.1	55.2	-252.8 ± 179.1	-2520.5	-304.8 ± 172.0	-2520.6
	MDPO-Robust (geometric)	63.1 ± 0.1	55.8	-148.0 ± 173.5	-2480.3	-219.3 ± 164.7	-2481.5
	MDPO-Robust-Lag (fixed)	34.4 ± 0.3	24.9	108.4 ± 4.4	65.9	34.2 ± 1.2	24.9
	MDPO-Robust-Lag (linear)	35.3 ± 0.3	25.2	107.1 ± 5.0	53.0	34.5 ± 1.2	25.0
	MDPO-Robust-Lag (restart)	34.9 ± 0.3	25.7	107.2 ± 4.9	53.1	34.3 ± 1.1	25.4
	MDPO-Robust-Lag (geometric)	34.8 ± 0.3	25.0	109.4 ± 4.8	60.2	34.5 ± 1.2	25.0
	MDPO-Robust-Augmented-Lag (fixed)	34.6 ± 0.3	24.7	110.8 ± 4.9	61.6	34.6 ± 1.2	24.7
	MDPO-Robust-Augmented-Lag (linear)	35.2 ± 0.3	25.4	106.6 ± 5.0	52.6	34.2 ± 1.2	22.9
	MDPO-Robust-Augmented-Lag (restart)	35.4 ± 0.3	25.7	107.1 ± 5.4	45.6	34.4 ± 1.3	21.7
	MDPO-Robust-Augmented-Lag (geometric)	35.1 ± 0.3	24.6	109.7 ± 5.1	56.4	34.7 ± 1.2	24.6
Cartpole (long runs)	MDPO-Robust (fixed)	63.2 ± 0.1	56.1	-109.7 ± 176.0	-2647.1	-185.5 ± 167.1	-2648.3
	MDPO-Robust (linear)	33.7 ± 0.1	29.2	-145.3 ± 24.1	-526.4	-165.0 ± 23.0	-541.6
	MDPO-Robust (restart)	63.1 ± 0.1	55.8	-173.6 ± 174.9	-2537.4	-237.1 ± 167.0	-2538.1
	MDPO-Robust (geometric)	63.2 ± 0.1	56.1	-120.4 ± 179.5	-2662.1	-193.7 ± 170.7	-2662.7
	MDPO-Robust-Lag (fixed)	41.6 ± 0.2	32.1	105.1 ± 9.5	9.6	38.6 ± 2.5	2.6
	MDPO-Robust-Lag (linear)	32.3 ± 0.3	22.9	72.0 ± 4.1	38.6	15.4 ± 1.8	-4.8
	MDPO-Robust-Lag (restart)	41.8 ± 0.2	32.2	107.6 ± 9.3	14.5	39.2 ± 2.3	5.8
	MDPO-Robust-Lag (geometric)	41.4 ± 0.2	31.7	103.7 ± 9.4	8.0	38.6 ± 2.5	1.3
	MDPO-Robust-Augmented-Lag (fixed)	41.2 ± 0.2	31.4	106.2 ± 9.1	13.7	38.6 ± 2.2	5.0
	MDPO-Robust-Augmented-Lag (linear)	30.0 ± 0.3	21.5	32.8 ± 2.8	12.2	-18.8 ± 1.3	-34.6
	MDPO-Robust-Augmented-Lag (restart)	40.5 ± 0.2	30.5	108.7 ± 8.2	24.1	38.3 ± 1.9	10.7
	MDPO-Robust-Augmented-Lag (geometric)	40.9 ± 0.2	31.4	102.1 ± 8.9	9.2	37.6 ± 2.7	-2.1
Inventory Management (short runs)	MDPO-Robust (fixed)	118.0 ± 1.8	72.3	-3306.4 ± 9.7	-3350.0	-3306.4 ± 9.7	-3350.0
	MPDO-Robust (linear)	98.4 ± 5.7	-59.6	-2764.0 ± 20.5	-2808.5	-2764.2 ± 20.2	-2808.5
	MPDO-Robust (restart)	105.1 ± 6.3	-20.2	-3404.8 ± 32.9	-3634.3	-3404.8 ± 32.9	-3634.3
	MPDO-Robust (geometric)	114.6 ± 2.3	49.7	-3166.4 ± 6.9	-3207.6	-3166.5 ± 6.9	-3207.6
	MPDO-Robust-Lag (fixed)	60.7 ± 10.7	-228.5	-930.9 ± 32.3	-1118.2	-932.2 ± 30.8	-1118.2
	MPDO-Robust-Lag (linear)	93.2 ± 4.7	-55.0	-2186.8 ± 6.9	-2248.8	-2187.3 ± 6.4	-2248.8
	MPDO-Robust-Lag (restart)	88.3 ± 4.8	-61.3	-1880.7 ± 16.5	-2003.7	-1881.1 ± 16.1	-2003.7
	MPDO-Robust-Lag (geometric)	67.2 ± 9.0	-156.8	-1046.3 ± 10.2	-1122.4	-1048.1 ± 8.0	-1122.4
	MPDO-Robust-Augmented-Lag (fixed)	60.9 ± 10.6	-221.4	-938.6 ± 34.4	-1172.2	-940.1 ± 32.9	-1172.2
	MPDO-Robust-Augmented-Lag (linear)	91.8 ± 4.4	-47.8	-2033.0 ± 8.0	-2054.6	-2033.3 ± 7.6	-2054.6
	MPDO-Robust-Augmented-Lag (restart)	85.6 ± 5.9	-94.9	-1902.5 ± 6.5	-1954.2	-1902.7 ± 6.2	-1954.2
	MPDO-Robust-Augmented-Lag (geometric)	60.0 ± 9.3	-206.9	-1095.9 ± 18.3	-1128.7	-1097.2 ± 16.3	-1128.7
Inventory Management (long runs)	MDPO-Robust (fixed)	119.7 ± 1.8	72.4	-3416.7 ± 4.1	-3429.7	-3416.7 ± 4.1	-3429.7
	MPDO-Robust (linear)	114.8 ± 2.3	53.6	-3166.7 ± 6.8	-3208.2	-3166.7 ± 6.8	-3208.2
	MPDO-Robust (restart)	119.9 ± 1.8	74.4	-3425.2 ± 5.4	-3460.2	-3425.2 ± 5.4	-3460.2
	MPDO-Robust (geometric)	114.7 ± 2.3	51.0	-3166.5 ± 6.9	-3207.6	-3166.6 ± 6.9	-3207.6
	MPDO-Robust-Lag (fixed)	66.9 ± 7.1	-166.5	-837.4 ± 21.4	-939.3	-838.0 ± 20.4	-939.3
	MPDO-Robust-Lag (linear)	66.8 ± 7.1	-165.7	-837.3 ± 21.4	-939.1	-838.0 ± 20.3	-939.1
	MPDO-Robust-Lag (restart)	66.8 ± 7.2	-169.1	-837.3 ± 21.6	-939.8	-838.0 ± 20.5	-939.8
	MPDO-Robust-Lag (geometric)	66.8 ± 7.2	-168.9	-837.3 ± 21.6	-939.7	-837.9 ± 20.6	-939.7
	MPDO-Robust-Augmented-Lag (fixed)	66.8 ± 7.1	-167.7	-837.2 ± 21.5	-939.1	-837.9 ± 20.5	-939.1
	MPDO-Robust-Augmented-Lag (linear)	66.8 ± 7.1	-163.4	-837.4 ± 21.3	-939.2	-838.0 ± 20.3	-939.2
	MPDO-Robust-Augmented-Lag (restart)	66.8 ± 7.1	-162.8	-837.4 ± 21.2	-939.0	-837.9 ± 20.4	-939.0
	MPDO-Robust-Augmented-Lag (geometric)	66.9 ± 7.1	-162.6	-837.4 ± 21.2	-939.2	-838.1 ± 20.2	-939.2
3-D Inventory Management (short runs)	MDPO-Robust (fixed)	54.9 ± 0.4	-9.6	-7956.9 ± 457.5	-14747.8	-9376.8 ± 330.2	-14836.3
	MPDO-Robust (linear)	56.5 ± 0.4	-8.1	-8716.0 ± 458.1	-15404.0	-9918.5 ± 343.1	-15469.0
	MPDO-Robust (restart)	56.3 ± 0.4	-8.7	-8693.9 ± 455.4	-15503.7	-9866.2 ± 333.9	-15531.6
	MPDO-Robust (geometric)	53.6 ± 0.4	-11.5	-7337.6 ± 459.1	-14074.2	-8846.6 ± 315.8	-14173.9
	MPDO-Robust-Lag (fixed)	41.4 ± 0.5	-24.6	-1416.9 ± 446.1	-7726.9	-5088.9 ± 221.7	-8560.6
	MPDO-Robust-Lag (linear)	41.8 ± 0.5	-25.5	-1584.6 ± 447.5	-8169.0	-5178.0 ± 222.3	-8982.4
	MPDO-Robust-Lag (restart)	41.9 ± 0.5	-24.9	-1648.0 ± 446.1	-8138.0	-5308.1 ± 223.1	-8995.2
	MPDO-Robust-Lag (geometric)	41.1 ± 0.5	-28.2	-1278.3 ± 449.6	-7571.7	-5060.9 ± 219.1	-8659.8
	MPDO-Robust-Augmented-Lag (fixed)	39.5 ± 0.5	-29.6	-525.0 ± 441.2	-6861.9	-4957.0 ± 211.2	-8289.3
	MPDO-Robust-Augmented-Lag (linear)	42.4 ± 0.5	-23.7	-1929.8 ± 446.9	-8420.5	-5311.8 ± 228.7	-9203.4
	MPDO-Robust-Augmented-Lag (restart)	40.7 ± 0.5	-26.8	-1070.1 ± 441.9	-7347.8	-5155.6 ± 216.3	-8671.2
	MPDO-Robust-Augmented-Lag (geometric)	42.8 ± 0.5	-25.1	-2123.5 ± 450.2	-8567.3	-5507.8 ± 228.9	-9303.0
3-D Inventory Management (long runs)	MDPO-Robust (fixed)	87.0 ± 0.4	24.4	-23312.6 ± 525.6	-31314.1	-23459.7 ± 499.3	-31314.1
	MPDO-Robust (linear)	96.3 ± 0.4	36.0	-27752.5 ± 547.7	-35936.5	-27902.6 ± 525.3	-35936.5
	MPDO-Robust (restart)	90.8 ± 0.4	27.4	-25169.7 ± 523.8	-32979.5	-25265.0 ± 504.2	-32979.5
	MPDO-Robust (geometric)	82.2 ± 0.4	20.0	-21022.5 ± 487.2	-28373.0	-21098.2 ± 469.0	-28373.0
	MPDO-Robust-Lag (fixed)	32.8 ± 0.5	-32.4	2617.9 ± 378.6	-2601.7	-1011.4 ± 169.9	-3767.4
	MPDO-Robust-Lag (linear)	33.2 ± 0.5	-31.9	2538.8 ± 368.0	-2793.7	-1353.1 ± 197.6	-4804.2
	MPDO-Robust-Lag (restart)	34.0 ± 0.4	-31.2	2062.3 ± 375.3	-3333.9	-1317.1 ± 191.5	-4687.1
	MPDO-Robust-Lag (geometric)	31.7 ± 0.5	-34.9	3269.5 ± 371.3	-2036.2	-1097.3 ± 173.3	-4299.3
	MPDO-Robust-Augmented-Lag (fixed)	26.8 ± 0.4	-38.6	5312.6 ± 340.2	758.7	-1219.2 ± 150.4	-3867.7
	MPDO-Robust-Augmented-Lag (linear)	23.5 ± 0.4	-41.1	6993.3 ± 325.6	2682.2	-963.4 ± 122.0	-3066.4
	MPDO-Robust-Augmented-Lag (restart)	24.2 ± 0.5	-41.3	6811.0 ± 333.5	2581.0	-1043.3 ± 159.2	-3775.2
	MPDO-Robust-Augmented-Lag (geometric)	26.5 ± 0.5	-40.3	5494.4 ± 348.4	824.2	-1644.2 ± 167.2	-4586.5