MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Lyapunov Robust Constrained-MDPs for Sim2Real Transfer Learning

Russel, Reazul Hasan; Benosman, Mouhacine; van Baar, Jeroen; Corcodel, Radu

TR2022-111 September 29, 2022

Abstract

Safety and robustness are two desired properties for any reinforcement learning algorithm. Constrained Markov Decision Processes (CMDPs) can handle additional safety constraints and Robust Markov Decision Processes (RMDPs) can perform well under model uncertainties. In this chapter, we propose to unify these two frameworks resulting in Robust Constrained MDPs (RCMDPs). The motivation is to develop a framework that can satisfy safety constraints while also simultaneously offer robustness to model uncertainties. We develop the RCMDP objective, derive gradient update formula to optimize this objective and then propose policy gradient based algorithms. We also independently propose Lyapunov-based reward shaping for RCMDPs, yielding better stability and convergence properties.

Federated and Transfer Learning 2022

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Lyapunov Robust Constrained-MDPs for Sim2Real Transfer Learning[*]

Reazul Hasan Russel, Mouhacine Benosman, Jeroen van Baar, Radu Corcodel

Abstract Safety and robustness are two desired properties for any reinforcement learning algorithm. Constrained Markov Decision Processes (CMDPs) can handle additional safety constraints and Robust Markov Decision Processes (RMDPs) can perform well under model uncertainties. In this chapter, we propose to unify these two frameworks resulting in Robust Constrained MDPs (RCMDPs). The motivation is to develop a framework that can satisfy safety constraints while also simultaneously offer robustness to model uncertainties. We develop the RCMDP objective, derive gradient update formula to optimize this objective and then propose policy gradient based algorithms. We also independently propose Lyapunov-based reward shaping for RCMDPs, yielding better stability and convergence properties.

1 Introduction

Reinforcement learning (RL) is a framework to address sequential decision-making problems Sutton & Barto (2018); Szepesvári (2010). In RL, a decision maker learns

[*] This is a preprint of the following chapter: [Russel et al.], [Lyapunov Robust Constrained-MDPs for Sim2Real Transfer Learning], R. Razavi-Far et al. (eds.), Federated and Transfer Learning, Adaptation, Learning, and Optimization 27, https://doi.org/10.1007/978-3-031-11748-0_13

Reazul Hasan Russel
Internship at Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA 02139, USA, e-mail: rrussel@cs.unh.edu

Mouhacine Benosman
MERL, e-mail: benosman@merl.com

Jeroen van Baar
MERL, e-mail: jeroen@merl.com

Radu Corcodel
MERL, e-mail: corcodel@merl.com

a policy to optimize a long-term objective by interacting with the (unknown or partially known) environment. The RL agent obtains evaluative feedback usually known as reward or cost for its actions at each time step, allowing it to improve the performance of subsequent actions Sutton & Barto (2018). With the advent of deep learning, RL has witnessed huge successes in recent times Silver et al. (2017). However, since most of these methods rely on model-free RL, there are several unsolved challenges, which restrict the use of these algorithms for many safety critical physical systems Vamvoudakis et al. (2015); Benosman (2018). For example, it is very difficult for most model-free RL algorithms to ensure basic properties like stability of solutions, robustness with respect to model uncertainties, etc. This has led to several research directions which study incorporating robustness, constraint satisfaction, and safe exploration during learning for safety critical applications. While robust constraint satisfaction and stability guarantees are highly desirable properties, they are also very challenging to incorporate in RL algorithms. The main goal of our work is to formulate this incorporation into robust constrained-MDPs (RCMDPs), and derive corresponding theories necessary to solve them.

Constrained Markov Decision Processes (CMDPs) are a super class of MDPs that incorporate expected cumulative cost constraints Altman (2004). Several solution methods have been proposed in the literature for solving CMDPs: trust region based methods Achiam et al. (2017), linear programming-based solutions Altman (2004), surrogate-based methods Chamiea et al. (2016); Dalal et al. (arXiv:1801.08757,2018), Lagrangian methods Geibel & Wysotzki (2005); Altman (2004). We refer to these CMDPs as *non-robust*, since they do not take model uncertainties into account. On the other hand, another line of work explicitly handles model uncertainties and is known as Robust MDPs (RMDPs) Nilim & Ghaoui (2004); Wiesemann et al. (2013). RMDPs consider a set of plausible models from so called ambiguity sets. They compute solutions that can perform well even for the worst possible realization of models Russel & Petrik (2019); Wiesemann et al. (2013); Iyengar (2005). However, unlike CMDPs, these RMDPs are not capable of handling safety constraints.

Safety constraints are important in real-life applications Altman (2004). One cannot afford to risk violating some given constraints in many real-life situations. For example, in autonomous cars, there are hard safety constraints on the car velocities and steering angles Lin et al. (2018). Moreover, training often occurs on a simulated environment for many practical applications. The goal is to mitigate the sample inefficiency of model-free RL algorithms van Baar et al. (2019). The result is then transferred to the real world, typically followed by fine-tuning, a process referred to as Sim2Real. The simulator is by definition inaccurate with respect to the targeted problem, due to approximations and lack of system identification. Heuristic approaches like domain randomization van Baar et al. (2019) and meta-learning Finn et al. (2017) try to address model uncertainty in this setting, but they often are not theoretically sound. In safety critical applications, it is expected that a trained policy in simulation will offer certain guarantees about safety, when transferred to the real-world Lazaric (2012). This point is equally important in the domain of federated learning. Indeed, in recent years there have been an increasing interest in the field of federated reinforcement learning (FRL), where several agents cooperated to obtain an optimal policy, in an average sense; for example by sharing their local optimal policies with a federating central agent, and without sharing their local data with other agents. Unfortunately, when such FRL algorithms are based on simulation models of the environment, these algorithms suffer from the same pitfalls of model-based RL, in the sense that model uncertainties will unavoidably hinder the overall performance of FRL algorithm at the deployment phase. This lack of robustness issue, is sometimes referred to in FRL research community, as simulation-reality gap, e.g., Qi et al. (2021).

In light of these practical motivations, we propose to merge the two concepts of RMDPs and CMDPs, leading to a new framework we refer to as RCMDPs. The motivation is to ensure both safety and robustness. The goal of RCMDPs is to learn policies that simultaneously satisfy certain safety constraints and also perform well under worst-case scenarios. The contributions of this paper are threefold: 1) formulate the concept of RCMDPs and derive related theories, 2) propose gradient based methods to optimize the RCMDP objective, and 3) independently derive a Lyapunov-based reward shaping technique.

The chapter is organized as follows: Section 2 describes the formulation of our RCMDP framework and the objective we seek to optimize. A Lagrange-based approach is presented in Section 3 along with required gradient update formulas and corresponding policy optimization algorithms. Section 4 is dedicated to the Lyapunov stable RCMDPs and presents the idea of Lyapunov-based reward shaping. We draw concluding remarks in Section 5.

2 Problem Formulation: RCMDP concept

We consider Robust Markov Decision Processes (RMDPs) with a finite number of states $\mathcal{S} = \{1, \ldots, S\}$ and finite number of actions $\mathcal{A} = \{1, \ldots, A\}$. Every action $a \in \mathcal{A}$ is available for the decision maker to take in every state $s \in \mathcal{S}$. After taking an action $a \in \mathcal{A}$ in state $s \in \mathcal{S}$, the decision maker transitions to a next state $s' \in \mathcal{S}$ according to the *true*, but *unknown*, transition probability $p^{\star}_{s,a} \in \Delta^{\mathcal{S}}$ and receives a reward $r_{s,a,s'} \in \mathbb{R}$. We use $p_{s,a}$ to denote transition probabilities from $s \in \mathcal{S}$ and $a \in \mathcal{A}$, and condense it to refer to a transition function as $p = \left(p_{s,a}\right)_{s\in\mathcal{S},a\in\mathcal{A}} \in \left(\Delta^{\mathcal{S}}\right)^{\mathcal{S}\times\mathcal{A}}$. We condense the rewards to vectors $r_{s,a} = \left(r_{s,a,s'}\right)_{s'\in\mathcal{S}} \in \mathbb{R}^{\mathcal{S}}$ and $r = \left(r_{s,a}\right)_{s\in\mathcal{S},a\in\mathcal{A}}$.

Our RMDP setting assumes that the transition $p_{s,a}$ is chosen adversarially from an ambiguity set $\mathcal{P}_{s,a} \in \left(\Delta^{\mathcal{S}}\right)^{\mathcal{S}\times\mathcal{A}}$ for each $s \in \mathcal{S}$ and $a \in \mathcal{A}$. An ambiguity set $\mathcal{P}_{s,a}$, defined for each state $s \in \mathcal{S}$ and action $a \in \mathcal{A}$, is a set of feasible transitions quantifying the uncertainty in transition probabilities. We restrict our attention to $s, a-$rectangular ambiguity sets which simply assumes independence between transition probabilities of different state-action pairs Le Tallec (2007); Wiesemann et al. (2013). We define the L_1-norm bounded ambiguity sets around the nominal transition probability $\bar{p}_{s,a} = \mathbb{E}[p^{\star}_{s,a}|\mathcal{D}]$, for some dataset $\mathcal{D}$ as:

$$\mathcal{P}_{s,a} = \left\{p \in \Delta^{S} \,:\, \left\|p - \bar{p}_{s,a}\right\|_1 \leq \psi_{s,a}\right\},$$

where $\psi_{s,a} \geq 0$ is the budget of allowed deviations. This budget $\psi_{s,a}$ can be computed for each $s \in \mathcal{S}$, $a \in \mathcal{A}$ using Hoeffding bound Russel & Petrik (2019): $\psi_{s,a} = \sqrt{\frac{2}{n_{s,a}} \log \frac{SA2^S}{\delta}}$, where $n_{s,a}$ is the number of transitions in dataset $\mathcal{D}$ originating from state s and an action a, and δ is the confidence level. This $\psi_{s,a}$, if used to compute a policy in RMDPs, then guarantees that the computed return is a lower bound with probability δ. Note, that this is just one specific choice for the ambiguity set. Our method can be extended to any other type of ambiguity set, e.g., $L_\infty-$norm, Bayesian, weighted, sampling based, etc. We use $\mathcal{P}$ to generally refer to[3] $\mathcal{P}_\tau = \bigotimes_{s_t \in \mathcal{S}, a_t \in \mathcal{A}} \mathcal{P}_{s,a}$, where τ denotes the total number of time steps starting from $T - \tau$, with T the length of the horizon, and $t \in \{T-\tau, T-\tau+1, \ldots, T\}$. For example, with $\tau = T$ we have $\mathcal{P}_T = \bigotimes_{s_t \in \mathcal{S}, a_t \in \mathcal{A}} \mathcal{P}_{s,a}$ starting from time step 0. This collectively represents the ambiguity set along with the notion of independence between state-action pairs in a tabular setting with discrete states and actions. Sampling-based sets under approximate methods, e.g., neural networks, for large and continuous problems also extend on this similar notion of ambiguity sets Tamar et al. (2014); Derman et al. (2018).

A stationary randomized policy $\pi(\cdot|s)$ for state $s \in \mathcal{S}$ defines a probability distribution over actions $a \in \mathcal{A}$. The set of all randomized stationary policies is denoted by $\Pi \in \left(\Delta^{\mathcal{A}}\right)^{\mathcal{S}}$. We parameterize the randomized policy for state $s \in \mathcal{S}$ as $\pi_\theta(\cdot|s)$ where $\theta \subseteq \mathbb{R}^k$ is a $k-$dimensional parameter vector. Let $\xi = \{s_0, a_0, r_0, \ldots, s_{T-1}, a_{T-1}, r_{T-1}, s_T\}$ be a sampled trajectory generated by executing a policy π_θ from a starting state $s_0 \sim p_0$ under transition probabilities $p \in \mathcal{P}$, where p_0 is the distribution of initial states. Then the probability of sampling a trajectory ξ is: $p^{\pi_\theta}(\xi) = p_0(s_0) \prod_{t=0}^{T-1} \pi_\theta(a_t|s_t) p(s_{t+1}|s_t, a_t)$ and the total reward along the trajectory ξ is: $g(\xi, r) = \sum_{t=0}^{T-1} \gamma^t r_{s_t, a_t, s_{t+1}}$ Puterman (2005); Sutton & Barto (2018). The value function $v_p^{\pi_\theta} : \mathcal{S} \to \mathbb{R}$ for a policy π_θ and transition probability p is: $v_p^{\pi_\theta} = \mathbb{E}_{\xi \sim p}\left[g(\xi, r)\right]$ and the total return is:

$$\rho(\pi_\theta, p, r) = p_0^T v_p^{\pi_\theta}.$$

Because the RMDP setting considers different possible transition probabilities within the ambiguity set $\mathcal{P}$, we use a subscript p (e.g. $v_p^{\pi_\theta}$) to indicate which one is used, in case it is not clear from the context.

We define a robust value function $\hat{v}_{\mathcal{P}}^{\pi_\theta}$ for an ambiguity set $\mathcal{P}$ as: $\hat{v}_{\mathcal{P}}^{\pi_\theta} = \min_{p \in \mathcal{P}} v_p^{\pi_\theta}$. Similar to ordinary MDPs, the robust value function can be computed using the robust Bellman operator Iyengar (2005); Nilim & Ghaoui (2005):

$$(\mathfrak{T}_{\mathcal{P}} v)(s) := \max_{a \in \mathcal{A}} \min_{p \in \mathcal{P}_{s,a}} (r_{s,a} + \gamma \cdot p^\mathsf{T} v).$$

The optimal robust value function $\hat{v}^\star$, and the robust value function $\hat{v}_{\mathcal{P}}^{\pi_\theta}$ for a policy π_θ are unique and satisfy $\hat{v}^\star = \mathfrak{T}_{\mathcal{P}} \hat{v}^\star$ and $\hat{v}_{\mathcal{P}}^{\pi_\theta} = \mathfrak{T}_{\mathcal{P}}^{\pi_\theta} \hat{v}^{\pi_\theta}$ Iyengar (2005). The robust return $\hat{\rho}(\pi_\theta, \mathcal{P}, r)$ for a policy π_θ and ambiguity set $\mathcal{P}$ is defined as Nilim & Ghaoui (2005); Russel & Petrik (2019):

[3] Where $\bigotimes$ denotes the Cartesian product of sets.

$$\hat{\rho}(\pi_\theta, \mathcal{P}, r) = \min_{p \in \mathcal{P}} \rho(\pi_\theta, p, r) = p_0^T \hat{v}_{\mathcal{P}}^{\pi_\theta},$$

where p_0 is the initial state distribution.

2.1 Robust Constrained MDP (RCMDP):

In addition to rewards $r_{s,a}$ for RMDPs described above, we incorporate a constraint cost $d'_{s,a,s'} \in \mathbb{R}$, where $s, s' \in \mathcal{S}$ and $a \in \mathcal{A}$, representing some kind of constraint on safety for the agent's behavior. Consider for example an autonomous car that makes money (reward r) for each complete trip but incurs a big fine (constraint cost d) for traffic violations or a collision. We define the constraint cost $d'_{s,a,s'}$ to be a negative reward $d_{s,a,s'} = -d'_{s,a,s'}$, which brings consistency in representing the *worst-case* with a minimum over the ambiguity set $\mathcal{P}$ for both the objective and the constraint. An associated constraint budget $\beta \in \mathbb{R}_+$ describes the total budget for constraint violations. This arrangement resembles the constrained-MDP setting as described in Altman (2004), but with additional robustness.

Similar to reward based estimates described above, the total constraint cost along a trajectory ξ is: $g(\xi, d) = \sum_{t=0}^{\infty} \gamma^t d_{s_t, a_t, s_{t+1}}$, the robust value function for policy π_θ and ambiguity set $\mathcal{P}$ is: $\hat{u}^{\pi_\theta} = \min_{p \in \mathcal{P}} \mathbb{E}_{\xi \sim p}\left[g(\xi, d)\right]$ and the robust return:

$$\hat{\rho}(\pi_\theta, \mathcal{P}, d) = \min_{p \in \mathcal{P}} \rho(\pi_\theta, p, d) = p_0^T \hat{u}^{\pi_\theta}.$$

Similar to $\hat{v}^\star$, the optimal constraint value function $\hat{u}^\star$ is also unique and independently satisfies the Bellman optimality equation Altman (2004). We now formally define the objective of Robust Constrained MDP (RCMDP) as below:

$$\underset{\pi_\theta \in \Pi}{\text{maximize}} \quad \hat{\rho}(\pi_\theta, \mathcal{P}, r), \tag{1a}$$

$$\text{subject to} \quad \hat{\rho}(\pi_\theta, \mathcal{P}, d) \geq \beta. \tag{1b}$$

This objective resembles the objective of a CMDP Altman (2004), but with additional robustness integrated by the quantification of the uncertainty about the model. The interpretation of the objective is to find a policy π_θ that maximizes the worst-case return estimates, while satisfying the constraints in all possible situations.

3 Robust Constrained Optimization

A standard approach for solving the optimization problem (1) is to apply the Lagrange relaxation procedure (Bertsekas (2003), Ch.3), which turns it into an unconstrained optimization problem:

$$\mathfrak{L}(\pi_\theta, \lambda) = \hat{\rho}(\pi_\theta, \mathcal{P}, r) - \lambda\Big(\beta - \hat{\rho}(\pi_\theta, \mathcal{P}, d)\Big), \tag{2}$$

where λ is known as the *Lagrange multiplier*. Note that, the objective in (2) is non-convex and therefore is not tractable. The dual function of $\mathfrak{L}(\pi_\theta, \lambda)$ involves a point-wise maximum with respect to π_θ and is written as Paternain et al. (2019):

$$d(\lambda) = \max_{\pi_\theta \in \Pi} \mathfrak{L}(\pi_\theta, \lambda).$$

The dual function $d(\lambda)$ provides an upper bound on (2) and therefore needs to be minimized to contract the gap from optimality:

$$\mathfrak{D}^\star = \min_{\lambda \in \mathbb{R}_+} d(\lambda). \tag{3}$$

The dual problem in (3) is convex and tractable, but the question remains about how large the duality gap is. In other words, how sub-optimal the solution $\mathfrak{D}^\star$ of the dual problem (3) is with respect to the solution of the original problem stated in (1). To answer that question, Paternain et al. (2019) show that strong duality holds in this case under some mild conditions and the duality gap is arbitrarily small even with the parameterization (π_θ) of policies. We thus aim to optimize the dual version of this problem using gradients.

Proposition 1 *The relaxed RCMDP objective of (2) can be restated as:*

$$\mathfrak{L}(\pi_\theta, \lambda) = \sum_{\xi \in \Xi} p^{\pi_\theta}(\xi)\Big(g(\xi, r) + \lambda g(\xi, d)\Big) - \lambda\beta. \tag{4}$$

Proof We defer the detailed derivation to Appendix 6.1. □

The goal is then to find a saddle point $({\pi_\theta}^*, \lambda^*)$ of $\mathfrak{L}$ in (4) that satisfies $\mathfrak{L}(\pi_\theta, \lambda^*) \leq \mathfrak{L}({\pi_\theta}^*, \lambda^*) \leq \mathfrak{L}({\pi_\theta}^*, \lambda)$, $\forall \theta \in \mathbb{R}^k$ and $\forall \lambda \in \mathbb{R}_+$. This is achieved by ascending in θ and descending in λ using the gradients of objective $\mathfrak{L}$ with respect to θ and λ respectively Chow & Ghavamzadeh (2014).

Theorem 1 *The gradient of $\mathfrak{L}$ with respect to θ and λ can be computed as:*

$$\nabla_\theta \mathfrak{L}(\pi_\theta, \lambda) = \sum_{\xi} p^{\pi_\theta}(\xi)\Big(g(\xi, r) + \lambda g(\xi, d)\Big) \sum_{t=0}^{T-1} \frac{\nabla_\theta \pi_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)},$$
$$\nabla_\lambda \mathfrak{L}(\pi_\theta, \lambda) = \sum_{\xi} p^{\pi_\theta}(\xi) g(\xi, d) - \beta.$$

Proof See Appendix 6.2 for the detailed derivation. □

With a fixed Lagrange multiplier λ, the constraint budget β in (4) offsets the sum by a constant amount. We can therefore omit this constant and define the Bellman operator for RCMDPs. We then show that this operator is a contraction.

Proposition 2 *(Bellman Equation) For a fixed policy π_θ and discount factor γ, the RCMDP value function $\hat{w}^{\pi_\theta}$ satisfies a Bellman equation for each $s \in \mathcal{S}$:*

$$\hat{w}^{\pi_\theta}(s) = \min_{p \in \mathcal{P}_{s,\pi_\theta(s)}} \mathbb{E}_{s' \sim p}\left[r'_{s,\pi_\theta(s),s'} + \gamma \hat{w}^{\pi_\theta}(s')\right], \tag{5}$$

where $r'_{s,\pi_\theta(s),s'} = r_{s,\pi_\theta(s),s'} + \lambda d_{s,\pi_\theta(s),s'}$.

Proof The proof is deferred to Appendix 6.3. □

We define the Bellman optimality equation for RCMDPs as:

$$(\mathfrak{T}^{rc}_{\mathcal{P}} \hat{w})(s) := \max_{a \in \mathcal{A}} \min_{p \in \mathcal{P}_{s,a}} (r'_{s,a} + \gamma \dot{p}^{\mathsf{T}} \hat{w}). \tag{6}$$

Proposition 3 *(Contraction) The Bellman operator $\mathfrak{T}^{rc}_{\mathcal{P}}$ defined in (6) is a contraction.*

Proof The proof follows directly from Theorem 3.2 of Iyengar (2005). □

The RCMDP Bellman operator $\mathfrak{T}^{rc}_{\mathcal{P}}$ therefore satisfies the Bellman optimality equation and converges to a fixed point of the optimal RCMDP value function $\hat{w}^\star$.

3.1 Policy Gradient Algorithm:

Algorithm 1 presents a robust constrained policy gradient algorithm based on the gradient update rules derived above in Theorem 1. The algorithm proceeds in an episodic way based on trajectories and updates parameters based on the Monte-Carlo estimates. The algorithm requires an ambiguity set $\mathcal{P}$ as its input, which can be constructed with empirical estimates for smaller problems Wiesemann et al. (2013); Russel & Petrik (2019); Behzadian et al. (2021). For larger problems it can be a parameterized estimate instead Janner et al. (2019).

The step size schedules used in Algorithm 1 satisfy the standard conditions for stochastic approximation algorithms Borkar (2009). That is, θ-update is on the fastest time-scale $\zeta_2(k)$, whereas λ-update is on a slower time-scale $\zeta_1(k)$, and thus results in a two time-scale stochastic approximation algorithm. We derive its convergence to a saddle point as below.

Theorem 2 *Under assumptions **(A1)** - **(A7)** as stated in Appendix 6.5, the sequence of parameter updates of Algorithm 1 converges almost surely to a locally optimal policy $\pi_\theta{}^\star$ as the number of trajectories $k \to \infty$.*

Proof We report the proof in Appendix 6.5.1. □

Algorithm 1: Robust-Constrained Policy Gradient (RCPG) Algorithm

```
Input: A differentiable policy parameterization $\pi^{\theta}$, ambiguity set $\mathcal{P}$, confidence level $\delta$,
       step size schedules $\zeta_2$ and $\zeta_1$.
Output: Policy parameters $\theta$
Initialize policy parameter: $\theta \leftarrow \theta_0$
for $k \leftarrow 0, 1, 2, \ldots$ do
    Sample initial state $s_0 \sim p_0$, initialize trajectory: $\xi \leftarrow \emptyset$
    for $t \leftarrow 0, 1, 2, \ldots, T$ do
        Sample action: $a_t \sim \pi_\theta(\cdot|s_t)$
        Worst-case transitions with confidence $\alpha$: $\hat{p}^{\pi_\theta} \leftarrow \arg\min_{p \in \mathcal{P}_{s,a}} p^T \hat{v}^{\pi_\theta}$
        Sample next state: $s_{t+1} \sim \hat{p}^{\pi_\theta}$, observe $r_{s_t, a_t, s_{t+1}}$ and $d_{s_t, a_t, s_{t+1}}$.
        Record transition: $\xi \leftarrow \left\{ s_t, a_t, s_{t+1}, r_{s_t,a_t,s_{t+1}}, d_{s_t,a_t,s_{t+1}}, \frac{\nabla_\theta \pi_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)} \right\}$
    $\theta$-update: $\theta \leftarrow \theta + \zeta_2(k) \nabla_\theta \mathfrak{L}(\pi_\theta, \lambda)$
    $\lambda$-update: $\lambda \leftarrow \lambda - \zeta_1(k) \nabla_\lambda \mathfrak{L}(\pi_\theta, \lambda)$
return $\theta$;
```

3.2 Actor-Critic Algorithm:

The general issue of having high variance in the Monte Carlo based policy gradient algorithm can be handled by introducing state values to use as baselines Sutton & Barto (2018). As the optimal value function for RCMDPs can be computed using Bellman style recursive updates as shown in (5), an extension of the above PG algorithm to the actor-critic framework is straightforward. Algorithm (2) reported in Appendix 6.4 presents an actor-critic (AC) algorithm for RCMDPs. The state-value parameterization with f brings a new dimension in algorithm (2) and results in a three time-scale stochastic algorithm. The convergence properties for this AC algorithm can be derived in a way similar to Theorem 2 and we therefore omit the detailed derivations.

4 Stable Robust-Constrained RL: Lyapunov-based RCMDP Concept

In this section, we propose Lyapunov-based[4] reward shaping for RCMDPs. The motivation of this is threefold: i) learn a good policy faster, ii) serve as a proxy to guide robustness when an estimate for the value function is not readily available and iii) guarantee stability (in the sense of Lyapunov) in the learning process. We first briefly introduce the idea of Lyapunov stability, Lyapunov function, and some of

[4] Other works have applied different notions of Lyapunov stability in the context of model-based RL Farahmand & Benosman (2017); Berkenkamp et al. (2017) and MDPs Perkins & Barto (2001); Chow et al. (2018), however, none of these works incorporate explicit robustness in their formulation, i.e., in the context of RCMDP.

its useful characteristics. We then introduce the notion of additive shaping reward strategy based on Lyapunov functions and analyze its properties.

Definition 1 (Lyapunov stability) Haddad (2008) Consider the general nonlinear discrete system (Sy) $s_{t+1} = f(s_t)$, where $s \in D \in \mathbb{R}^n$, D is an open set containing $s^\star$, $f: D \longrightarrow D$ is a continuous function on D. Then, the equilibrium point $s^\star$ of (Sy) satisfying $s^\star = f(s^\star)$, is said to be:

- Lyapunov stable if $\forall \epsilon > 0, \exists \gamma(\epsilon) > 0$, s.t., if $\|s_0 - s^\star\| < \gamma$, then $\|s_t - s^\star\| < \epsilon, \forall t \in \mathbb{Z}_+$

- Asymptotically stable if Lyapunov stable and $\exists \gamma > 0$, s.t., if $\|s_0 - s^\star\| < \gamma$, then $\lim_{t \longrightarrow \infty} \|s_t - s^\star\| = 0$.

Definition 2 (Lyapunov direct method) Haddad (2008) Consider the system (Sy), and assume that there exists a continuous Lyapunov function $\mathbb{V}: D \longrightarrow \mathbb{R}$, s.t.,

$$\begin{cases} \mathbb{V}(s^\star) = 0 & \text{(7a)} \\ \mathbb{V}(s) > 0, \; s \in D \setminus \{s^\star\} & \text{(7b)} \\ \mathbb{V}(f(s)) - \mathbb{V}(s) \le 0, \; s \in D, & \text{(7c)} \end{cases}$$

then the equilibrium point $s^\star$ is Lyapunov stable. If, in addition $\mathbb{V}(f(s)) - \mathbb{V}(s) < 0, \; s \in D \setminus \{s^\star\}$, then $s^\star$ is asymptotically stable.

4.1 Stability Constraints for RMDPs

We propose to incorporate the Lyapunov stability descent property (7c) as a constraint in the RCMDP objective (1) , where the constraint cost is given by $d \equiv d_s = -(\mathbb{V}(s_{t+1}) - \mathbb{V}(s_t))$. We set the budget $\beta = 0$ to enforce Lyapunov stability or set $\beta > 0$ for achieving asymptotic stability. Note that in this setting, we assume that the only constraint cost is the stability cost d_s, and thus we are in the setting of RMPDs to which we add a virtual stability constraint cost.

In this setting, we apply Algorithm 1 to propose a Lyapunov stable-RCPG algorithm, and use the results of Theorem 2, to deduce its asymptotic convergence to a local optimal stationary policy for the infinite horizon case. We summarize this in the following proposition.

Proposition 4 *Under assumptions **(A1)** - **(A7)** as stated in 6.5, the sequence of parameter updates of Algorithm 1, where* $d \equiv d_s$, $\beta = 0$, *converges almost surely to a locally optimal a.s. Lyapunov stable policy* $\theta^\star$ *as* $k \longrightarrow \infty$. *Furthermore, if* $\beta > 0$, *the policy is a.s. asymptotically stable.*

Proof Consider the control problem defined by (1), under assumptions **(A1)** - **(A7)**, and where $d \equiv d_s = -(\mathbb{V}(s_{t+1}) - \mathbb{V}(s_t))$. Then, based on Theorem 2, we can conclude that Algorithm 1, converges asymptotically almost surely to a local optimal policy $\theta^\star$. Furthermore, since $\theta^\star$ is computed under the constraint of Lyapunov descent property in expectation, the equilibrium point of the controlled system is a.s.[5] Lyapunov stable (Definition 3.5, Mahmoud et al. (2003)) when $\beta = 0$, and a.s. asymptotically Lyapunov stable (Definition 3.8, Mahmoud et al. (2003)) when $\beta > 0$. □

Remark 1 The convergence speed of the Lyapunov-RL algorithm defined in Proposition 4 is correlated with the Lyapunov descent constraint (7c). If the Lyapunov descent gap $\mathbb{V}(s_{t+1}, \pi(s_{t+1})) - \mathbb{V}(s_t, \pi(s_t))$ is large, the RL algorithm will reach the optimal (Lyapunov stable) policy faster. Unfortunately, the choice of the 'best' Lyapunov function is a difficult problem, and remains a case-by-case design problem, often informed by the physics of the controlled system.

4.2 Stability Constraints for RCMDPs

In the case where the problem at hand is an RCMPD with a constraint cost d (e.g. physical obstacle avoidance constraints for a mobile robot). We take the parallel between the notions of soft constraints, where the Lyapunov descent constraints is not enforced as a constraint cost as in Sec. 4.1, and reward shaping Ng et al. (1999). Indeed, we propose to add the Lyapunov stability descent constraint directly to the reward r of the RCMDP (1).

4.3 Reward Shaping with Lyapunov Constraint:

We define the shaping reward function $f_{s,a,s'} \to \mathbb{R}$ based on this Lyapunov descent property.

$$f_{s,a,s'} = -(\mathbb{V}(s') - \mathbb{V}(s)) \tag{8}$$

The motivation behind this is quite intuitive: a transition towards descend direction leads to a desired region of the state space faster and therefore should be rewarded. So, if we were to receive a reward $r_{s,a,s'}$ in the original setting, we instead would pretend to receive a reward of $r_{s,a,s'} + f_{s,a,s'}$ on the same event. This renders a transformed RCMDP $\mathfrak{M}'$ with same state space, action space and transition probabilities. Only the reward function is reshaped with additional reward signals f.

Theorem 3 *Every optimal finite-horizon policy in transformed RCMDP $\mathfrak{M}'$ is also an optimal finite-horizon policy in the original RCMDP $\mathfrak{M}$ under Lyapunov-based reward transformation stated in (8). Furthermore, under the assumption of transient MDP, every infinite-horizon policy in transformed RCMDP $\mathfrak{M}'$ is also an optimal finite-horizon policy in the original RCMDP $\mathfrak{M}$.*

Proof In the finite-horizon case, this result is a simple extension of Theorem 1 of Ng et al. (1999) into the RCMDP setting and the proof follows directly from Ng et al.

[5] Almost surely–a.s.–(asymptotic) Lyapunov stability is to be understood as (asymptotic) Lyapunov stability for almost all samples of the states.

(1999). In the infinite-horizon case, one needs to rely on the transient assumption for the MDP (in the sense of Def. 7.1 in Altman (2004)) to conclude about the convergence of the finite-horizon problem to the infinite-horizon problem, using the arguments in (Theorem 15.1, Altman (2004)) . See 7.1 for the full derivation. □

Remark 2 Note that the concept of Lyapunov reward transformation is independent of the RL algorithm, and thus can be applied with any existing mainstream approaches such as TRPO, PPO, or CPO. The Lyapunov reward transformation will allow faster convergence for these existing approaches.

5 Conclusion

In this chapter, we introduced the concept of robust constrained MDPs (RCMDPs) to simultaneously deal with constraints and model uncertainties in reinforcement learning, which has clear applications in sim2real transfer RL and in closing the simulation-reality gap in federated RL, e.g., Qi et al. (2021). We proposed the RCMDP framework, derived related theoretical analysis and proposed algorithms to optimize the objective of RCMDPs. We also proposed an extension to Lyapunov-RCMDPs (L-RCMDPs) for RCMDPs based on the Lyapunov stability theory. We analyzed the performance of our L-RCMDP algorithms in the context of reward-shaping. We provided theoretical analysis of Lyapunov stability and asymptotic convergence for our methods. Future work should focus on automated learning of the Lyapunov function from the domain itself, quantification of the Lyapunov induced acceleration in the learning, and application of the proposed approach to some practical problem domains.

Acknowledgements If you want to include acknowledgments of assistance and the like at the end of an individual chapter please use the `acknowledgement` environment – it will automatically be rendered in line with the preferred layout.

References

Achiam, J., Held, D., Tamar, A., and Abbeel, P. Constrained Policy Optimization. *International Conference on Machine Learning*, 2017.

Altman, E. Constrained Markov Decision Processes. 2004.

Behzadian, B., Russel, R. H., Petrik, M., and Ho, C. P. Optimizing Percentile Criterion Using Robust MDPs. *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2021.

Benosman, M. Model-based vs data-driven adaptive control: An overview. *International Journal of Adaptive Control and Signal Processing*, 2018.

Berkenkamp, F., Turchetta, M., Schoellig, A. P., and Krause, A. Safe model-based reinforcement learning with stability guarantees. In *31st Conference on Neural Information Processing Systems*, 2017.

Bertsekas, D. P. *Nonlinear programming*. Athena Scientific, 2003.

Borkar, V. S. Stochastic Approximation: A Dynamical Systems Viewpoint. *International Statistical Review*, 2009.

Chamiea, M. E., Yu, Y., and Acikmese, B. Convex synthesis of randomized policies for controlled markov chains with density safety upper bound constraints. In *IEEE American Control Conference*, pp. 6290–6295, 2016.

Chow, Y. and Ghavamzadeh, M. Algorithms for CVaR optimization in MDPs. *Advances in Neural Information Processing Systems*, 2014.

Chow, Y., Nachum, O., Duenez-Guzman, E., and Ghavamzadeh, M. A lyapunov-based approach to safe reinforcement learning. *Advances in Neural Information Processing Systems*, 2018.

Dalal, G., Dvijotham, K., Vecerik, M., Hester, T., Paduraru, C., and Tassa, Y. Safe exploration in continuous action spaces, arXiv:1801.08757,2018.

Derman, E., Mankowitz, D. J., Mann, T. A., and Mannor, S. Soft-robust actor-critic policy-gradient. *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2018.

Farahmand, A.-M. and Benosman, M. Towards stability in learning based control: A bayesian optimization based adaptive controller. In *The Multi-disciplinary Conference on Reinforcement Learning and Decision Making*, 2017.

Finn, C., Yu, T., Zhang, T., Abbeel, P., and Levine, S. In Levine, S., Vanhoucke, V., and Goldberg, K. (eds.), *One-Shot Visual Imitation Learning via Meta-Learning*, 2017.

Geibel, P. and Wysotzki, F. Risk-sensitive reinforcement learning applied to control under constraints. *Journal of Artificial Intelligence Research*, 2005.

Haddad, W. M. *Nonlinear dynamical systems and control: A Lyapunov-based approach*. Princeton University Press, 2008.

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

Janner, M., Fu, J., Zhang, M., and Levine, S. When to trust your model: Model-based policy optimization. *arXiv:1906.08253*, 2019.

Lazaric, A. *Transfer in reinforcement learning: a framework and a survey. In Reinforcement Learning*. Springer, 2012.

Le Tallec, Y. *Robust, Risk-Sensitive, and Data-driven Control of Markov Decision Processes*. PhD thesis, MIT, 2007.

Lin, S. C., Zhang, Y., Hsu, C. H., Skach, M., Haque, M. E., Tang, L., and Mars, J. The architectural implications of autonomous driving: Constraints and acceleration. *ACM SIGPLAN Notices*, 2018.

Mahmoud, M. M., Jiang, J., and Zhang, Y. *Active Fault Tolerant Control Systems: Stochastic Analysis and Synthesis*. Springer, 2003.

Ng, A., Harada, D., and Russell, S. Policy invariance under reward transformations: Theory and application to reward shaping. In *International Conference on Machine Learning*, 1999.

Nilim, A. and Ghaoui, L. E. Robust solutions to Markov decision problems with uncertain transition matrices. *Operations Research*, 53(5):780, 2004.

Nilim, A. and Ghaoui, L. E. Robust control of Markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, sep 2005. ISSN 0030-364X. doi: 10.1287/opre.1050.0216.

Paternain, S., Chamon, L. F., Calvo-Fullana, M., and Ribeiro, A. Constrained reinforcement learning has zero duality gap. *Conference on Neural Information Processing Systems*, 2019.

Perkins, T. J. and Barto, A. G. Lyapunov-Constrained Action Sets for Reinforcement Learning. *International conference on machine learning*, 2001.

Puterman, M. L. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2005.

Qi, J., Zhou, Q., Lei, L., and Zheng, K. Federated reinforcement learning: Techniques, applications, and open challenges. *arXiv:2108.11887v2*, 2021.

Russel, R. H. and Petrik, M. Beyond Confidence Regions: Tight Bayesian Ambiguity Sets for Robust MDPs. *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the game of go without human knowledge. *Nature*, 2017.

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

Szepesvári, C. *Algorithms for Reinforcement Learning*. Morgan & Claypool Publishers, 2010.

Tamar, A., Glassner, Y., and Mannor, S. Optimizing the CVaR via Sampling. In *AAAI Conference on Artificial Intelligence*, 2014.

Vamvoudakis, K., Antsaklis, P., Dixon, W., Hespanha, J., Lewis, F., Modares, H., and Kiumarsi, B. Autonomy and machine intelligence in complex systems: A tutorial. In *American Control Conference (ACC)*, 2015.

van Baar, J., Sullivan, A., Corcodel, R., Jha, D., Romeres, D., and Nikovski, D. N. Sim-to-real transfer learning using robustified controllers in robotic tasks involving complex dynamics. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

Wiesemann, W., Kuhn, D., and Rustem, B. Robust Markov decision processes. *Mathematics of Operations Research*, 2013.

Appendix

6 RCMDP Derivations

6.1 Proof of Proposition 1

We rewrite the objective (2) and perform some algebraic manipulation as below:

$$\begin{aligned}
\mathfrak{L}(\pi_\theta,\lambda) &= \hat{\rho}(\pi_\theta,\mathcal{P},r) - \lambda\Big(\beta - \hat{\rho}(\pi_\theta,\mathcal{P},d)\Big) \\
&\overset{(a)}{=} \min_{p\in\mathcal{P}} \mathbb{E}_{\xi_1\sim p}\left[g(\xi_1,r)\right] - \lambda\Big(\beta - \min_{q\in\mathcal{P}} \mathbb{E}_{\xi_2\sim q}\left[g(\xi_2,d)\right]\Big) \\
&\overset{(b)}{=} \mathbb{E}_{\xi_1\sim\tilde{p}}\left[g(\xi_1,r)\right] + \lambda\mathbb{E}_{\xi_2\sim\tilde{q}}\left[g(\xi_2,d)\right] - \lambda\beta \\
&= \sum_{\xi_1\in\Xi_{\tilde{p}}} p^{\pi_\theta}(\xi_1)g(\xi_1,r) + \lambda \sum_{\xi_2\in\Xi_{\tilde{q}}} p^{\pi_\theta}(\xi_2)g(\xi_2,d) - \lambda\beta
\end{aligned}$$

Where $\Xi_{\tilde{p}}$ is the set of all possible trajectories induced by policy π_θ under transition function $\tilde{p}$. Similarly, $\Xi_{\tilde{q}}$ is the set of all possible trajectories induced by policy π_θ under transition function $\tilde{q}$. Step (a) above follows by assuming that the initial state distribution p_0 concentrates all of its mass to one single state s_0. And (b) follows with $\tilde{p} = \arg\min_{p\in\mathcal{P}} \mathbb{E}_{\xi_1\sim p}\left[g(\xi_1,r)\right]$ and $\tilde{q} = \arg\min_{q\in\mathcal{P}} \mathbb{E}_{\xi_2\sim q}\left[g(\xi_2,d)\right]$. Note that, $\tilde{p}$ and $\tilde{q}$ are distinct, independent and depend on rewards r and constraint costs d respectively. However, the rewards and constraint costs are coupled together in reality, meaning that the set of two trajectories $\Xi_{\tilde{p}}$ and $\Xi_{\tilde{q}}$ would not be different. So we select one set of trajectories Ξ being either $\Xi_{\tilde{p}}$ or $\Xi_{\tilde{q}}$. This selection of Ξ may happen based on our priorities toward robustness of reward r (with corresponding trajectory $\Xi_{\tilde{p}}$) or constraint cost d (with corresponding trajectory $\Xi_{\tilde{q}}$). Or, it can also be the best (e.g. yielding higher objective value) set among $\Xi_{\tilde{p}}$ and $\Xi_{\tilde{q}}$ satisfying the constraint. We then have a simplified formulation for $\mathfrak{L}$ as below:

$$\mathfrak{L}(\pi_\theta,\lambda) = \sum_{\xi\in\Xi} p^{\pi_\theta}(\xi)\Big(g(\xi,r) + \lambda g(\xi,d)\Big) - \lambda\beta$$

6.2 Proof of Theorem 1

Proof The objective as specified in (4):

$$\mathfrak{L}(\pi_\theta,\lambda) = \sum_{\xi\in\Xi} p^{\pi_\theta}(\xi)\Big(g(\xi,r) + \lambda g(\xi,d)\Big) - \lambda\beta$$

We first derive the gradient update rule of $\mathfrak{L}(\pi_\theta,\lambda)$ with respect to θ as below:

$$\begin{aligned}
\nabla_\theta \mathfrak{L}(\pi_\theta, \lambda) &= \sum_{\xi \in \Xi} \nabla_\theta p^{\pi_\theta}(\xi) \Big(g(\xi, r) + \lambda g(\xi, d) \Big) \\
&= \sum_{\xi \in \Xi} p^{\pi_\theta}(\xi) \Big(g(\xi, r) + \lambda g(\xi, d) \Big) \nabla_\theta \log p^{\pi_\theta}(\xi) \\
&= \sum_{\xi \in \Xi} p^{\pi_\theta}(\xi) \Big(g(\xi, r) + \lambda g(\xi, d) \Big) \nabla_\theta \log \left(p_0(s_0) \prod_{t=0}^{T-1} p(s_{t+1} | s_t, a_t) \pi_\theta(a_t | s_t) \right) \\
&= \sum_{\xi \in \Xi} p^{\pi_\theta}(\xi) \Big(g(\xi, r) + \lambda g(\xi, d) \Big) \nabla_\theta \left(\log p_0(s_0) + \sum_{t=0}^{T-1} \log p(s_{t+1} | s_t, a_t) + \log \pi_\theta(a_t | s_t) \right) \\
&= \sum_{\xi \in \Xi} p^{\pi_\theta}(\xi) \Big(g(\xi, r) + \lambda g(\xi, d) \Big) \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t | s_t) \\
&= \sum_{\xi \in \Xi} p^{\pi_\theta}(\xi) \Big(g(\xi, r) + \lambda g(\xi, d) \Big) \sum_{t=0}^{T-1} \frac{\nabla_\theta \pi_\theta(a_t | s_t)}{\pi_\theta(a_t | s_t)}
\end{aligned}$$

Next, we derive the gradient update rule for $\mathfrak{L}(\pi_\theta, \lambda)$ with respect to λ:

$$\begin{aligned}
\nabla_\lambda \mathfrak{L}(\pi_\theta, \lambda) &= \nabla_\lambda \left(\sum_{\xi \in \Xi} p^{\pi_\theta}(\xi) \Big(g(\xi, r) + \lambda g(\xi, d) \Big) - \lambda \beta \right) \\
&= \sum_{\xi \in \Xi} p^{\pi_\theta}(\xi) g(\xi, d) - \beta
\end{aligned}$$

6.3 Proof of Proposition 2

Proof

$$\begin{aligned}
\hat{w}^{\pi_\theta}(s) &= \min_{p \in \mathcal{P}_T} \mathbb{E}_{\xi \sim p}\big[g(\xi, r) + \lambda g(\xi, d)\big] \\
&\overset{(a)}{=} \min_{p \in \mathcal{P}_T} \mathbb{E}_{\xi \sim p}\Big[r_{s,\pi_\theta(s),s'} + \gamma r_{s',\pi_\theta(s'),s''} + \gamma^2 r_{s'',\pi_\theta(s''),s'''} \ldots \\
&\qquad\qquad + \lambda\big(d_{s,\pi_\theta(s),s'} + \gamma d_{s',\pi_\theta(s'),s''} + \gamma^2 d_{s'',\pi_\theta(s''),s'''} + \ldots\big) | \xi\Big] \\
&= \min_{p \in \mathcal{P}_T} \mathbb{E}_{\xi \sim p}\Big[\big(r_{s,\pi_\theta(s),s'} + \lambda d_{s,\pi_\theta(s),s'}\big) + \gamma\big(r_{s',\pi_\theta(s'),s''} + \lambda d_{s',\pi_\theta(s'),s''}\big) \\
&\qquad\qquad + \gamma^2\big(r_{s'',\pi_\theta(s''),s'''} + \lambda d_{s'',\pi_\theta(s''),s'''}\big) + \ldots | \xi\Big] \\
&= \min_{p \in \mathcal{P}_T} \mathbb{E}_{\xi \sim p}\Big[r'_{s,\pi_\theta(s),s'} + \gamma r'_{s',\pi_\theta(s'),s''} + \gamma^2 r'_{s'',\pi_\theta(s''),s'''} + \ldots | \xi\Big] \\
&\overset{(b)}{=} \min_{p \in \mathcal{P}_{s,\pi_\theta(s)}} \mathbb{E}_{s' \sim p}\Big[r'_{s,\pi_\theta(s),s'} + \gamma \min_{p \in \mathcal{P}_{T-1}} \mathbb{E}_{\xi' \sim p}\big[r'_{s',\pi_\theta(s'),s''} + \gamma r'_{s'',\pi_\theta(s''),s'''} + \ldots | \xi'\big]\Big] \\
&= \min_{p \in \mathcal{P}_{s,\pi_\theta(s)}} \mathbb{E}_{s' \sim p}\Big[r'_{s,\pi_\theta(s),s'} + \gamma \hat{w}^{\pi_\theta}(s')\Big]
\end{aligned}$$

Here (a) follows by expanding total return given a trajectory ξ and (b) follows by evaluating the one-step immediate transition apart. □

6.4 Actor-Critic Algorithm

Algorithm 2: Robust Constrained Actor Critic (RC-AC) Algorithm

```
Input: A differentiable policy parameterization $\pi_\theta$, a differentiable state-value function
       $w^{\pi_\theta}(s, f)$, confidence level $\alpha$, step size schedule $\zeta_1$, $\zeta_2$ and $\zeta_3$.
Output: Policy parameters $\theta$
Initialize policy parameter $\theta \in \mathbb{R}^k$ and state-value weights $f \in \mathbb{R}^{k'}$;
for $j \leftarrow 0, 1, 2, \ldots$ do
    Sample initial state $s_0 \sim p_0$, set time-step $t \leftarrow 0$;
    while $s_t$ is not terminal do
        Sample action: $a_t \sim \pi_\theta(\cdot|s_t)$
        Worst-case transitions with confidence $\alpha$: $\hat{p}^{\pi_\theta} \leftarrow \arg\min_{p \in \mathcal{P}_{s,a}} p^T w^{\pi_\theta}$
        Sample next state $s_{t+1} \sim \hat{p}^{\pi_\theta}$ and observe $r_{s_t,a_t,s_{t+1}}$ and $d_{s_t,a_t,s_{t+1}}$;
        TD error: $\delta_t \leftarrow r'_{s_t,a_t,s_{t+1}} + \gamma w^{\pi_\theta}(s_{t+1}, f) - w^{\pi_\theta}(s_t, f)$;
        $f$ update: $f \leftarrow f + \zeta_1(k)\delta_t \nabla_f w^{\pi_\theta}(s_t, f)$;
        $\theta$ update: $\theta \leftarrow \theta + \zeta_2(k)\delta_t \nabla_\theta \mathfrak{L}(\pi_\theta, \lambda)$;
        $\lambda$ update: $\lambda \leftarrow \lambda - \zeta_3(k)\nabla_\lambda \mathfrak{L}(\pi_\theta, \lambda)$;
        $t \leftarrow t + 1$;
return $\theta$ ;
```

6.5 Convergence Analysis of RCPG Algorithm

Assumptions

(A1) For any state s, policy $\pi_\theta(.|s)$ is continuously differentiable with respect to parameter θ and $\nabla_\theta \pi_\theta(.|s)$ is a Lipschitz function in θ for every $s \in \mathcal{S}$ and $a \in \mathcal{A}$.

(A2) The step size schedules $\{\zeta_2(t), \zeta_1(t)\}$ satisfy:

$$\sum_t \zeta_1(t) = \sum_t \zeta_2(t) = \sum_t \zeta_3(t) = \infty \tag{9}$$

$$\sum_t \zeta_1(t)^2, \sum_t \zeta_2(t)^2 \leq \infty \tag{10}$$

$$\zeta_1(t) = o\big(\zeta_2(t)\big) \tag{11}$$

These assumptions are basically standard step-size conditions for stochastic approximation algorithms (Borkar, 2009). Equation (9) ensures that the discretization covers the entire time axis. (10) ensures that the errors resulting from the discretization of the Ordinary Differential Equation (ODE) and errors due to the noise both becomes negligible asymptotically with probability one (Borkar, 2009). Equations (9) and (10) together ensure that the iterates asymptotically capture the behavior of the ODE. (11) mandates that, updates corresponding to $\zeta_1(t)$ are on a slower time scale than $\zeta_2(t)$.

6.5.1 Policy Gradient Algorithm

The general stochastic approximation scheme used by Borkar (2009) is of the form:

$$x_{t+1} = t_n + a(t)[h(x_t) + \Delta_{t+1}] \tag{12}$$

where $\{\Delta_t\}$ are a sequence of integrable random variables representing the noise sequence and $\{a_t\}$ are step sizes (e.g. $\zeta(t)$). The expression $h(x_t) + \Delta_{t+1}$ inside the square bracket is the noisy measurement where $h(x_t)$ and Δ_{t+1} are not separately available, only their sum is available. The terms of (12) need to satisfy below additional assumptions:

(A3)

The function $h : \mathbb{R}^d \to \mathbb{R}^d$ is Lipschitz. That is $\|h(x) - h(y)\| \leq L\|x - y\|$ for some $0 \leq L \leq \infty$.

(A4)

$\{\Delta_t\}$ are martingale difference sequence:

$$\mathbb{E}[\Delta_{t+1}|x_n, \Delta_n, n \leq t] = 0$$

In addition to that, $\{\Delta_t\}$ are square-integrable:

$$\mathbb{E}[\|\Delta_{t+1}\|^2|x_n, \Delta_n, n \leq t] \leq K(1 + \|x_t\|^2) \text{ a.s. for } t \geq 0,$$

and for some constant $K > 0$.

Our proposed policy gradient algorithm is a two time-scale stochastic approximation algorithm. The parameter update iterations of the policy gradient algorithm are defined as below:

$$\theta_{t+1} = \theta_t + \zeta_2(t)\nabla_\theta \mathfrak{L}(\pi_\theta, \lambda) \tag{13}$$

$$\lambda_{t+1} = \lambda_t - \zeta_1(t)\nabla_\lambda \mathfrak{L}(\pi_\theta, \lambda) \tag{14}$$

These gradient update rules defined in (13) and (14) are in a special form as:

$$x_{t+1} = x_t + a(t)f(x_t, \epsilon_t), t \geq 0 \tag{15}$$

Where $\{\epsilon\}$ is a zero mean i.i.d. random variable representing noise.

To apply general convergence analysis techniques derived for (12) in Borkar (2009), we take the special form in (15) and transform it to the general format of (12) as below:

$$h(x) = \mathbb{E}\big[f(x, \epsilon_1)\big] \text{ and } \Delta_{n+1} = f(x_n, \epsilon_{n+1}) - h(x_n) \tag{16}$$

With these transformation techniques, we obtain the general update for θ from (13):

θ update:

$$\theta_{t+1} = \theta_t + \zeta_2(t)\big[h(\theta_t, \lambda_t) + \Delta^{(1)}_{t+1}\big] \tag{17}$$

where, $f^{(1)}(\theta_t, \lambda_t) = \nabla_\theta L(\pi_\theta, \lambda)$ is the gradient w.r.t θ, $h(\theta_t, \lambda_t) = \mathbb{E}[f^{(1)}(\theta_t, \lambda_t)]$, and $\Delta^{(1)}_{t+1} = f^{(1)}(\theta_t, \lambda_t) - h(\theta_t, \lambda_t)$. Note that, the noise term ϵ is omitted because the noise is inherent in our sample based iterations.

Proposition 5 *$h(\theta_t, \lambda_t)$ is Lipschitz in θ.*

Proof Recall that the gradient of $\mathfrak{L}(\pi_\theta, \lambda)$ with respect to θ is:

$$\nabla_\theta \mathfrak{L}(\pi_\theta, \lambda) = \sum_{\xi \in \Xi} p^{\pi_\theta}(\xi)\Big(g(\xi, r) + \lambda g(\xi, d)\Big) \sum_{t=0}^{T-1} \frac{\nabla_\theta \pi_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)} \tag{18}$$

Assumption (A1) implies that, $\nabla_\theta \pi_\theta(a_t|s_t)$ in the equation (18) is a Lipschitz function in θ for any $s \in \mathcal{S}$ and $a \in \mathcal{A}$. As the expectation of sum of $|T|$ number of Lipschitz functions is also Lipschitz, we conclude that $h(\theta_t, \lambda_t)$ is Lipschitz in θ.

Proposition 6 $\Delta^{(1)}_{t+1}$ *of (17) satisfies assumption (A4).*

We transform our update rule of (14) as:

λ update:

$$\lambda_{t+1} = \lambda_t - \zeta_1(t)\big[g(\theta_t, \lambda_t) + \Delta^{(2)}_{t+1}\big] \tag{19}$$

where, $f^{(2)}(\theta_t, \lambda_t) = \nabla_\lambda L(\pi_\theta, \lambda)$ is the gradient w.r.t λ, $g(\theta_t, \lambda_t) = \mathbb{E}_M[f^{(2)}(\theta_t, \lambda_t)]$, and $\Delta^{(2)}_{t+1} = f^{(2)}(\theta_t, \lambda_t) - h(\theta_t, \lambda_t)$.

Notice that $\nabla_\lambda \mathfrak{L}(\pi_\theta, \lambda) = \sum_\xi \hat{p}^\theta(\xi) g(\xi, d) - \beta$ is a constant function of λ. And therefore, $g(\theta_t, \lambda_t)$ is a constant function of λ.

Proposition 7 $\Delta^{(2)}_{t+1}$ *of (19) satisfies assumption (A4).*

We now focus on the singularly perturbed ODE obtained from (17) and (19).

$$\dot{\theta} = \zeta_2(t) h(\theta_t, \lambda_t) \tag{20}$$

$$\dot{\lambda} = -\zeta_1(t) g(\theta_t, \lambda_t) \tag{21}$$

With assumption (A2), $\lambda(\cdot)$ is quasi-static from the perspective of $\theta(\cdot)$ turning (20) into an ODE. where λ is held fixed:

$$\dot{\theta} = \zeta_2(t) h(\theta_t, \lambda) \tag{22}$$

We additionally assume that:

(A5)

(22) has a globally asymptotically stable equilibrium $x(\lambda)$ such that x is a Lipschitz map.

Assumption (A5) turns (21) into:

$$\dot{\lambda}(t) = g(x(\lambda_t), \lambda_t) \tag{23}$$

Let's further assume that:

(A6)

The ODE (23) has a globally asymptotically stable equilibrium $\lambda^\star$.

(A7)

$\sup_t(\|\theta_t\| + \|\lambda_t\|) < \infty$ almost surely.

Proof of Theorem 2

Proof Above are the necessary conditions to apply Theorem 2 from chapter 6 of Borkar (2009), which shows that $(\theta_t, \lambda_t) \to (x(\lambda^\star), \lambda^\star)$. Now the saddle point theorem assures that $\theta^\star = x(\lambda^\star)$ maximizes the Lagrange optimization problem stated in (4). □

7 Reward Shaping in RCMDPs

7.1 Proof of Theorem 3

Proof The robust optimal q-function satisfy the robust Bellman equation for the original RCMDP $\mathfrak{M}$:

$$\hat{q}^\star_{\mathfrak{M}}(s,a) = \min_{p \in \mathcal{P}_{s,a}} \mathbb{E}_{s' \sim p}\left[r'_{s,a,s'} + \gamma \max_{a' \in \mathcal{A}} \hat{q}^\star_{\mathfrak{M}}(s',a')\right]$$

Subtracting $\mathbb{V}(s)$ and some algebraic manipulation gives:

$$\begin{aligned}\hat{q}^\star_{\mathfrak{M}}(s,a) + \mathbb{V}(s) &= \min_{p \in \mathcal{P}_{s,a}} \mathbb{E}_{s' \sim p}\left[r'_{s,a,s'} - \gamma\mathbb{V}(s') + \mathbb{V}(s) + \gamma \max_{a' \in \mathcal{A}} \left(\hat{q}^\star_{\mathfrak{M}}(s',a') + \mathbb{V}(s')\right)\right] \\ &\overset{(a)}{=} \min_{p \in \mathcal{P}_{s,a}} \mathbb{E}_{s' \sim p}\left[r'_{s,a,s'} - \left(\mathbb{V}(s') - \mathbb{V}(s)\right) + \max_{a' \in \mathcal{A}} \left(\hat{q}^\star_{\mathfrak{M}}(s',a') + \mathbb{V}(s')\right)\right]\end{aligned}$$

Here (a) follows by setting $\gamma = 1$, and considering the finite-horizon setting, i.e., $g^{\pi}(\xi, r) = \sum_{t=0}^{N} \gamma^t r_{s_t, a_t, s_{t+1}}$.

We now define $\hat{q}_{\mathfrak{M}'}(s,a) \overset{\delta}{=} \hat{q}^{\star}_{\mathfrak{M}}(s,a) + \mathbb{V}(s)$ and set $f_{s,a,s'} = -\big(\mathbb{V}(s') - \mathbb{V}(s)\big)$. We therefore have:

$$\hat{q}_{\mathfrak{M}'}(s,a) = \min_{p \in \mathcal{P}_{s,a}} \mathbb{E}_{s' \sim p}\Big[\big(r'_{s,a,s'} + f_{s,a,s'}\big) + \max_{a' \in \mathcal{A}} \hat{q}_{\mathfrak{M}'}(s',a')\Big]$$

But this is exactly the Bellman equation for reward transformed RCMDP $\mathfrak{M}'$. We then have:

$$\hat{q}^{\star}_{\mathfrak{M}'}(s,a) = \hat{q}_{\mathfrak{M}'}(s,a) = \hat{q}^{\star}_{\mathfrak{M}}(s,a) + \mathbb{V}(s)$$

And the optimal policy for $\mathfrak{M}'$ satisfies:

$$\begin{aligned} \pi^{\star}_{\mathfrak{M}'}(s) &\in \operatorname*{argmax}_{a \in \mathcal{A}} \hat{q}^{\star}_{\mathfrak{M}'}(s,a) \\ &= \operatorname*{argmax}_{a \in \mathcal{A}} \hat{q}^{\star}_{\mathfrak{M}}(s,a) + \mathbb{V}(s) \\ &= \operatorname*{argmax}_{a \in \mathcal{A}} \hat{q}^{\star}_{\mathfrak{M}}(s,a) \end{aligned}$$

And is optimal for the original RCMDP $\mathfrak{M}$ as well. Similarly, it can be shown that every optimal policy of original RCMDP $\mathfrak{M}$ is also optimal for the transformed RCMDP $\mathfrak{M}'$ simply by following exactly same steps as shown above, but with shaping function $-f_{s,a,s'}$ and the role of $\mathfrak{M}$ and $\mathfrak{M}'$ interchanged.

Next, consider the case of infinite-horizon,i.e., $\lambda < 1$. To extend the convergence result obtained for the case of finite-horizon with $\lambda = 1$ to this case, we rely on the results in Altman (2004). Indeed, under the reasonable[6] assumption of transient MDPs (Def. 7.1, p. 75, Altman (2004)), we can conclude, in our specific case of finite-state and finite-action MDPs, that our MDPs are contracting (using the argument in Altman (2004), p. 99). Next, using Theorem 7.5, Altman (2004), we conclude that our MDPs admit a uniform Lyapunov function (in the sense of Def. 7.4, p. 77, Altman (2004)). Finally, under the Slater feasibility condition, i.e., inequality (1b) satisfied, and using Theorem 15.5, p. 201, Altman (2004), we conclude that the value of the infinite-horizon problem converges to the value of the finite-horizon one. □

[6] Transient MDPs assume that the expected time we spend (under policy π) in any state s is finite.