Robust Reinforcement Learning via Adversarial training with Langevin Dynamics

Parameswaran Kamalaruban[*]
The Alan Turing Institute
kparameswaran@turing.ac.uk

Yu-Ting Huang
EPFL
yu.huang@epfl.ch

Ya-Ping Hsieh
LIONS, EPFL
ya-ping.hsieh@epfl.ch

Paul Rolland
LIONS, EPFL
paul.rolland@epfl.ch

Cheng Shi
University of Basel
cheng.shi@unibas.ch

Volkan Cevher
LIONS, EPFL
volkan.cevher@epfl.ch

Abstract

We introduce a *sampling* perspective to tackle the challenging task of training robust Reinforcement Learning (RL) agents. Leveraging the powerful Stochastic Gradient Langevin Dynamics, we present a novel, scalable two-player RL algorithm, which is a sampling variant of the two-player policy gradient method. Our algorithm consistently outperforms existing baselines, in terms of generalization across different training and testing conditions, on several MuJoCo environments. Our experiments also show that, even for objective functions that entirely ignore potential environmental shifts, our sampling approach remains highly robust in comparison to standard RL algorithms.

1 Introduction

Reinforcement learning (RL) promise automated solutions to many real-world tasks with beyond-human performance. Indeed, recent advances in policy gradient methods [1, 2, 3, 4] and deep reinforcement learning have demonstrated impressive performance in games [5, 6], continuous control [7], and robotics [8].

Despite the success of deep RL, the progress is still upset by the fragility in real-life deployments. In particular, the majority of these methods fail to perform well when there is some difference between training and testing scenarios, thereby posting serious safety and security concerns. To this end, learning policies that are *robust* to environmental shifts, mismatched configurations, and even mismatched control actions are becoming increasingly more important.

A powerful framework to learning robust policies is to interpret the changing of the environment as an adversarial perturbation. This notion naturally lends itself to a two-player max-min problem involving a pair of agents, a protagonist and an adversary, where the protagonist learns to fulfill the original task goals while being robust to the disruptions generated by its adversary. Two prominent examples along this research vein, differing in how they model the adversary, are the Robust Adversarial Reinforcement Learning (RARL) [9] and Noisy Robust Markov Decision Process (NR-MDP) [10].

Despite the impressive empirical progress, the training of the robust RL objectives remains an open and critical challenge. In particular, [10] prove that it is in fact strictly suboptimal to directly apply (deterministic) policy gradient steps to their NR-MDP max-min objectives. Owing to the lack of a better algorithm, the policy gradient is nonetheless still employed in their experiments; similar comments also apply to [9].

[*]Work done while Parameswaran Kamalaruban and Cheng Shi were working at LIONS, EPFL.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

The main difficulty originates from the highly non-convex-concave nature of the robust RL objectives, posing significant burdens to all optimization methods. In game-theoretical terms, these methods search for pure Nash Equilibria (pure NE) which might not even exist [11]. Worse, even when pure NE are well-defined, we show that optimization methods can still get stuck at non-equilibrium stationary points on certain extremely simple non-convex-concave objectives; *cf.* Section 4.

In this paper, we contend that, instead of viewing robust RL as a max-min optimization problem, the *sampling* perspective [12] from the so-called *mixed Nash Equilibrium* (mixed NE) presents a potential solution to the grand challenge of training robust RL agents. We substantiate our claim by demonstrating the advantages of sampling algorithms over-optimization methods on three fronts:

1. We show in Section 4 that, even in stylized examples that trap the common optimization methods, the sampling algorithms can still make progress towards the optimum in expectation, even tracking the NE points.
2. We conduct extensive experiments to show that sampling algorithms consistently outperform state-of-the-arts in training robust RL agents. Moreover, our experiments on the MuJoCo dataset reveal that sampling algorithms are able to handle previous failure cases of optimization methods, such as the inverted pendulum.
3. Finally, we provide strong empirical evidence that sampling algorithms are inherently more robust than optimization methods for RL. Specifically, we apply sampling algorithms to train an RL agent with *non-robust* objective (*i.e.*, the standard expected cumulative reward maximizing objective in RL), and we compare against the policy learned by *optimizing the robust objective* (*i.e.*, the max-min formulation). Despite the disadvantage, our results show that the sampling algorithms still achieve comparable or better performance than optimization methods (*cf.* Appendix C).

2 Background

Stochastic Gradient Langevin Dynamics. For any probability distribution $p(z) \propto \exp(-g(z))$, the Stochastic Gradient Langevin Dynamics (SGLD) [13] iterates as

$$z_{k+1} \leftarrow z_k - \eta \left[\widehat{\nabla_z g(z)}\right]_{z=z_k} + \sqrt{2\eta}\epsilon\xi_k, \tag{1}$$

where η is the step-size, $\widehat{\nabla_z g(z)}$ is an unbiased estimator of $\nabla_z g(z)$, $\epsilon > 0$ is a temperature parameter, and $\xi_k \sim \mathcal{N}(0, I)$ is a standard normal vector, independently drawn across different iterations. In some cases, the convergence rate of SGLD can be improved by scaling the noise using a positive-definite symmetric matrix C. We thus define a preconditioned variant of the above update (1) as follows:

$$z_{k+1} \leftarrow z_k - \eta C^{-1} \left[\widehat{\nabla_z g(z)}\right]_{z=z_k} + \sqrt{2\eta}\epsilon C^{-\frac{1}{2}}\xi_k. \tag{2}$$

In the experiments, we use a RMSProp-preconditioned version of the SGLD [14].

Saddle Point Problems and Pure NE. Consider the following Saddle Point Problem (SPP):

$$\max_{\theta\in\mathbb{R}^n} \min_{\omega\in\mathbb{R}^m} f(\theta, \omega). \tag{3}$$

Solving (3) equals finding a point $(\theta^\star, \omega^\star)$ such that

$$f(\theta, \omega^\star) \leq f(\theta^\star, \omega^\star) \leq f(\theta^\star, \omega), \quad \forall \theta \in \mathbb{R}^n, \omega \in \mathbb{R}^m. \tag{4}$$

In the language of game theory, we say that $(\theta^\star, \omega^\star)$ is a *pure* Nash Equilibrium (pure NE). If (4) holds only locally, we say that $(\theta^\star, \omega^\star)$ is a local pure NE. A major source of SPPs is the Generative Adversarial Networks (GANs) in deep learning [15], which give rise to a variety of algorithms. However, virtually all search for a (local) pure NE; see Section 4.1.

Sampling for Mixed NE. Here, we review some of the key results from [12]. We denote the set of all probability measures on $\mathcal{Z}$ by $\mathcal{P}(\mathcal{Z})$, and the set of all functions on $\mathcal{Z}$ by $\mathcal{F}(\mathcal{Z})$. Given a (sufficiently regular) function $h : \Theta \times \Omega \to \mathbb{R}$, consider the following objective (a two-player game with mixed strategies):

$$\max_{p\in\mathcal{P}(\Theta)} \min_{q\in\mathcal{P}(\Omega)} f(p, q) := \mathop{\mathbb{E}}_{\theta\sim p}\left[\mathop{\mathbb{E}}_{\omega\sim q}[h(\theta, \omega)]\right]. \tag{5}$$

Algorithm 1 MixedNE-LD

Input: step-size $\{\eta_t\}_{t=1}^T$, thermal noise $\{\epsilon_t\}_{t=1}^T$, warmup steps $\{K_t\}_{t=1}^T$, damping factor β.
Initialize (randomly) ω_1, θ_1
for $t = 1, 2, \ldots, T-1$ **do**
  $\bar{\omega}_t, \omega_t^{(1)} \leftarrow \omega_t \; ; \; \bar{\theta}_t, \theta_t^{(1)} \leftarrow \theta_t$
  for $k = 1, 2, \ldots, K_t$ **do**
    $\theta_t^{(k+1)} \leftarrow \theta_t^{(k)} + \eta_t \nabla_\theta h \widehat{\left(\theta_t^{(k)}, \omega_t\right)} + \sqrt{2\eta_t}\epsilon_t \xi$, where $\xi \sim \mathcal{N}(0, I)$
    $\omega_t^{(k+1)} \leftarrow \omega_t^{(k)} - \eta_t \nabla_\omega h \widehat{\left(\theta_t, \omega_t^{(k)}\right)} + \sqrt{2\eta_t}\epsilon_t \xi'$, where $\xi' \sim \mathcal{N}(0, I)$
    $\bar{\omega}_t \leftarrow (1-\beta)\,\bar{\omega}_t + \beta \omega_t^{(k+1)} \; ; \; \bar{\theta}_t \leftarrow (1-\beta)\,\bar{\theta}_t + \beta \theta_t^{(k+1)}$
  end for
  $\omega_{t+1} \leftarrow (1-\beta)\,\omega_t + \beta \bar{\omega}_t \; ; \; \theta_{t+1} \leftarrow (1-\beta)\,\theta_t + \beta \bar{\theta}_t$
end for
Output: ω_T, θ_T.

A pair $(p^\star, q^\star)$ achieving the max-min value in (5) is called a *mixed Nash Equilibrium* (mixed NE).

Conceptually, problem (5) can be solved via several infinite-dimensional algorithms, such as the so-called entropic mirror descent or mirror-prox; see [12]. However, these algorithms are infinite-dimensional and require infinite computational power to implement. For practical interest, by leveraging the SGLD sampling techniques and using some practical relaxations, [12] features a simplified variant of these infinite-dimensional algorithms.

For the robust RL formulation (5), it suffices to use the simplest algorithm in [12]. The pseudocode for their resulting algorithm, termed MixedNE-LD (**mixed NE** via **L**angevin **d**ynamics), can be found in **Algorithm 1**.

3 Two-Player Markov Games

Markov Decision Process. We consider a Markov Decision Process (MDP) represented by $\mathcal{M}_1 := (\mathcal{S}, \mathcal{A}, T_1, \gamma, P_0, R_1)$, where the state and action spaces are denoted by $\mathcal{S}$ and $\mathcal{A}$ respectively. We focus on continuous control tasks, where the actions are real-valued, *i.e.*, $\mathcal{A} = \mathbb{R}^d$. $T_1 : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0, 1]$ captures the state transition dynamics, *i.e.*, $T_1(s' \mid s, a)$ denotes the probability of landing in state s' by taking action a from state s. Here γ is the discounting factor, $P_0 : \mathcal{S} \to [0, 1]$ is the initial distribution over $\mathcal{S}$, and $R_1 : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward.

Two-Player Zero-Sum Markov Games. Consider a two-player zero-sum Markov game [16, 17], where at each step of the game, both players simultaneously choose an action. The reward each player gets after one step depends on the state and the joint action of both players. Furthermore, the transition kernel of the game is controlled jointly by both players. In this work, we only consider simultaneous games, not the turn-based games.

This game can be described by an MDP $\mathcal{M}_2 = (\mathcal{S}, \mathcal{A}, \mathcal{A}', T_2, \gamma, R_2, P_0)$, where $\mathcal{A}$ and $\mathcal{A}'$ are the continuous set of actions the players can take, $T_2 : \mathcal{S} \times \mathcal{A} \times \mathcal{A}' \times \mathcal{S} \to \mathbb{R}$ is the state transition probability, and $R_2 : \mathcal{S} \times \mathcal{A} \times \mathcal{A}' \to \mathbb{R}$ is the reward for both players. Consider an agent executing a policy $\mu : \mathcal{S} \to \mathcal{A}$, and an adversary executing a policy $\nu : \mathcal{S} \to \mathcal{A}'$ in the environment $\mathcal{M}$. At each time step t, both players observe the state s_t and take actions $a_t = \mu(s_t)$ and $a_t' = \nu(s_t)$. In the zero-sum game, the agent gets a reward $r_t = R_2(s_t, a_t, a_t')$ while the adversary gets a negative reward $-r_t$.

This two-player zero-sum Markov game formulation has been used to model the following robust RL settings:

- Robust Adversarial Reinforcement Learning (RARL) [9], where the power of the adversary is limited by its action space $\mathcal{A}'$.

- Noisy Robust Markov Decision Process (NR-MDP) [10], where $\mathcal{A}' = \mathcal{A}$, $T_2(s_{t+1} \mid s_t, a_t, a_t') = T_1(s_{t+1} \mid s_t, \bar{a}_t)$, and $R_2(s_t, a_t, a_t') = R_1(s_t, \bar{a}_t)$, with $\bar{a}_t = (1-\delta)a_t + \delta a_t'$, for a chosen $\delta \in (0,1)$, which limits the adversary.

In our adversarial game, we consider the following performance objective:

$$J(\mu, \nu) \;=\; \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r_t \;\middle|\; \mu, \nu, \mathcal{M}_2\right],$$

where $\sum_{t=1}^{\infty} \gamma^{t-1} r_t$ be the random cumulative return. In particular, we consider the parameterized policies $\{\mu_\theta : \theta \in \Theta\}$, and $\{\nu_\omega : \omega \in \Omega\}$. By an abuse of notation, we denote $J(\theta, \omega) = J(\mu_\theta, \nu_\omega)$. We consider the following objective:

$$\max_{\theta \in \Theta} \min_{\omega \in \Omega} \; J(\theta, \omega). \tag{6}$$

Note that J is non-convex-concave in both θ and ω. Instead of solving (6) directly, we focus on the mixed strategy formulation of (6). In other words, we consider the set of all probability distributions over Θ and Ω, and we search for the optimal distribution that solves the following program:

$$\max_{p \in \mathcal{P}(\Theta)} \min_{q \in \mathcal{P}(\Omega)} \; f(p, q) \;:=\; \mathop{\mathbb{E}}_{\theta \sim p}\left[\mathop{\mathbb{E}}_{\omega \sim q}\left[J(\theta, \omega)\right]\right]. \tag{7}$$

Then, we can use the Algorithm 1 to solve the above problem.

4 Simple Non-Convex-Concave SPPs

In Section 3, we have formulated the robust RL problems in either its pure strategy form (6) and mixed strategy form (7). The goal of the present section is to demonstrate that solving (7) as a *sampling* problem has superior performance over methods that seek pure NE for non-convex-concave SPPs. We do so by providing theoretical and empirical justifications on several simple, yet nontrivial, low-dimensional examples. Pseudocodes for all algorithms in the section and the omitted proofs can be found in Appendix B.

Remark 1. *The goal of our examples is to show that sampling leads to a **stabler training algorithm**, rather than a **better solution concept**. In particular, whether mixed NE is more meaningful over pure NE (or any other equilibrium such as logistic stochastic best response equilibrium [18]) is outside of the scope of the this paper.*

4.1 Existing Algorithms

We will consider three algorithmic frameworks:

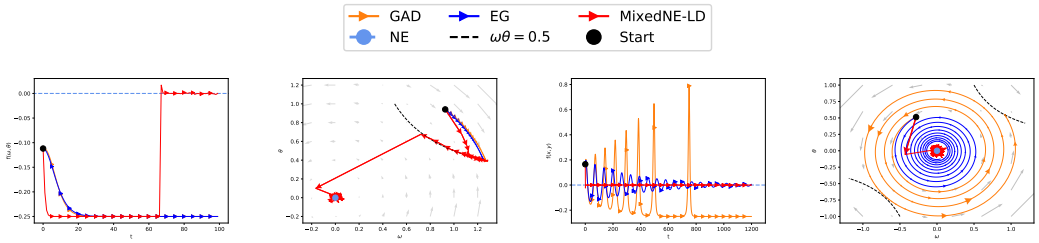
1. GAD: Finding pure NE via **G**radient **a**scent-**d**escent.
2. EG: Finding pure NE via **E**xtra-**g**radient methods.
3. MixedNE-LD: Finding mixed NE via **Algorithm 1**.

Most existing methods to solving SPPs in deep learning can be classified as (adaptive) variants of these frameworks. For instance, Adam, being an adaptive version of GAD, is the predominant algorithm when it comes to learning GANs [19], which was also employed by [10] to train robust RL agents. EG was originally developed by Korpelevich in 1976 to solve variational inequalities for convex problems, and was recently shown to outperform (adaptive) GAD when it comes to training GANs [20]. Finally, the MixedNE-LD framework was recently put forth by [12], whose defining feature is to *sample* from the mixed NE.

It is common in practice to asymptotically decrease the step-size for GAD and EG to 0. According to the theory of [21], these first-order methods with vanishing step-size behave asymptotically the same as their continuous-time counterpart (note that GAD and EG has the same continuous-time limit):

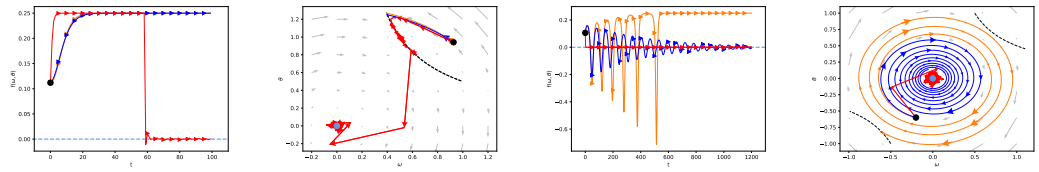
$$\begin{bmatrix} \frac{\mathrm{d}\theta}{\mathrm{d}t}(t) \\ \frac{\mathrm{d}\omega}{\mathrm{d}t}(t) \end{bmatrix} = \begin{bmatrix} \nabla_\theta f(\theta, \omega) \\ -\nabla_\omega f(\theta, \omega) \end{bmatrix} \tag{8}$$

Moreover, this result is robust to gradient noise, and so applies to stochastic variants of GAD and EG. Therefore, we will henceforth focus on (11) in our theory.



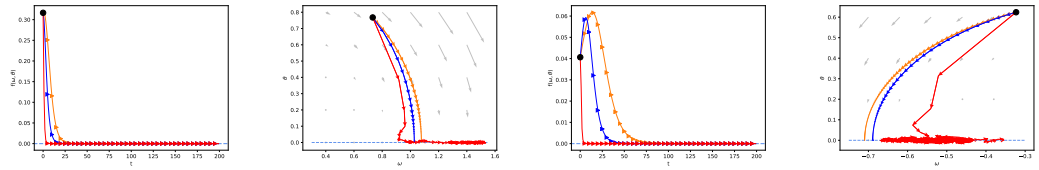
(a) $f(\theta_t, \omega_t)$, far from NE. (b) (θ_t, ω_t), far from NE. (c) $f(\theta_t, \omega_t)$, close to NE. (d) (θ_t, ω_t), close to NE.

Figure 1: $f(\theta, \omega) = \theta^2\omega^2 - \theta\omega$. The NE is $(0, 0)$ with reward value 0. The dashed curve $\theta\omega = 0.5$ describe all stationary points that are *not* NE. (a), (b) shows the objective value and the training dynamics when initializing far away from NE. (c), (d) shows the objective value and the training dynamics when (θ_1, ω_1) is initializing close to NE.



(a) $f(\theta_t, \omega_t)$, far from NE. (b) (θ_t, ω_t), far from NE. (c) $f(\theta_t, \omega_t)$, close to NE. (d) (θ_t, ω_t), close to NE.

Figure 2: $f(\theta, \omega) = \theta y - \theta^2\omega^2$. The NE is (0,0) with reward value 0. The dashed curve $\theta\omega = 0.5$ are stationary points that are *not* NE. (a), (b) shows the objective value and the training dynamics when initializing far away from NE. (c), (d) shows the objective value and the training dynamics when initializing close to NE.



(a) $f(\theta_t, \omega_t)$, far from NE. (b) (θ_t, ω_t), far from NE. (c) $f(\theta_t, \omega_t)$, close to NE. (d) (θ_t, ω_t), close to NE.

Figure 3: $f(\theta, \omega) = \theta^2\omega^2$. The NE are represented with the line $\{(\theta, 0) \mid \theta \text{ arbitrary}\}$ with reward value 0. (a), (b) shows the objective value and the training dynamics when initializing far away from NE. (c), (d) shows the objective value and the training dynamics when initializing close to NE.

4.2 Degree-2 Polynomials: Stationary Points *v.s.* NE

We now turn to the objectives. Suppose that the objective J in (6) or (7) is non-concave non-convex in d directions. Since in practice one rarely acquires information higher than second-order, we will only consider quadratic local approximations of J. Finally, let us consider optimizing each dimension separately, each leading to a 2-dimensional subproblem.

We will show, in **Theorem 1** below, that even under this extremely simplified setting, and under simple non-convexity as in (9) or (10), existing approaches can only succeed if the initialization is close enough to the equilibrium **along every direction**. As a result, the probability of successful training for existing algorithms will be exponential small in the number of non-convex non-concave directions.

We now construct nontrivial examples where there exist stationary points that are *not* NE. To this end, we may simply use the degree-2 polynomials:

$$\max_{\theta \in [-2,2]} \min_{\omega \in [-2,2]} f(\theta, \omega) = \theta^2\omega^2 - \theta\omega \tag{9}$$

and

$$\max_{\theta \in [-2,2]} \min_{\omega \in [-2,2]} f(\theta, \omega) = \theta\omega - \theta^2\omega^2. \tag{10}$$

The constraint interval $[-2, 2]$ is included only for ease of presentation; it has no impact on our conclusion. Moreover, the following facts can be readily verified:

- The pure and mixed NE are the same: $(0, 0)$.
- The curve $\{(\theta, \omega) \mid \theta\omega = 0.5\}$ presents stationary points that are *not* NE.

Consider a single-player continuous bandit problem with 1d action space $\mathcal{A} = \mathbb{R}$, state space $\mathcal{S} = \{s_0\}$, and reward function $R_1(s_0, a)$. For a policy $\pi_\theta(s_0) = \theta$, we have: $\max_a R_1(s_0, a) = \max_\theta R_1(s_0, \theta) = \max_{\pi_\theta} R_1(s_0, \pi_\theta(s_0))$. In light of this, one can easily see that Eqs. (9) and (10) correspond to two-player bandit problems with $R_2(s_0, a, a') = f(\theta, \omega)$.

4.3 Main Result

We now present the main result in this section.

Theorem 1. *Consider the (continuous-time) GAD and EG dynamics:*

$$\begin{bmatrix} \frac{\mathrm{d}\theta}{\mathrm{d}t}(t) \\ \frac{\mathrm{d}\omega}{\mathrm{d}t}(t) \end{bmatrix} = \begin{bmatrix} \nabla_\theta f(\theta, \omega) \\ -\nabla_\omega f(\theta, \omega) \end{bmatrix} \tag{11}$$

where $f(\theta, \omega)$ is either (9) *or* (10)*. (Note that GAD and EG are different discretizations of the same continuous-time process [22].) Suppose that the initial point $(\theta(0), \omega(0))$ is far away from NE: $\theta(0) \cdot \omega(0) > 0.5$. Then* (11) *converges to a non-equilibrium stationary point on $\{\theta\omega = 0.5\}$.*

On the other hand, even when initialized at a stationary point such that $\theta_1 \cdot \omega_1 = 0.5$, the MixedNE-LD still decreases the distance to NE in expectation:

$$\mathbb{E}\theta_3 \cdot \omega_3 = \theta_1\omega_1 - 4\eta^2 \left(\eta \left(\theta_1^2 + \omega_1^2\right) + 14\eta^2\right) < \theta_1 \cdot \omega_1 \tag{12}$$

where η is the step-size, and the expectation is over the randomness of the algorithm.

In words, depending on the initialization, the (continuous-time) training dynamics of GAD and EG will either get trapped by non-equilibrium stationary points, or converge to NE. In contrast, the MixedNE-LD is always able to escape non-equilibrium stationary points in expectation.

Figures 1 and 2 demonstrate the empirical behavior of the three algorithms, which is in perfect accordance with the theory. When initialized far away from NE, Figure (1a), (1b), (2a), and (2b) show that GAD and EG get trapped by local stationary points, while MixedNE-LD is able to escape after staying a few iterations near the non-equilibrium states. On the other hand, if initialized sufficiently close to NE, then EG tends to perform better than GAD, as indicated by previous work; see Figure (1c), (1d), (2c), (2d).

Finally, one can ask whether the negative results for GAD and EG are sensitive to the choice of step-size. For instance, we have implemented the vanilla GAD and EG, while in practice one always uses adaptive step-size based on approximate second-order information [23, 24]. However, our next theorem shows that, even with *perfect* second-order information, the training dynamics of GAD and EG still are unable to escape stationary points.

Theorem 2. *Consider the Newton's dynamics for solving either* (9) *or* (10)*:*

$$\begin{bmatrix} \frac{\mathrm{d}\theta}{\mathrm{d}t}(t) \\ \frac{\mathrm{d}\omega}{\mathrm{d}t}(t) \end{bmatrix} = \begin{bmatrix} \nabla_\theta^2 f(\theta, \omega) & 0 \\ 0 & \nabla_\omega^2 f(\theta, \omega)] \end{bmatrix}^{-1} \begin{bmatrix} \nabla_\theta f(\theta, \omega) \\ -\nabla_\omega f(\theta, \omega) \end{bmatrix}. \tag{13}$$

Then we have $\theta(t) \cdot \omega(t) = \theta(0) \cdot \omega(0)$.

A consequence of of **Theorem 2** is that if we initialize at any point such that $\theta(0) \cdot \omega(0) \neq 0$, the training dynamics will remain far away from $(0, 0)$, which is the desired NE. Indeed, in Section 5, we shall see that MixedNE-LD outperforms GAD and EG even with adaptivity.

4.4 A Digression: Sampling *v.s.* Optimization

We demonstrate an additional intriguing behavior of the sampling nature of MixedNE-LD, which we deem as a benefit over deterministic optimization algorithms. Consider the following SPP:

$$\max_{\theta \in [-2,2]} \min_{\omega \in [-2,2]} f(\theta, \omega) = \theta^2\omega^2. \tag{14}$$

This is a simple SPP where the stationary points $\{(\theta, 0) \mid \theta \in [-2, 2]\}$ are all NE. Consequently, both GAD and EG succeed in finding an NE, regardless of the initialization; see Figure 3. The MixedNE-LD, nonetheless, does something slightly more than finding an NE: The MixedNE-LD *explores* among all the NE, inducing a *distribution* on the set of all equilibria; see Figure (3b) and (3d).

Our theory suggests that, when there are many suboptimal stationary points, MixedNE-LD outperforms the GAD and EG baselines. We expect MixedNE-LD to perform slightly worse in the absence of this property since we can focus on converging to stationary points without adding the explorative noise in MixedNE-LD.

5 Experiments

In this section, we demonstrate the effectiveness of using the MixedNE-LD framework to solve the robust RL problem. As a case study, we consider NR-MDP setting with $\delta = 0.1$ (as recommended in Section 6.3 of [10]). This setting can cover only the changes in the transition dynamics that can be simulated via the changes in the action. In the H_∞ control literature [25, 26], an equivalence between environmental and action robustness has already been noted. The NR-MDP setting cannot handle: (i) the adversarial disturbances considered in [9], as the action spaces of both the agent and adversary are same in the NR-MDP setting; and (ii) the feature changes like style, and illumination. Nevertheless, the MixedNE-LD framework applies to general two-player Markov Games as well.

Two-Player DDPG. We design a two-player variant of DDPG [7] algorithm by adapting the Algorithm 1. As opposed to standard DDPG, in two-player DDPG two actor networks output two deterministic policies, the protagonist and adversary policies, denoted by μ_θ and ν_ω. The critic is trained to estimate the Q-function of the joint-policy. The gradients of the protagonist and adversary parameters are given in Proposition 5 of [10]. The resulting algorithm is given in Algorithm 3.

We compare the performance of our algorithm against the baseline algorithm proposed in [10] (see Algorithm 4 with GAD). [10] have suggested a training ratio of $1:1$ for actors and critic updates. Note that the action noise is injected while collecting transitions for the replay buffer. In [27], authors noted that the action noise drawn from the Ornstein-Uhlenbeck [28] process offered no performance benefits. Thus we also consider uncorrelated Gaussian noise. In addition to the baseline from [10], we have also considered another baseline, namely Algorithm 4 with Extra-Adam [20].

Setup. We evaluate the performance of Algorithm 3 and Algorithm 4 (with GAD and Extra-Adam) on standard continuous control benchmarks available on OpenAI Gym [29] utilizing the MuJoCo environment [30]. Specifically, we benchmark on eight tasks: Walker, Hopper, Half-Cheetah, Ant, Swimmer, Reacher, Humanoid, and InvertedPendulum. Details of these environments can be found in [29] and on the GitHub website.

The Algorithm 3 implementation is based on the codebase from [10]. For all the algorithms, we use a two-layer feedforward neural network structure of (64, 64, tanh) for both actors (agent and adversary) and critic. The optimizer we use to update the critic is Adam [31] with a learning rate of 10^{-3}. The target networks are soft-updated with $\tau = 0.999$. For the GAD baseline, the actors are trained with RMSProp optimizer. For our algorithm (MixedNE-LD), the actors are updated according to Algorithm 1 with warmup steps $K_t = \min\left\{15, \lfloor(1 + 10^{-5})^t\rfloor\right\}$, and thermal noise $\sigma_t = \sigma_0 \times (1 - 5 \times 10^{-5})^t$. The hyperparameters that are not related to exploration (see Table 1) are identical to all the algorithms that are compared. And we tuned only the exploration-related hyperparameters (for all the algorithms) by grid search: (a) Algorithm 3 with $(\sigma_0, \sigma) \in \left\{10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\right\} \times \{0, 0.01, 0.1, 0.2, 0.3, 0.4\}$; (b) Algorithm 4 with $\sigma \in \{0, 0.01, 0.1, 0.2, 0.3, 0.4\}$. For each algorithm-environment pair, we identified the best performing exploration hyperparameter configuration (see Tables 2 and 3). Each algorithm is trained on 0.5M samples (i.e., 0.5M time steps in the environment). We run our experiments, for each environment, with 5 different seeds. The exploration noise is turned off for evaluation.

Evaluation. We evaluate the robustness of all the algorithms under different testing conditions, and in the presence of adversarial disturbances in the testing environment. We train the algorithms with the standard mass and friction variables in OpenAI Gym. At test time, we evaluate the learned policies by changing the mass and friction values (without adversarial perturbations) and estimating the cumulative rewards. As shown in Figures 4 and 6, our Algorithm 3 outperforms the baselines

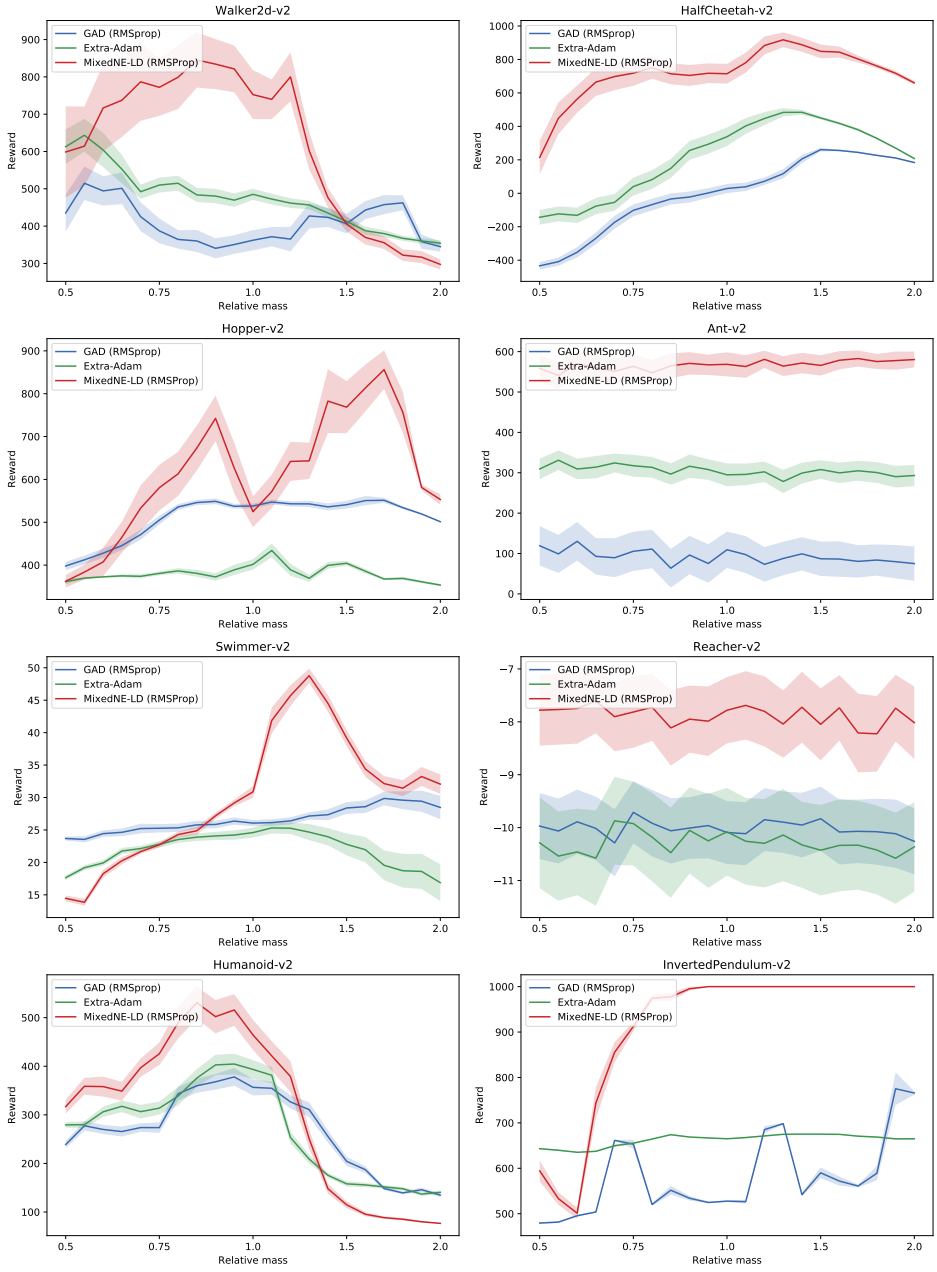


Figure 4: Average performance (over 5 seeds) of Algorithm 3 (DDPG with MixedNE-LD), and Algorithm 4 (DDPG with GAD and Extra-Adam), under the NR-MDP setting with $\delta = 0.1$. The evaluation is performed without adversarial perturbations, on a range of mass values not encountered during training.

Algorithm 4 (with GAD and Extra-Adam) in terms of robustness. Note that we obtain superior performance on the inverted pendulum, which is a failure case for [10]. We also evaluate the robustness of the learned policies under both test condition changes, and adversarial disturbances (*cf.* Appendix D).

Additional Experiments. We discuss with experimental evidence on improving the computation time of our algorithm in Appendix C. The One-Player DDPG with SGLD is a significant computational relaxation of DDPG with MixedNE-LD without compromising the empirical performance.

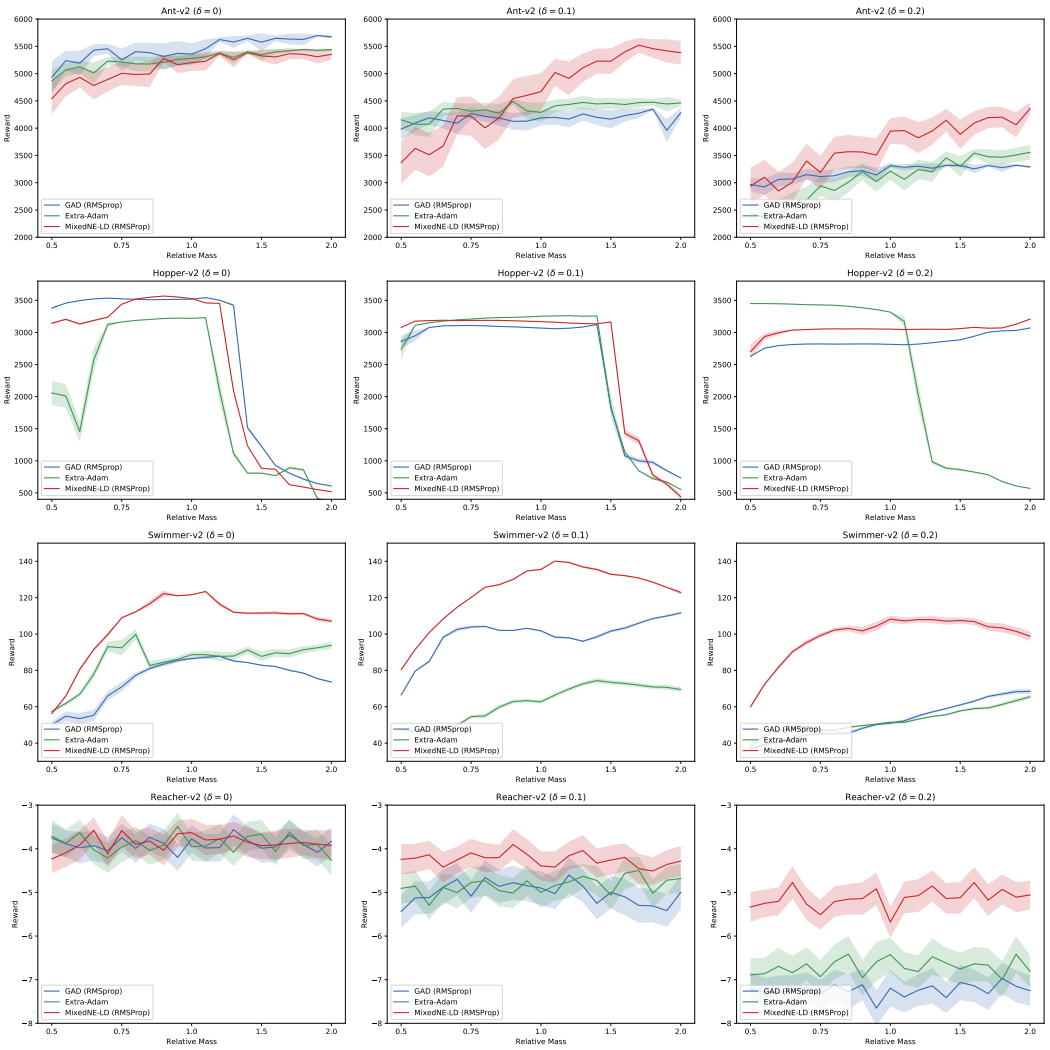


Figure 5: Average performance (over 5 seeds) of Algorithm 5 (TD3 with MixedNE-LD), and Algorithm 6 (TD3 with GAD and Extra-Adam), under the NR-MDP setting with $\delta = 0, 0.1, 0.2$. The evaluation is performed without adversarial perturbations, on a range of mass values not encountered during training.

We note that the MixedNE-LD can be adapted to any policy-gradient or actor-critic methods. We have already extended MixedNE-LD for TD3 [27] (cf. Appendix E and Figure 5) and vanilla policy gradient [32] (cf. Appendix F) methods. Also, we have compared the robust TD3 with MixedNE-LD against the non-robust SAC [33] algorithm (cf. Figures 12 and 13 in the appendix). Except for the Walker environment, our robust TD3 algorithm outperforms the SAC baseline as well.

6 Conclusion

In this work, we study the robust reinforcement learning problem. By adapting the approximate infinite-dimensional entropic mirror descent from [12], we design a robust variant of DDPG algorithm, under the NR-MDP setting. To the best of our knowledge, this is the first work to apply SGLD for the robust RL problem. In our experiments, we evaluate the robustness of our algorithm on several continuous control tasks, and found that our algorithm clearly outperforms the robust and non-robust baselines while tackling the failure case (i.e., inverted pendulum) for the earlier literature. Intriguingly, even the simple version of the algorithm with a single Langevin step results in competitive results with a desirable computational complexity.

Broader Impact

Our work would be beneficial in control/robotics applications where the training happens in a simulation environment which is only a rough estimation of the real domain where the agent will be deployed after training. Thus our adversarial training would account for this mismatch and would lead to a stable performance. Even though our adversarial training method improves robustness, being overly conservative might result in lower performance. Thus one should carefully tune the robustness related hyperparameters, in our case δ. We could not imagine any immediate negative ethical/societal impact of our work.

Acknowledgments and Disclosure of Funding

This work has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement n 725594 - time-data), the Swiss National Science Foundation (SNSF) under grant number 407540_167319, and the Army Research Office under grant number W911NF-19-1-0404.

References

[1] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

[2] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *ICML*, 2014.

[3] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015.

[4] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

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

[6] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

[7] 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*, 2015.

[8] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

[9] Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. Robust adversarial reinforcement learning. In *International Conference on Machine Learning*, 2017.

[10] Chen Tessler, Yonathan Efroni, and Shie Mannor. Action robust reinforcement learning and applications in continuous control. *arXiv preprint arXiv:1901.09184*, 2019.

[11] Partha Dasgupta and Eric Maskin. The existence of equilibrium in discontinuous economic games, i: Theory. *The Review of economic studies*, 53(1):1–26, 1986.

[12] Ya-Ping Hsieh, Chen Liu, and Volkan Cevher. Finding mixed nash equilibria of generative adversarial networks. In *International Conference on Machine Learning*, pages 2810–2819, 2019.

[13] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 681–688, 2011.

[14] Chunyuan Li, Changyou Chen, David E Carlson, and Lawrence Carin. Preconditioned stochastic gradient langevin dynamics for deep neural networks. In *AAAI*, volume 2, page 4, 2016.

[15] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[16] Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine Learning Proceedings*. Elsevier, 1994.

[17] Julien Perolat, Bruno Scherrer, Bilal Piot, and Olivier Pietquin. Approximate dynamic programming for two-player zero-sum Markov games. In *International Conference on Machine Learning*, 2015.

[18] Lantao Yu, Jiaming Song, and Stefano Ermon. Multi-agent adversarial inverse reinforcement learning. In *International Conference on Machine Learning*, pages 7194–7201, 2019.

[19] Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. Are gans created equal? a large-scale study. In *Advances in neural information processing systems*, pages 700–709, 2018.

[20] Gauthier Gidel, Hugo Berard, Gaëtan Vignoud, Pascal Vincent, and Simon Lacoste-Julien. A variational inequality perspective on generative adversarial networks. *arXiv preprint arXiv:1802.10551*, 2018.

[21] Ioannis Panageas, Georgios Piliouras, and Xiao Wang. First-order methods almost always avoid saddle points: The case of vanishing step-sizes. In *Advances in Neural Information Processing Systems*, pages 6471–6480, 2019.

[22] Jelena Diakonikolas and Lorenzo Orecchia. The approximate duality gap technique: A unified theory of first-order methods. *SIAM Journal on Optimization*, 29(1):660–689, 2019.

[23] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(Jul):2121–2159, 2011.

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

[25] John C Doyle, Bruce A Francis, and Allen R Tannenbaum. *Feedback control theory*. Courier Corporation, 2013.

[26] Jun Morimoto and Kenji Doya. Robust reinforcement learning. *Neural computation*, 17(2):335–359, 2005.

[27] Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. *arXiv preprint arXiv:1802.09477*, 2018.

[28] George E Uhlenbeck and Leonard S Ornstein. On the theory of the brownian motion. *Physical review*, 36(5):823, 1930.

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

[30] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 5026–5033. IEEE, 2012.

[31] Diederik P Kingma and Jimmy Ba. A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, volume 5, 2015.

[32] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

[33] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pages 1861–1870, 2018.

[34] Jacob Abernethy, Kevin A Lai, and Andre Wibisono. Last-iterate convergence rates for min-max optimization. *arXiv preprint arXiv:1906.02027*, 2019.

[35] Constantinos Daskalakis and Ioannis Panageas. Last-iterate convergence: Zero-sum games and constrained min-max optimization. *Innovations in Theoretical Computer Science*, 2019.

[36] Sébastien Bubeck, Michael B Cohen, Yin Tat Lee, James R Lee, and Aleksander Madry. K-server via multiscale entropic regularization. In *Proceedings of the 50th annual ACM SIGACT symposium on theory of computing*, pages 3–16, 2018.

[37] Yang Liu, Prajit Ramachandran, Qiang Liu, and Jian Peng. Stein variational policy gradient. In *Proceedings of the 33rd Conference on Uncertainty in Artificial Intelligence*, 2017.

[38] Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124018, 2019.

[39] Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, Yuhuai Wu, and Peter Zhokhov. Openai baselines. `https://github.com/openai/baselines`, 2017.