POLICY MIRROR DESCENT WITH REVERSED KL PROJECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Policy optimization is a basic problem in reinforcement learning. This paper provides Reversed Entropy Policy Mirror Descent (REPMD), achieving two properties that enhance on-line exploration: preventing early convergence to sub-optimal policies, and monotonically increasing a performance measure. REPMD adopts maximum entropy exploration within the classic mirror descent framework, and updates policy by a reversed KL projection. This approach overcomes undesirable mode seeking behaviour, while still enjoying the policy improvement guarantee. Experiments on bandit and algorithmic tasks demonstrate that the proposed method achieves better exploration than both undirected maximum entropy exploration and directed exploration with standard entropy projection.

1 INTRODUCTION

Model-free deep reinforcement learning (RL) has recently been demonstrated to successfully solve a wide range of difficult sequential decision making problems (Schulman et al., 2015; Mnih et al., 2015; Silver et al., 2016), it also significantly introduces additional complications in understanding the behavior of the model.

In practice, policy based Deep RL is different than a traditional optimization problem, in the sense that the argument to be optimized, i.e. the policy, is also used to collect training data from the environment. This interaction may lead to lack of exploration, since the learner's policy may get stuck in a local optimum and fail to collect high reward trajectories, preventing any learning from useful signals. An on-line learning algorithm should have the ability to explore the policy space properly and efficiently, to avoid getting trapped in a locally optima, while discovering the globally optimal policy quickly.

In this paper, we propose a method with a better exploration strategy, such that (a) it retains the exploration efficiency of existing methods; (b) it monotonically increases its chance of exploring trajectories generated by the optimal policies, or evolving closer to the optimal policies. Our proposed method Reversed Entropy Policy Mirror Descent (REPMD), takes both the entropy and relative entropy regularizers. Unlike common policy gradient based methods, REPMD is in a two-stage manner. REPMD first updates the policy in the entire policy-simplex, ignoring the constraint induced by its parametrization, then a projection step is performed to update the policy in the parametrized policy space. Such a two-stage update guarantees REPMD to increase performance monotonically. The proposed REPMD method is then justified from both theoretical and empirical perspectives.

The rest of the paper is organized as follows. After introducing exploration of RL in Section 2, we propose the REPMD method in Section 3. We provide the analysis for the monotonically increasing performance property of REPMD in Section 3, and conduct experiments to validate our algorithm in Section 5. Some related work is discussed in Section 4, and the conclusion and directions for future work are presented in Section 6.

The lift-and-project formulation differs our method from most of the policy search methods. On the one hand, such formulation provides an easier way in analyzing the behavior of the algorithm in parameter space. For example, the monotonical improvement guarantee can be proved in a fairly direct and simple way in this paper, even for some particular non-convex π_θ. On the other hand, such formulation suggests to perform multiple steps of gradient descent on the project step with the current

policy fixed as the reference policy. As shown in Section 5, multiple steps of gradient descent leads to a significant improvement in performances compared to single step of gradient descent.

We assume discount factor to be 1. Assume deterministic state transition function (WLOG).

1.1 NOTATIONS AND SETTINGS

We consider finite horizon reinforcement learning settings with finite state and action spaces. The behavior of an agent is modelled by a policy $\pi(a|s)$, which estimates a probability distribution over a finite set of actions given an observed state. At each time step t, the agent takes an action a_t by sampling from $\pi(a_t|s_t)$. The environment then returns a reward $r_t = r(s_t, a_t)$ and the next state $s_{t+1} = f(s_t, a_t)$, where f is the transition function and it is not revealed to the agent. Given a trajectory, a sequence of states and actions $\rho = (s_1, a_1, \ldots, a_{T-1}, s_T)$, the policy probability and the total reward of ρ are defined as $\pi(\rho) = \prod_{t=1}^{T-1} \pi(a_t|s_t)$ and $r(\rho) = \sum_{t=1}^{T-1} r(s_t, a_t)$. We use $\Delta \triangleq \{\pi | \sum_\rho \pi(\rho) = 1, \pi(\rho) \geq 0, \forall \rho\}$ to refer to the probabilistic simplex over all possible trajectories. For simplicity we also assume that the state transition function is deterministic, while our results can be easily extended to the general setting with random transition functions.

2 EXPLORATION IN POLICY OPTIMIZATION

Policy optimization has been widely used across reinforcement learning (RL) settings. Given a set of parametrized policy functions $\pi_\theta \in \Pi$, policy optimization aims to search the optimal policy π_θ^* that achieves the highest expected reward,

$$\pi_\theta^* \in \arg\max_{\pi_\theta \in \Pi} \mathop{\mathbb{E}}_{\rho \sim \pi_\theta} r(\rho), \tag{1}$$

3 REVERSED ENTROPY POLICY MIRROR DESCENT

Our algorithm is derived from the basic idea of maximizing an entropy-regularized expected reward. Such objective has also been considered in a vast literature including Williams & Peng (1991); Fox et al. (2015); Schulman et al. (2017); Nachum et al. (2017b); Haarnoja et al. (2017), among others. As mentioned in Section 1, we focus on analyzing its learning properties in the *non-convex* setting in the parameter space. In particular, we follow the idea of Mirror Descent to reformulate our objective function into a lift-and-project procedure, and discuss its good properties and potential deficiencies based on such reformulation Section 3.1. We then propose our method to mitigate these deficiencies, called ****. We further develop performance guarantees for our algorithm in the non-convex setting Section 3.2. Our algorithm is developed purely policy-based. We then briefly discussed how our method can cooperate with value function approximation, and present its actor-critic version in Section 3.5.

3.1 REVISITING TRUST REGION POLICY OPTIMIZATION (TRPO)

Recall that TRPO[1] learns the policy by maximizing the expected reward with a relative entropy regularizer. In particular, given a *reference policy* $\bar{\pi}$ (usually the current policy), TRPO learns $\pi_{\theta_{t+1}}$ by

$$\pi_{\theta_{t+1}} = \arg\max_{\pi_\theta \in \Pi} \mathop{\mathbb{E}}_{\rho \sim \pi_\theta} r(\rho) - \tau D_{\mathrm{KL}}(\pi_\theta \| \bar{\pi}). \tag{2}$$

A gradient method with line search is proposed to solve the above optimization problem in Schulman et al. (2015). To better understand its behavior in the parameter space, in this paper we instead reformulate the problem into the following lift-and-project procedure.

$$\begin{aligned} \textbf{(Project Step)} \quad & \arg\min_{\pi_\theta \in \Pi} D_{\mathrm{KL}}(\pi_\theta \| \bar{\pi}_\tau^*), \\ \textbf{(Lift Step)} \quad & \text{where } \bar{\pi}_\tau^* = \arg\max_{\pi \in \Delta} \mathop{\mathbb{E}}_{\rho \sim \pi} r(\rho) - \tau D_{\mathrm{KL}}(\pi \| \bar{\pi}). \end{aligned} \tag{3}$$

[1]Here we present its regularized version, which is equivalent to constraint version that is presented in Schulman et al. (2015).

Proposition 1 shows that the two different formulations are in fact equivalent. Note that although the lift step is non-convex in π, we can actually solve the problem analytically. In fact, by simple calculations, we have

$$\bar{\pi}_\tau^*(\rho) = \frac{\bar{\pi}(\rho)\exp\{r(\rho)/\tau\}}{\sum_{\rho'}\bar{\pi}(\rho')\exp\{r(\rho')/\tau\}}. \tag{4}$$

Proposition 1. *Given a* reference policy $\bar{\pi}$*,*

$$\arg\max_{\pi_\theta\in\Pi} \mathop{\mathbb{E}}_{\rho\sim\pi_\theta} r(\rho) - \tau D_{\mathrm{KL}}(\pi_\theta\|\bar{\pi}) = \arg\min_{\pi_\theta\in\Pi} D_{\mathrm{KL}}(\pi_\theta\|\bar{\pi}_\tau^*).$$

Remark 1. *The lift-and-project procedure in Eq.* (3) *is not completely new in the literature. In fact, it has also been proposed in ******. We postpone the detailed discussion to Section ****.*

Remark 2. *The lift-and-project reformulation suggests an alternative way in solving Eq.* (2)*: Lift the current policy π_{θ_t} to $\bar{\pi}_\tau^*$, then perform multiple steps of gradient descent on the project step to update $\pi_{\theta_{t+1}}$.* [2] *Note that vanilla gradient descent methods for TRPO can be interpreted as performing only one step gradient descent for the project step.*

Ruitong: Is it correct?

One can show that the above lift-and-project procedure asymptotically converges to the optimal policy when Π is a convex set (Nemirovskii et al., 1983; Beck & Teboulle, 2003). However, in practice, the policy π_θ is often parameterized by a complex non-convex function, such as a neural network, which violates the convex constraint set assumption. The next proposition shows that despite of the non-convexity of Π, TRPO still has some desirable properties.

Proposition 2. *Given the projection step $\min_{\pi_\theta\in\Pi} D_{\mathrm{KL}}(\pi_\theta\|\bar{\pi}_\tau^*)$ can be solved optimally, for arbitrary parametrization of π, TRPO satisfies the following properties.*

1. **(Monotonic Improvement Guarantee)** *Assume that π_{θ_t} is the update sequence, then*
$$\mathbb{E}_{\rho\sim\pi_{\theta_{t+1}}} r(\rho) - \mathbb{E}_{\rho\sim\pi_{\theta_t}} r(\rho) \geq 0.$$
2. **(Global optimum inclusion)** *Every stationary point of the expected reward $\mathbb{E}_{\rho\sim\pi_\theta} r(\rho)$, including the globally optimal policy π_{θ^*}, is a fixed point of TRPO.*

Remark 3. *The monotonic improvement guarantee has been derived in (Schulman et al., 2015). We give a simpler and direct proof based on our lift-and-project formulation in Appendix B .*

Despite of its stable and reliable performance, in practice TRPO is observed to get trapped in some poor local optima. Indeed, while the relative entropy regularizer helps in preventing large policy update, it may also limit the exploration of TRPO. Moreover, minimizing the KL divergence $D_{\mathrm{KL}}(\pi_\theta\|\bar{\pi}_\tau^*)$ is known to be *mode seeking* (Kevin, 2012), which can cause mode collapse during the learning process. Once the policy $\bar{\pi}_\tau^*$ drops some of the modes, learning could be trapped into sub-optimal policies. At this point, the relative entropy regularizer will NOT encourage TRPO for further exploration.

Another problem about TRPO is that Proposition 2 relies on the condition that the projection step can be globally optimally solved. However, oftentimes in practice this is not true when π_θ is non-convex in θ, which hinders the applicability of Proposition 2.

3.2 Reversed Entropy Policy Mirror Descent

Ruitong: New name

In this section, we propose two modifications to the lift-and-project procedure to overcome its aforementioned drawbacks. The first modification is an additional entropy regularizer to the lift step, controlled by a separate parameter $\tau' \geq 0$, to encourage the exploration of the algorithm. Second, we employ the reversed *mean seeking* direction of KL divergence $D_{\mathrm{KL}}(\bar{\pi}_{\tau,\tau'}^*\|\pi_\theta)$ to update the policy. The new algorithm, called ****, solves the following optimization problem to update the policy $\pi_{\theta_{t+1}}$:

$$\begin{aligned} &\textbf{(Project Step)} \quad \arg\min_{\pi_\theta\in\Pi} D_{\mathrm{KL}}(\boldsymbol{\bar{\pi}^*_{\tau,\tau'}}\|\boldsymbol{\pi_\theta}), \\ &\textbf{(Lift Step)} \quad \text{where } \bar{\pi}_{\tau,\tau'}^* = \arg\max_{\pi\in\Delta} \mathop{\mathbb{E}}_{\rho\sim\pi} r(\rho) - \tau D_{\mathrm{KL}}(\pi\|\pi_{\theta_t}) + \boldsymbol{\tau'\mathcal{H}(\pi)}. \end{aligned} \tag{5}$$

[2] To estimate this gradient, one would need to use the self-normalized importance sampling Owen (2013). We omit the implementation details since it is not the main algorithm of the paper.

The idea of optimizing the reverse direction of KL divergence has proven to be effective for structured prediction and reinforcement learning in previous work, such as reward augmented maximum likelihood (Norouzi et al., 2016) and UREX (Nachum et al., 2017a). Its *mean seeking* behavior would further encourage the exploration of the algorithm. More importantly, as shown in Proposition 3, reversing the direction of the KL divergence makes the projection step solvable even for the one-layer-softmax neural network π, thus guarantees the desirable properties in practice.

Proposition 3. *Assuem $\pi_\theta(s) = \mathrm{softmax}(\phi_s^\top \theta)$. Given a reference policy $\bar{\pi}$, the projection step $\min_{\theta \in \mathbb{R}^d} D_{\mathrm{KL}}(\bar{\pi} \| \pi_\theta)$ is convex in θ.*

We now prove a similar monotonic improvement guarantee for **** on a surrogate reward $\mathrm{SR}(\pi_\theta)$, as shown in Theorem 1.

Theorem 1. *Assume that π_{θ_t} is the update sequence of the reversed entropy policy mirror descent algorithm, then*

$$SR(\pi_{\theta_{t+1}}) - SR(\pi_{\theta_t}) \geq 0,$$

where

$$SR(\pi_\theta) \triangleq (\tau + \tau') \log \sum_\rho \exp\left\{ \frac{r(\rho) + \tau \log \pi_\theta(\rho)}{\tau + \tau'} \right\}. \tag{6}$$

Therefore, the fixed points of REPMD have a correspondence with the stationary point of $SR(\pi_\theta)$.

Ruitong: Why?

3.3 Behavior of $\mathrm{SR}(\pi)$

Although $\mathrm{SR}(\pi_\theta)$ is different than the expected reward $\mathbb{E}_{\rho \sim \pi_\theta} r(\rho)$, in this section we present some theoretical and empirical evidences that $\mathrm{SR}(\pi_\theta)$ is a reasonable surrogate that may provide good guidance to the learning. In fact, by properly adjusting the two temperature parameters τ and τ', $\mathrm{SR}(\pi_\theta)$ recovers several existing performance measures, as shown in Proposition 4.

Proposition 4. *$SR(\pi_\theta)$ satisfies the following properties:*

(i) $SR(\pi_\theta) \to \max_\rho r(\rho)$, as $\tau \to 0, \tau' \to 0$.

(ii) $SR(\pi_\theta) \to \mathbb{E}_{\rho \sim \pi_\theta} r(\rho)$, as $\tau \to \infty, \tau' \to 0$.

Remark 4. *Note that $SR(\pi_\theta)$ also resembles "softmax value function" that appeared in value based RL (Nachum et al., 2017b; Haarnoja et al., 2018; Ding & Soricut, 2017). The standard soft value can be recovered by $SR(\pi_\theta)$ as a special case when $\tau = 0$ or $\tau' = 0$.*

Ruitong: But Proposition 4 says that $SR(\pi)$ converges to max_ρ when τ and τ' are 0?

According to Proposition 4, one should gradually decrease τ' to reduce the level of exploration as sufficient reward landscape information has been collected during the learning process. Now we can make different choices for τ, depending on the policy constraint set Π.

Given $\tau' \to 0$ and the reward landscape has been sufficiently explored, the constructed unconstrained policy $\bar{\pi}^*_{\tau,\tau'} \to \pi^*$ as $\tau \to 0$, where π^* is the global deterministic optimal policy. Therefore, in the project step π_θ is obtained by directly projecting π^* into Π. When the policy constraint Π has nice properties, such as convexity, that support good behavior of KL projection, π_θ may achieve good performance. However, in practice, Π is typically non-convex. Setting $\tau \to 0$ might not work very well, since directly projecting π^* into Π does not always lead to a π_θ with large expected reward.

Algorithm 1 The REPMD algorithm

Input: τ, τ', K
Output: Policy π_θ

```
Random initialized π_{θ_1};
For t = 1, 2, ..., T do
    Set π̄ = π_{θ_t};
    Repeat
        Sample a mini-batch of K trajectories from π̄;
        Compute the gradient according to Eq. (8);
        Update π_{θ_{t+1}} by the gradient;
    Until converged or reach max_iter;
Return π_{θ_T}.
```

On the other hand, as $\tau \to \infty$ the stationary point set of $\mathrm{SR}(\pi_\theta)$ will approach the stationary point set of $\sum_\rho \pi_\theta(\rho) r(\rho)$. There exists an ideal sequence of τ values and $\tau \to \infty$ that

make π_θ finally converge to $\pi_\theta^* \in \arg\max_{\pi_\theta} \sum_\rho \pi_\theta(\rho) r(\rho)$, i.e., the optimal policy in Π with highest expected reward, recovering the target of policy optimization Eq. (1). The following simulation results suggest that $\mathrm{SR}(\pi)$ could be a good guidance for maximizing the true expected reward. A principled way to find such an ideal sequence of τ is under investigation.

Ruitong: Why?

Ruitong: Add the simulation result for true loss and $\mathrm{SR}(\pi)$ with different value of τ.

3.4 Learning

We now discuss the implementation details of *****. The full algorithm is presented in Algorithm 1. Similarly by simple calculation, we have the analytical solution for the lift step of ****:

$$\bar{\pi}^*_{\tau,\tau'}(\rho) \triangleq \frac{\bar{\pi}(\rho) \exp\left\{\frac{r(\rho) - \tau' \log \bar{\pi}(\rho)}{\tau + \tau'}\right\}}{\sum_{\rho'} \bar{\pi}(\rho') \exp\left\{\frac{r(\rho') - \tau' \log \bar{\pi}(\rho')}{\tau + \tau'}\right\}}. \tag{7}$$

The project step in Eq. (5), $\min_{\pi_\theta \in \Pi} D_{\mathrm{KL}}(\bar{\boldsymbol{\pi}}^*_{\boldsymbol{\tau},\boldsymbol{\tau}'} \| \boldsymbol{\pi}_{\boldsymbol{\theta}})$, can be optimized via stochastic gradient descent, given that one can sample trajectories from $\bar{\pi}^*_{\tau,\tau'}$ to estimate its gradient. To see that, note that

$$\arg\min_{\pi_\theta \in \Pi} D_{\mathrm{KL}}(\bar{\pi}^*_{\tau,\tau'} \| \pi_\theta) = \arg\min_{\pi_\theta \in \Pi} \mathop{\mathbb{E}}_{\rho \sim \bar{\pi}^*_{\tau,\tau'}} - \log \pi_\theta(\rho).$$

The next theorem shows that sampling from $\bar{\pi}^*_{\tau,\tau'}$ can be done using self-normalized importance sampling (Owen, 2013), following the idea of UREX (Nachum et al., 2017a).

Theorem 2. *Let* $\omega_k = \frac{r(\rho_k) - \tau' \log \bar{\pi}(\rho_k)}{\tau + \tau'}$. *Given* K i.i.d. *samples* $\{\rho_1, \ldots, \rho_K\}$ *from the* reference policy $\bar{\pi}$*, we have the following unbiased gradient estimator,*

$$\nabla_\theta D_{\mathrm{KL}}(\bar{\pi}^*_{\tau,\tau'} \| \pi_\theta) \approx -\sum_{k=1}^{K} \frac{\exp\{\omega_k\}}{\sum_{j=1}^{K} \exp\{\omega_j\}} \nabla_\theta \log \pi_\theta(\rho_k), \tag{8}$$

3.5 Cooperate with Value Function Approximation

We consider an infinite trajectory ρ while keeping the discounted factor $\gamma = 1$ for simplicity. All the results have no difficulty to extend to a general value of γ. Given a reference policy $\bar{\pi}$ and an initial state s, recall that the objective in the lift step is

$$\mathcal{O}_{\text{RELENT}}(\pi, s) = \mathop{\mathbb{E}}_{\rho \sim \pi} r(\rho) - \tau D_{\mathrm{KL}}(\pi \| \bar{\pi}) + \tau' \mathcal{H}(\pi),$$

where $\rho = (s_1 = s, a_1, s_2, a_2, \ldots)$. To cooperate with value function approximation, we will need to derive the temporal consistency for this objective. It can be verified that

$$\mathcal{O}_{\text{RELENT}}(\pi, s) = \mathbb{E}_{a \sim \pi(\cdot|s)} \left[r(s,a) + \mathcal{O}_{\text{RELENT}}(\pi, s') + \tau \log \bar{\pi}(a|s) - (\tau + \tau') \log \pi(a|s) \right].$$

Let $\bar{\pi}^*_{\tau,\tau'}(\cdot|s) = \arg\max_\pi \mathcal{O}_{\text{RELENT}}(\pi, s)$ denote the optimal policy on state s. By Eq. (7),

$$\bar{\pi}^*_{\tau,\tau'}(a|s) = \frac{1}{Z} \exp\left\{ \frac{\left[r(s,a) + \mathcal{O}_{\text{RELENT}}(\bar{\pi}^*_{\tau,\tau'}(\cdot|s'), s') \right] + \tau \log \bar{\pi}(a|s)}{\tau + \tau'} \right\}.$$

Further denote the soft optimal state value function $\mathcal{O}_{\text{RELENT}}(\bar{\pi}^*_{\tau,\tau'}(\cdot|s), s)$ by $\bar{V}^*_{\tau,\tau'}(s)$.

The soft optimal state value can be defined by $\bar{V}^*_{\tau,\tau'}(s) = \mathcal{O}_{\text{RELENT}}(\bar{\pi}^*_{\tau,\tau'}, s)$. According to Eq. (7), both $\bar{\pi}^*_{\tau,\tau'}(\cdot|s)$ and $\bar{V}^*_{\tau,\tau'}(s)$ have closed form solution that,

$$\begin{aligned} \bar{V}^*_{\tau,\tau'}(s) &= (\tau + \tau') \log \sum_a \exp\left\{ \frac{\bar{Q}^*_{\tau,\tau'}(s,a) + \tau \log \bar{\pi}(a|s)}{\tau + \tau'} \right\} \\ \bar{\pi}^*_{\tau,\tau'}(a|s) &= \exp\left\{ \frac{\bar{Q}^*_{\tau,\tau'}(s,a) + \tau \log \bar{\pi}(a|s) - \bar{V}^*_{\tau,\tau'}(s)}{\tau + \tau'} \right\} \end{aligned} \tag{9}$$

where $\bar{Q}^*_{\tau,\tau'}(s,a) = r(s,a) + \gamma \bar{V}^*_{\tau,\tau'}(s')$ is the soft-Q function.

3.5.1 LEARNING

We propose to train a soft state value function V_ϕ parameterized by ϕ, a soft Q-function Q_ψ parameterized by ψ, and a policy π_θ parameterized by θ, based on the idea of (5). We now derive the update rules for these parameters.

The soft state value function approximates the soft optimal state value $\bar{V}^*_{\tau,\tau'}$. Note that we can re-express $\bar{V}^*_{\tau,\tau'}$ by

$$\begin{aligned}\bar{V}^*_{\tau,\tau'}(s) =&(\tau+\tau')\log\sum_a \pi(a|s)\exp\left\{\frac{\bar{Q}^*_{\tau,\tau'}(s,a)-\tau'\log\bar{\pi}(a|s)}{\tau+\tau'}\right\}\\ =&(\tau+\tau')\log\mathbb{E}_{a\sim\bar{\pi}}\left[\exp\left\{\frac{\bar{Q}^*_{\tau,\tau'}(s,a)-\tau'\log\bar{\pi}(a|s)}{\tau+\tau'}\right\}\right].\end{aligned}$$

This suggests a Monte-Carlo estimation of $\bar{V}^*_{\tau,\tau'}(s)$: by sampling an action a according to the reference policy $\bar{\pi}$, we have $\bar{V}^*_{\tau,\tau'}(s)\approx\bar{Q}^*_{\tau,\tau'}(s,a)-\tau'\log\bar{\pi}(a|s)$. Then the soft state value function is trained to minimize the mean squared error,

$$\mathcal{O}_{\text{RMAC}}(\phi)=\mathbb{E}_{s\sim\mathcal{D}}\left[\frac{1}{2}\left(V_\phi(s)-\left[Q_\psi(s,a)-\tau'\log\bar{\pi}(a|s)\right]\right)^2\right]$$

where $\mathcal{D}$ is a reply buffer. Furthermore, to increase the stability of the training, we include a target state value network $V_{\bar{\phi}}$, where $\bar{\phi}$ is an exponentially moving average of the value network weights ϕ. One may note that there is no need in principle to include a separate state value function approximation, since it can be computed directly given a soft-Q function and reference policy according to (9). However, including a separate function approximation for the state value can stabilize training as suggested in (Haarnoja et al., 2018). The soft Q-function parameters ψ can be trained to minimize the soft Bellman error using the target state value network,

$$\mathcal{O}_{\text{RMAC}}(\psi)=\mathbb{E}_{(s,a,s')\sim\mathcal{D}}\left[\frac{1}{2}\left(Q_\psi(s,a)-\left[r(s,a)+\gamma V_{\bar{\phi}}(s')\right]\right)^2\right]$$

Our approach also use two soft-Q functions in order to mitigate the overestimation problem caused by value function approximation (Haarnoja et al., 2018; Fujimoto et al., 2018). Specifically, we apply two soft-Q function approximations, $Q_{\psi_1}(s,a)$ and $Q_{\psi_2}(s,a)$, and train them independently. The minimum of the two Q-functions will be used whenever the soft-Q value is needed.

Finally, the policy parameters is updated by doing the project step in (5) with stochastic gradient descent,

$$\mathcal{O}_{\text{RMAC}}(\theta)=\mathbb{E}_{s\sim\mathcal{D}}\left[D_{\text{KL}}\left(\exp\left\{\frac{Q_\psi(s,\cdot)+\tau\log\bar{\pi}(\cdot|s)-V_\phi(s)}{\tau+\tau'}\right\}\middle\|\pi_\theta(\cdot|s)\right)\right]$$

where we approximate $\bar{\pi}^*_{\tau,\tau'}$ with the soft-Q and state value function approximations. The gradient of this objective can be computed by importance sampling,

$$\begin{aligned}\nabla_\theta\mathcal{O}_{\text{RMAC}}(\theta) =&\nabla_\theta\mathbb{E}_{s\sim\mathcal{D}}\left[-\sum_a\exp\left\{\frac{Q_\psi(s,a)+\tau\log\bar{\pi}(a|s)-V_\phi(s)}{\tau+\tau'}\right\}\log\pi_\theta(a|s)\right]\\ =&\nabla_\theta\mathbb{E}_{s\sim\mathcal{D}}\left[\mathbb{E}_{a\sim\bar{\pi}}\left[\exp\left\{\frac{Q_\psi(s,a)-\tau'\log\bar{\pi}(a|s)-V_\phi(s)}{\tau+\tau'}\right\}\log\pi_\theta(a|s)\right]\right]\end{aligned}$$

The complete algorithm is described in Algorithm **??**.

4 RELATED WORKS

The lift-and-project formulation differs our method from most of the previous policy search methods. Two methods that also use such lift-and-project formulation are Mirror Descent Guided Policy Search (MDGPS) and Guide Actor-Critic (GAC) Montgomery & Levine (2016); Tangkaratt et al. (2017). Mirror Descent Guided Policy Search (MDGPS) has a fundamentally different learning scheme from

our method. In particular, MDGPS use the lift step to learn multiple (rather than one) local simple policies, and use the project step to align all the local policies with the global one. Our method also has an additional entropy regularization term in the objective of the lift step to encourage the exploration. Instead, since the lift step in MDGPS is to learn simple policy to fit local trajectories well, such exploration encouragement is naturally not used. GAC has the same objective in the lift step as our method, but uses the mode seeking direction of the KL divergence for the project step, different from the mean seeking direction used in our method (Tangkaratt et al., 2017). Another two concurrent works also use the lift-and-project procedure: Maxmimum aposteriori (MPO) and Soft Actor-Critic (SAC) (Abdolmaleki et al., 2018; Haarnoja et al., 2018). Similar to GAC, SAC use the mode seeking KL divergence for the project step. Our algorithm differs from MPO in the lift step that we have the additional entropy regularizer. As shown in Section 5.4, such entropy term with an annealing τ' could significantly improve the efficiency of the algorithm. Although we may or may not be able to reformulate other "one-step" policy search methods into a lift-and-project procedure following the same idea in Section 3.1, they would still differs from ****, as we use different directions of KL divergence for the lift step and the project step.

In terms of the optimization objective, several existing methods are also similar to our algorithm, by either considering the (relative) entropy regularizer in policy search/optimization, or using KL divergence as the objective to optimize the policy. As mentioned in Section 3.1, our algorithm ensembles the policy gradient descent method in maximizing expected reward with an entropy regularizer (Williams & Peng, 1991; Fox et al., 2015; Nachum et al., 2017b). Using KL divergence, or Bregman divergence, type fo regularization has also been explored in Liu et al. (2015); Thomas et al. (2013); Mahadevan & Liu (2012), but in a different way. In particular, they apply such regularization to the parameters of the linear approximated value functions, while in this paper, the KL divergence is applied to the policy space as a "distance" measure for polices. The literature of relative entropy policy search also uses similar KL divergence regularizer (Peters et al., 2010; Van Hoof et al., 2015) but on the joint distributions of state and action in the purpose of . Instead of the KL divergence, Reward-Weighted Regression (RWR) uses a log of the correlation between π_τ^* and π_θ as its objective, which is then approximated similar to a cross entropy loss (Peters & Schaal, 2007; Wierstra et al., 2008).

TRPO also has a similar formulation to Eq. (2) in a constraint version (Schulman et al., 2015). However, Eq. (2) uses a different direction of the KL divergence. The monotonical improvement guarantee only exists for an impractical formulation of TRPO [3]. Based on our lift-and-project reformulation, we were also able to prove the monotonical improvement guarantee in a fairly simple and direct way. Our method also includes additional modifications to address its potential drawbacks. As shown in 5, such modifications improve its performance significantly. UREX uses the same mean seeking KL divergence for regularization, which encourages exploration but also makes the optimization more difficult. As shown in 5, UREX is significantly less efficient than our method.

Trust PCL method adopts the same objective defined in Eq. (5), which includes both entropy and relative entropy regularizer (Nachum et al., 2017c). However, the policy update strategy is different: while REPMD uses KL projection, Trust PCL inherits the same idea from PCL that minimizes path inconsistency error between value and policy for any trajectories (Nachum et al., 2017b). Although policy optimization by minimizing path inconsistency error can efficiently utilize off-policy data, it loses the desirable monotonical improvement guarantee.

These related works include maximum a posterior policy optimization (Abdolmaleki et al., 2018), and soft actor-critic (Haarnoja et al., 2018).

5 EXPERIMENTS

We present our experimental results in this section. REPMD is compared with standard policy based reinforcement learning algorithms on several different test domains.

[3]For the version that monotonical improvement guarantee holds, TRPO needs to use $D_{\mathrm{KL}}^{\max}$ rather than the stanard KL divergence.

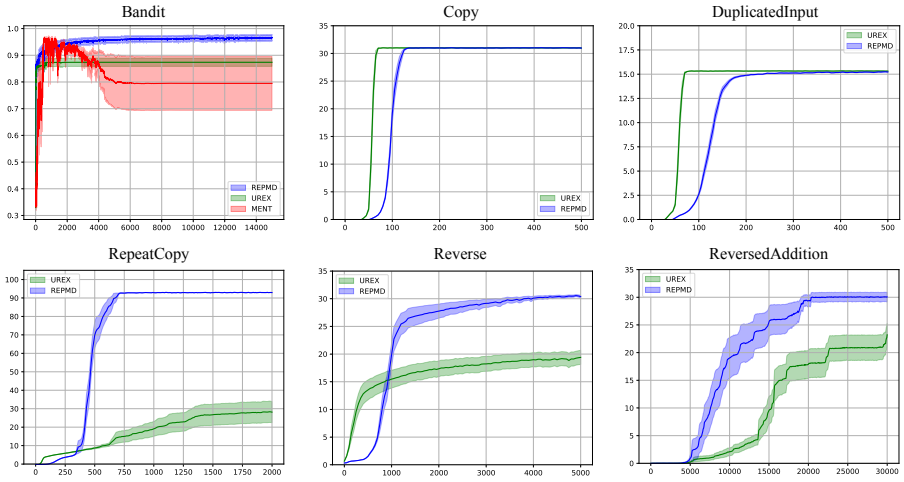

Figure 1: Results using the best hyper-parameters for each method: MENT (red), UREX (green), and REPMD (blue). Plots show average reward with standard error during training. Synthetic bandit results averaged over 5 runs. Algorithmic task results averaged over 25 random training runs (5 runs $\times$ 5 random seeds for neural network initialization). X-axis is number of sampled trajectories.

5.1 TASKS

We test the performance of REPMD on a synthetic bandit problem and the algorithmic tasks from OpenAI gym library (Brockman et al., 2016).

A simple synthetic multi-armed bandit problem is firstly used as an initial testbed. The problem has 10,000 distinct actions. The reward of each action i is initialized by $r_i = s_i^8$, where s_i is randomly sampled from a uniform distribution over $[0, 1)$. We represent each action i with a feature vector $\phi_i \in \mathbb{R}^{20}$, randomly sampled from a standard normal distribution. Let $\Phi = [\phi_1, \ldots, \phi_{10,000}]$ denote the feature matrix. The policy is parameterized by a weight vector $\theta \in \mathbb{R}^{20}$ and defined by $\text{softmax}(\Phi^\top \theta)$. Note that we only learn the θ parameter, the features Φ are fixed during training.

We further test our method on five algorithmic tasks from the OpenAI gym library, in rough order of difficulty: Copy, DuplicatedInput, RepeatCopy, Reverse, and ReversedAddition (Brockman et al., 2016). In each problem, the agent operates on a tape of characters or digits. At each time step, the agent read one character or digit, and then decide to either move the read pointer one step in any direction of the tape, or write a character or digit to output. The total reward of each sampled trajectory is only observed at the end. The goal for each task is:

- **Copy:** Copy a sequence of characters to output.
- **DuplicatedInput:** Duplicate a sequence of characters.
- **RepeatCopy:** Copy a sequence of characters, reverse it, then forward the sequence again.
- **Reverse:** Reverse a sequence of characters.
- **ReversedAddition:** Observe two numbers in base 3 in little-endian order on a $2 \times n$ grid tape. The agent should add the two numbers together.

Lastly, we test our method with value function approximation on the Mujoco library.

Ruitong: Add a description of the Mujoco tasks

5.2 IMPLEMENTATION DETAILS

For all of these algorithmic tasks, the policy is parameterized by a recurrent neural network with LSTM cells of hidden dimension 128 (Hochreiter & Schmidhuber, 1997).

As shown in Algorithm 1, the policy is updated by performing KL divergence projection using stochastic gradient descent (SGD). In our experiments, the end condition of SGD is controlled by two

parameters: $\epsilon > 0$ and F_STEP $\in \{0, 1\}$. First, SGD halts if the change of the KL divergence is below or equal to ϵ. Second, F_STEP decides the maximum number of SGD steps. If F_STEP $= 1$, the maximum number is $\sqrt{t}$ at iteration t; while if F_STEP $= 0$, there is no restriction on the maximum number of gradient steps, and stopping condition of SGD only depends on ϵ.

For the synthetic bandit problem, we explore the following main hyper-parameters: learning rate $\eta \in \{0.1, 0.01, 0.001\}$; entropy regularizer of UREX and MENT $\tau \in \{1.0, 0.5, 0.1, 0.05\}$; relative entropy regularizer of REPMD $\tau \in \{1.0, 0.5, 0.1, 0.05\}$; $\epsilon \in \{0.01, 0.005, 0.001\}$ and F_STEP $\in \{0, 1\}$ for the stop condition of SGD in REPMD. The entropy regularizer τ' of REPMD is set to 0.

For the algorithmic tasks, N distinct environments are used to generate samples. On each environment, K random trajectories are sampled using the agent's policy to estimate gradient according to (8), which gives the batch size $N \times K$ in total. We apply the same batch training setting as in UREX (Nachum et al., 2017a), where $N = 40$ and $K = 10$. The following main hyper-parameters are explored: learning rate $\eta \in \{0.1, 0.01, 0.001\}$; relative entropy regularizer of REPMD $\tau \in \{1.0, 0.5, 0.1, 0.05\}$; entropy regularizer of REPMD $\tau' \in \{0, 0.01, 0.005, 0.001\}$; gradient clipped norm for training LSTM $c \in \{1, 10, 40, 100\}$; $\epsilon \in \{0.01, 0.005, 0.001\}$ and F_STEP $\in \{0, 1\}$ for the stopping condition of SGD in REPMD. Parameters of UREX are set according to the ones reported in Nachum et al. (2017a). Implementations of all algorithm are based on the open source code by the author of UREX [4].

5.3 Comparative Evaluation

For the synthetic bandit problem, we compare REPMD against REINFORCE with entropy regularization (MENT) (Williams, 1992) and under-appreciated reward exploration (UREX) (Nachum et al., 2017a). For the algorithmic tasks, we compare REPMD only against UREX, since UREX has been shown to outperform MENT in these cases (Nachum et al., 2017a). The results are reported in Figure (1). It is clear that REPMD substantially outperforms the competitors on all of these benchmark tasks. REPMD is able to consistently achieve the highest score and learn substantially faster than UREX. We also find the performance of UREX is very unstable. On the difficult tasks, including RepeatCopy, Reverse and ReversedAddition, UREX can only successfully find appropriate solutions a few times out of 5 runs for each random seed, which brings the overall scores down. This observation creates the gap between our presented results with the ones reported in the paper[5]. Note that the performance of REPMD is sill significantly better than UREX even compared with the results reported in Nachum et al. (2017a).

5.4 Ablation Study

Importance of entropy regularizer. The main difference between our objective Eq. (3) with the original MD is to add another entropy regularizer. We demonstrate the importance of this choice by presenting the results of REPMD with $\tau' = 0$.

Importance of KL divergence projection. The main difference between REPMD and the UREX and MENT training methods is to use a projection step to optimize policy rather than performing a single gradient step. To show the importance of the projection step, we reimplement REPMD without projection, which only performs one step of gradient update at each iteration of training.

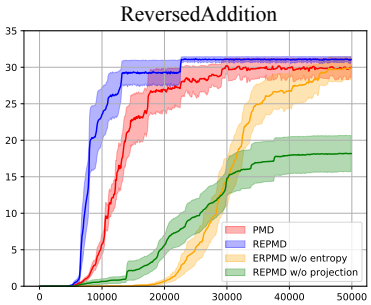

Figure 2: Ablation Study.

Importance of direction of KL divergence. We implemented Policy Mirror Descent (PMD) as another baseline to prove the effectiveness of using the *mean seeking* direction of KL divergence for policy optimization. Like in REPMD, we add a separate tempreture parameter $\tau' \geq 0$ to the original objective function (2) of PMD to encourage further exploration of the policy, which gives $\arg\max_{\pi_\theta \in \Pi} \mathbb{E}_{\rho \sim \pi_\theta} r(\rho) - \tau \mathrm{KL}(\pi_\theta \| \bar{\pi}) + \tau' \mathcal{H}(\pi_\theta)$.

[4] https://github.com/tensorflow/models/tree/master/research/pcl_rl

[5] The results reported in Nachum et al. (2017a) averages over 5 runs of random restart, while our results are averaged over 25 random training runs (5 runs $\times$ 5 random seed for neural network initialization).

Results on ReversedAddition are reported in Figure (2). It clearly shows that optimizing policy by performing the *mean seeking* KL divergence projection is very important as suggested in REPMD.

6 Conclusion and Future Work

In this paper, we have proposed the reversed entropy policy mirror descent (REPMD) method for policy based reinforcement learning, which guarantees monotonic improvement in a well motivated objective. We show that the resulting method achieves better exploration than both a directed exploration method (UREX) and undirected maximum entropy exploration (MENT). It would be interesting to further extend the REPMD method within the actor-critic framework, by developing proper value function learning approach.

References

Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Remi Munos, Nicolas Heess, and Martin Riedmiller. Maximum a posteriori policy optimisation. In *ICLR*, 2018.

Amir Beck and Marc Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3):167–175, 2003.

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

Nan Ding and Radu Soricut. Cold-start reinforcement learning with softmax policy gradient. In *Advances in Neural Information Processing Systems*, pp. 2817–2826, 2017.

Roy Fox, Ari Pakman, and Naftali Tishby. Taming the noise in reinforcement learning via soft updates. *arXiv preprint arXiv:1512.08562*, 2015.

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

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. *arXiv preprint arXiv:1702.08165*, 2017.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

P Murphy Kevin. *Machine learning: a probabilistic perspective*. Cambridge, MA, 2012.

Bo Liu, Ji Liu, Mohammad Ghavamzadeh, Sridhar Mahadevan, and Marek Petrik. Finite-sample analysis of proximal gradient td algorithms. In *UAI*, pp. 504–513. Citeseer, 2015.

Sridhar Mahadevan and Bo Liu. Sparse q-learning with mirror descent. *arXiv preprint arXiv:1210.4893*, 2012.

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.

William H Montgomery and Sergey Levine. Guided policy search via approximate mirror descent. In *Advances in Neural Information Processing Systems*, pp. 4008–4016, 2016.

Ofir Nachum, Mohammad Norouzi, and Dale Schuurmans. Improving policy gradient by exploring under-appreciated rewards. In *ICLR*, 2017a.

Ofir Nachum, Mohammad Norouzi, Kelvin Xu, and Dale Schuurmans. Bridging the gap between value and policy based reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 2772–2782, 2017b.

Ofir Nachum, Mohammad Norouzi, Kelvin Xu, and Dale Schuurmans. Trust-pcl: An off-policy trust region method for continuous control. In *ICLR*, 2017c.

Arkadii Nemirovskii, David Borisovich Yudin, and Edgar Ronald Dawson. Problem complexity and method efficiency in optimization. 1983.

Mohammad Norouzi, Samy Bengio, Navdeep Jaitly, Mike Schuster, Yonghui Wu, Dale Schuurmans, et al. Reward augmented maximum likelihood for neural structured prediction. In *Advances In Neural Information Processing Systems*, pp. 1723–1731, 2016.

Art B. Owen. *Monte Carlo theory, methods and examples*. 2013.

Jan Peters and Stefan Schaal. Reinforcement learning by reward-weighted regression for operational space control. In *Proceedings of the 24th international conference on Machine learning*, pp. 745–750. ACM, 2007.

Jan Peters, Katharina Mülling, and Yasemin Altun. Relative entropy policy search. In *AAAI*, pp. 1607–1612. Atlanta, 2010.

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

John Schulman, Xi Chen, and Pieter Abbeel. Equivalence between policy gradients and soft q-learning. *arXiv preprint arXiv:1704.06440*, 2017.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Voot Tangkaratt, Abbas Abdolmaleki, and Masashi Sugiyama. Guide actor-critic for continuous control. *arXiv preprint arXiv:1705.07606*, 2017.

Philip S Thomas, William C Dabney, Stephen Giguere, and Sridhar Mahadevan. Projected natural actor-critic. In *Advances in neural information processing systems*, pp. 2337–2345, 2013.

Herke Van Hoof, Jan Peters, and Gerhard Neumann. Learning of non-parametric control policies with high-dimensional state features. In *Artificial Intelligence and Statistics*, pp. 995–1003, 2015.

Daan Wierstra, Tom Schaul, Jan Peters, and Juergen Schmidhuber. Episodic reinforcement learning by logistic reward-weighted regression. In *International Conference on Artificial Neural Networks*, pp. 407–416. Springer, 2008.

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

Ronald J Williams and Jing Peng. Function optimization using connectionist reinforcement learning algorithms. *Connection Science*, 3(3):241–268, 1991.

A PROOF OF PROPOSITION 1

Proof. Note that $-\tau D_{\mathrm{KL}}(\pi_\theta \| \bar{\pi}_\tau^*) = -\tau \sum_\rho \pi_\theta(\rho) \log \pi_\theta(\rho) + \tau \sum_\rho \pi_\theta(\rho)(\log \bar{\pi}(\rho) + r(\rho)/\tau) - Z_{\bar{\pi}} = \mathbb{E}_{\rho\sim\pi_\theta} r(\rho) - \tau D_{\mathrm{KL}}(\pi_\theta \| \bar{\pi}) - Z_{\bar{\pi}}$. Note the fact that $Z_{\bar{\pi}} \triangleq \tau \log \sum_\rho \bar{\pi}(\rho) \exp\{r(\rho)/\tau\}$ is indenpendent of π_θ given the reference policy $\bar{\pi}$. □

B PROOF OF PROPOSITION 2

Proof. **(Monotonic Improvement Guarantee)** By the definition of $\pi_{\theta_{t+1}}$, note that $D_{\mathrm{KL}}(\pi_{\theta_{t+1}} \| \bar{\pi}_\tau^*) = \min_{\pi_\theta \in \Pi} D_{\mathrm{KL}}(\pi_\theta \| \bar{\pi}_\tau^*) \leq D_{\mathrm{KL}}(\pi_{\theta_t} \| \bar{\pi}_\tau^*)$. By expanding the KL divergence and rearranging terms, we have $\tau D_{\mathrm{KL}}(\pi_{\theta_{t+1}} \| \pi_{\theta_t}) - \sum_\rho \pi_{\theta_{t+1}}(\rho) r(\rho) \leq -\sum_\rho \pi_{\theta_t}(\rho) r(\rho)$, which gives $\mathbb{E}_{\rho\sim\pi_{\theta_{t+1}}} r(\rho) - \mathbb{E}_{\rho\sim\pi_{\theta_t}} r(\rho) \geq \tau D_{\mathrm{KL}}(\pi_{\theta_{t+1}} \| \pi_{\theta_t}) \geq 0$.

(Global optimum inclusion) □

C PROOF OF THEOREM 1

Proof. Using $D_{\mathrm{KL}}(\bar{\pi}_{\tau,\tau'}^* \| \pi_{\theta_{t+1}}) = \min_{\pi_\theta \in \Pi} D_{\mathrm{KL}}(\bar{\pi}_{\tau,\tau'}^* \| \pi_\theta) \leq D_{\mathrm{KL}}(\bar{\pi}_{\tau,\tau'}^* \| \pi_{\theta_t})$ and Jensen's inequality,

$$
\begin{aligned}
&\mathrm{SR}(\pi_{\theta_{t+1}}) - \mathrm{SR}(\pi_{\theta_t}) = (\tau+\tau') \log \sum_\rho \frac{\exp\left\{\frac{r(\rho)+\tau \log \pi_{\theta_{t+1}}(\rho)}{\tau+\tau'}\right\}}{\sum_\rho \exp\left\{\frac{r(\rho)+\tau \log \pi_{\theta_t}(\rho)}{\tau+\tau'}\right\}} \\
=&(\tau+\tau') \log \sum_\rho \frac{\exp\left\{\frac{r(\rho)+\tau \log \pi_{\theta_t}(\rho)}{\tau+\tau'}\right\}}{\sum_\rho \exp\left\{\frac{r(\rho)+\tau \log \pi_{\theta_t}(\rho)}{\tau+\tau'}\right\}} \cdot \exp\left\{\frac{\tau \log \pi_{\theta_{t+1}}(\rho) - \tau \log \pi_{\theta_t}(\rho)}{\tau+\tau'}\right\} \\
=&(\tau+\tau') \log \sum_\rho \bar{\pi}_{\tau,\tau'}^*(\rho) \cdot \exp\left\{\frac{\tau \log \pi_{\theta_{t+1}}(\rho) - \tau \log \pi_{\theta_t}(\rho)}{\tau+\tau'}\right\} \\
\geq&(\tau+\tau') \sum_\rho \bar{\pi}_{\tau,\tau'}^*(\rho) \log \exp\left\{\frac{\tau \log \pi_{\theta_{t+1}}(\rho) - \tau \log \pi_{\theta_t}(\rho)}{\tau+\tau'}\right\} \\
=&\tau \sum_\rho \bar{\pi}_{\tau,\tau'}^*(\rho) \log \frac{\pi_{\theta_{t+1}}(\rho)}{\pi_{\theta_t}(\rho)} = \tau \left[D_{\mathrm{KL}}(\bar{\pi}_{\tau,\tau'}^* \| \pi_{\theta_t}) - D_{\mathrm{KL}}(\bar{\pi}_{\tau,\tau'}^* \| \pi_{\theta_{t+1}})\right] \geq 0.
\end{aligned}
$$

□

D PROOF OF PROPOSITION 3

Proof. Note that $\pi_\theta = \frac{\exp\{\Phi^\top \theta\}}{\mathbf{1}^\top \exp\{\Phi^\top \theta\}}$, where Φ is the feature matrix and θ is the policy parameter. Simply compute the Hessian matrix of the objective,

$$
\frac{d^2 \mathrm{KL}(\bar{\pi} \| \pi_\theta)}{d\theta^2} = \Phi^\top (\Delta(\pi_\theta) - \pi_\theta \pi_\theta^\top) \Phi \succeq 0.
$$

Thus $D_{\mathrm{KL}}(q \| \pi_\theta)$ is convex in θ. □

E PROOF OF PROPOSITION 4

Proof. To prove (i), note that as $\tau \to 0$, $\mathrm{SR}(\pi_\theta) \to \tau' \log \sum_\rho \exp\left\{\frac{r(\rho)}{\tau'}\right\}$, the standard softmax value. Taking limit on τ' gives the hardmax value $\max_\rho r(\rho)$ as $\tau' \to 0$.

To prove (ii), we have

$$\lim_{\tau\to\infty}(\tau+\tau')\log\sum_{\rho}\exp\left\{\frac{r(\rho)+\tau\log\pi_\theta(\rho)}{\tau+\tau'}\right\}=\lim_{\tau\to\infty}\frac{\sum_\rho \pi_\theta(\rho)\exp\{\frac{r(\rho)-\tau'\log\pi_\theta(\rho)}{\tau+\tau'}\}\big(r(\rho)-\tau'\log\pi_\theta(\rho)\big)}{\sum_\rho \pi_\theta(\rho)\exp\{\frac{r(\rho)-\tau'\log\pi_\theta(\rho)}{\tau+\tau'}\}}$$

$$=\sum_\rho \pi_\theta(\rho)\left[r(\rho)-\tau'\log\pi_\theta(\rho)\right]=\mathop{\mathbb{E}}_{\rho\sim\pi_\theta} r(\rho)+\tau'\mathcal{H}(\pi_\theta)$$

As $\tau'\to 0$, $\mathrm{SR}(\pi_\theta)\to\mathbb{E}_{\rho\sim\pi_\theta} r(\rho)$. □

F Proof for Section ***

Lemma 1. *The lift step of Eq.* (5) *has the following closed form expression:*

$$\bar{\pi}^*_{\tau,\tau'}(\rho)\triangleq\frac{\bar{\pi}(\rho)\exp\left\{\frac{r(\rho)-\tau'\log\bar{\pi}(\rho)}{\tau+\tau'}\right\}}{\sum_{\rho'}\bar{\pi}(\rho')\exp\left\{\frac{r(\rho')-\tau'\log\bar{\pi}(\rho')}{\tau+\tau'}\right\}}.$$

Proof. Rewrite the objective function defined in Eq. (5),

$$\mathop{\mathbb{E}}_{\rho\sim\pi} r(\rho)-\tau D_{\mathrm{KL}}(\pi\|\bar{\pi})+\tau'\mathcal{H}(\pi)=\mathop{\mathbb{E}}_{\rho\sim\pi}[r(\rho)+\tau\log\bar{\pi}(\rho)]+(\tau+\tau')\mathcal{H}(\pi), \tag{10}$$

which is an entropy regularized reshaped reward objective. The optimal policy of this objective can be obtained by directly applying Lemma 4 of Nachum et al. (2017b), i.e.

$$\bar{\pi}^*_{\tau,\tau'}(\rho)\propto\exp\left\{\frac{r(\rho)+\tau\log\bar{\pi}(\rho)}{\tau+\tau'}\right\}=\bar{\pi}(\rho)\exp\left\{\frac{r(\rho)-\tau'\log\bar{\pi}(\rho)}{\tau+\tau'}\right\}. \tag{11}$$

□

G Proof of Theorem 2

Proof. Note that

$$D_{\mathrm{KL}}(\bar{\pi}^*_{\tau,\tau'}\|\pi_\theta)=\mathop{\mathbb{E}}_{\rho\sim\bar{\pi}^*_{\tau,\tau'}}\left[\log\bar{\pi}^*_{\tau,\tau'}(\rho)-\log\pi_\theta(\rho)\right]=\mathop{\mathbb{E}}_{\rho\sim\bar{\pi}}\left[\frac{\bar{\pi}^*_{\tau,\tau'}(\rho)}{\bar{\pi}(\rho)}\left(\log\bar{\pi}^*_{\tau,\tau'}(\rho)-\log\pi_\theta(\rho)\right)\right].$$

Therefore, taking gradient on both sides,

$$\nabla_\theta D_{\mathrm{KL}}(\bar{\pi}^*_{\tau,\tau'}\|\pi_\theta)\approx-\frac{1}{K}\sum_{k=1}^K\frac{\bar{\pi}^*_{\tau,\tau'}(\rho_k)}{\bar{\pi}(\rho_k)}\nabla_\theta\log\pi_\theta(\rho_k)$$

$$\approx-\frac{1}{K}\sum_{k=1}^K\frac{\exp\{\omega_k\}}{\frac{1}{K}\sum_{j=1}^K\exp\{\omega_j\}}\nabla_\theta\log\pi_\theta(\rho_k)=-\sum_{k=1}^K\frac{\exp\{\omega_k\}}{\sum_{j=1}^K\exp\{\omega_j\}}\nabla_\theta\log\pi_\theta(\rho_k).$$ □

Ruitong: More details

H Stochastic Transition Setting

In **??**, we assume that the state transition function is deterministic for simplicity. For completeness, we consider the general stochastic transition setting here.

H.1 Notations and Settings

Recall in **??**, the policy probability of trajectory $\rho=(s_1,a_1,\ldots,a_{T-1},s_T)$ is denoted as $\pi(\rho)=\prod_{t=1}^{T-1}\pi(a_t|s_t)$. We define transition probability of ρ as $f(\rho)\triangleq\prod_{t=1}^{T-1}f(s_{s+1}|s_t,a_t)$. The total probability of ρ under policy π and transition f is then $p_{\pi,f}(\rho)\triangleq\pi(\rho)f(\rho)=\prod_{t=1}^{T-1}\pi(a_t|s_t)f(s_{s+1}|s_t,a_t)$. We use $\Delta_f\triangleq\{\pi|\sum_\rho p_{\pi,f}(\rho)=\sum_\rho\pi(\rho)f(\rho)=1,\pi(\rho)\geq 0,f(\rho)>0,\forall\rho\}$ to refer to the probabilistic simplex over all possible trajectories. It is obvious that $p_{\pi,f}(\rho)=\pi(\rho)$ and $\Delta_f=\Delta$ under deterministic transition setting, i.e., $f(\rho)=1,\forall\rho$.

H.2 REPMD OPTIMIZATION PROBLEM

The proposed REPMD algorithm solves Eq. (5) in the deterministic transition setting. In the stochastic setting, the corresponding problem is,

$$\begin{aligned} &\arg\min_{\pi_\theta \in \Pi} D_{\mathrm{KL}}(p_{\bar{\pi}^*_{\tau,\tau'},f} \| p_{\pi_\theta,f}), \\ \text{where } \bar{\pi}^*_{\tau,\tau'} &= \arg\max_{\pi \in \Delta_f} \mathop{\mathbb{E}}_{\rho \sim p_{\pi,f}} [r(\rho) - \tau' \log \pi(\rho)] - \tau D_{\mathrm{KL}}(p_{\pi,f} \| p_{\pi_{\theta_t},f}), \end{aligned} \tag{12}$$

which also recovers Eq. (5) as a special case when $f(\rho) = 1, \forall \rho$.

Like Eq. (5), $\bar{\pi}^*_{\tau,\tau'}$ in Eq. (12) also has a closed form expression,

Lemma 2. *The unconstrained optimal policy of Eq.* (12) *has the following closed form expression:*

$$\bar{\pi}^*_{\tau,\tau'}(\rho) \triangleq \frac{\bar{\pi}(\rho) \exp\left\{\frac{r(\rho) - \tau' \log \bar{\pi}(\rho)}{\tau + \tau'}\right\}}{\sum_{\rho'} \bar{\pi}(\rho') f(\rho') \exp\left\{\frac{r(\rho') - \tau' \log \bar{\pi}(\rho')}{\tau + \tau'}\right\}}.$$

Proof. Rewrite the maximization problem in Eq. (12) as (take π_{θ_t} as the reference policy $\bar{\pi}$),

$$\begin{aligned} &\underset{\pi}{\text{maximize}} \sum_\rho \pi(\rho) f(\rho) \left[r(\rho) - (\tau + \tau') \log \pi(\rho) + \tau \log \bar{\pi}(\rho)\right] \\ &\text{subject to} \sum_\rho \pi(\rho) f(\rho) = 1. \end{aligned}$$

The KKT condition of the above problem is,

$$\begin{aligned} f(\rho) \left[r(\rho) - (\tau + \tau') \log \pi(\rho) + \tau \log \bar{\pi}(\rho) + \lambda - (\tau + \tau')\right] = 0, \ \forall \rho \\ \sum_\rho \pi(\rho) f(\rho) = 1. \end{aligned}$$

Using $f(\rho) > 0, \forall \rho$ and solving the KKT condition, we obtain the expression of $\bar{\pi}^*_{\tau,\tau'}$. □

Lemma 2 recovers Lemma 1 as a special case when $f(\rho) = 1, \forall \rho$.

H.3 THEORETICAL ANALYSIS

In stochastic transition setting, we define the follow softmax approximated expected reward of π_θ

$$\mathrm{SR}_f(\pi_\theta) \triangleq (\tau + \tau') \log \sum_\rho f(\rho) \exp\left\{\frac{r(\rho) + \tau \log \pi_\theta(\rho)}{\tau + \tau'}\right\},$$

which recovers $\mathrm{SR}(\pi_\theta)$ when $f(\rho) = 1, \forall \rho$. The monotonic improvement property is for $\mathrm{SR}_f(\pi_\theta)$.

Theorem 3. *Assume that π_{θ_t} is the update sequence of the REPMD algorithm in Eq.* (12)*, then*

$$\mathit{SR}_f(\pi_{\theta_{t+1}}) - \mathit{SR}_f(\pi_{\theta_t}) \geq 0.$$

Proof. Using $D_{\mathrm{KL}}(p_{\bar{\pi}^*_{\tau,\tau'},f}\|p_{\pi_{\theta_{t+1}},f}) = \min_{\pi_\theta\in\Pi} D_{\mathrm{KL}}(p_{\bar{\pi}^*_{\tau,\tau'},f}\|p_{\pi_\theta,f}) \le D_{\mathrm{KL}}(p_{\bar{\pi}^*_{\tau,\tau'},f}\|p_{\pi_{\theta_t},f})$ and Jensen's inequality,

$$
\begin{aligned}
&\mathrm{SR}_f(\pi_{\theta_{t+1}}) - \mathrm{SR}_f(\pi_{\theta_t}) = (\tau+\tau')\log\sum_\rho \frac{f(\rho)\exp\left\{\frac{r(\rho)+\tau\log\pi_{\theta_{t+1}}(\rho)}{\tau+\tau'}\right\}}{\sum_\rho f(\rho)\exp\left\{\frac{r(\rho)+\tau\log\pi_{\theta_t}(\rho)}{\tau+\tau'}\right\}} \\
=&(\tau+\tau')\log\sum_\rho \frac{f(\rho)\exp\left\{\frac{r(\rho)+\tau\log\pi_{\theta_t}(\rho)}{\tau+\tau'}\right\}}{\sum_\rho f(\rho)\exp\left\{\frac{r(\rho)+\tau\log\pi_{\theta_t}(\rho)}{\tau+\tau'}\right\}}\cdot\exp\left\{\frac{\tau\log\pi_{\theta_{t+1}}(\rho)-\tau\log\pi_{\theta_t}(\rho)}{\tau+\tau'}\right\} \\
=&(\tau+\tau')\log\sum_\rho \bar{\pi}^*_{\tau,\tau'}(\rho)f(\rho)\cdot\exp\left\{\frac{\tau\log\pi_{\theta_{t+1}}(\rho)-\tau\log\pi_{\theta_t}(\rho)}{\tau+\tau'}\right\} \\
\ge&(\tau+\tau')\sum_\rho \bar{\pi}^*_{\tau,\tau'}(\rho)f(\rho)\cdot\log\exp\left\{\frac{\tau\log\pi_{\theta_{t+1}}(\rho)-\tau\log\pi_{\theta_t}(\rho)}{\tau+\tau'}\right\} \\
=&\tau\sum_\rho \bar{\pi}^*_{\tau,\tau'}(\rho)f(\rho)\cdot\log\frac{\pi_{\theta_{t+1}}(\rho)}{\pi_{\theta_t}(\rho)} \\
=&\tau\sum_\rho \bar{\pi}^*_{\tau,\tau'}(\rho)f(\rho)\cdot\log\frac{\pi_{\theta_{t+1}}(\rho)f(\rho)}{\pi_{\theta_t}(\rho)f(\rho)} \\
=&\tau\left[D_{\mathrm{KL}}(p_{\bar{\pi}^*_{\tau,\tau'},f}\|p_{\pi_{\theta_t},f}) - D_{\mathrm{KL}}(p_{\bar{\pi}^*_{\tau,\tau'},f}\|p_{\pi_{\theta_{t+1}},f})\right] \ge 0.
\end{aligned}
$$

□

$\mathrm{SR}_f(\pi_\theta)$ also recovers corresponding performance measures in the stochastic transition setting.

Proposition 5. *$\mathrm{SR}_f(\pi_\theta)$ satisfies the following properties:*

(i) $\mathrm{SR}_f(\pi_\theta) \to \max_\rho r(\rho)$, *as* $\tau\to 0, \tau'\to 0$.

(ii) $\mathrm{SR}_f(\pi_\theta) \to \mathop{\mathbb{E}}_{\rho\sim p_{\pi_\theta,f}} r(\rho)$, *as* $\tau\to\infty, \tau'\to 0$.

Proof. To prove (i), note that as $\tau\to 0$, $\mathrm{SR}_f(\pi_\theta)\to\tau'\log\sum_\rho f(\rho)\exp\left\{\frac{r(\rho)}{\tau'}\right\}$. Taking limit on τ' gives the hardmax value $\max_\rho r(\rho)$ as $\tau'\to 0$.

To prove (ii), we have

$$
\begin{aligned}
&\lim_{\tau\to\infty}(\tau+\tau')\log\sum_\rho f(\rho)\exp\left\{\frac{r(\rho)+\tau\log\pi_\theta(\rho)}{\tau+\tau'}\right\} \\
=&\lim_{\tau\to\infty}\frac{\sum_\rho \pi_\theta(\rho)f(\rho)\exp\{\frac{r(\rho)-\tau'\log\pi_\theta(\rho)}{\tau+\tau'}\}\left(r(\rho)-\tau'\log\pi_\theta(\rho)\right)}{\sum_\rho \pi_\theta(\rho)f(\rho)\exp\{\frac{r(\rho)-\tau'\log\pi_\theta(\rho)}{\tau+\tau'}\}} \\
=&\sum_\rho \pi_\theta(\rho)f(\rho)\left[r(\rho)-\tau'\log\pi_\theta(\rho)\right] = \mathop{\mathbb{E}}_{\rho\sim p_{\pi_\theta,f}} r(\rho) - \tau'\cdot\mathop{\mathbb{E}}_{\rho\sim p_{\pi_\theta,f}}\log\pi_\theta(\rho)
\end{aligned}
$$

As $\tau'\to 0$, $\mathrm{SR}_f(\pi_\theta)\to\mathbb{E}_{\rho\sim p_{\pi_\theta,f}} r(\rho)$. □

H.4 Learning

The REPMD learning process is intact under the stochastic transition setting. Similar with Appendix G, we can estimate the KL divergence in the projection step of Eq. (12) by drawing K *i.i.d.* samples $\{\rho_1,\ldots,\rho_K\}$ from $p_{\bar{\pi},f}$, i.e., the mixture of $\bar{\pi}$ and f, which is exactly the process of sampling from $\bar{\pi}$ and interacting with the environment,

$$
\begin{aligned}
D_{\mathrm{KL}}(p_{\bar{\pi}^*_{\tau,\tau'},f}\|p_{\pi_\theta,f}) &= \mathop{\mathbb{E}}_{\rho\sim p_{\bar{\pi}^*_{\tau,\tau'},f}}\left[\log\bar{\pi}^*_{\tau,\tau'}(\rho)-\log\pi_\theta(\rho)\right] \\
&= \mathop{\mathbb{E}}_{\rho\sim p_{\bar{\pi},f}}\frac{\bar{\pi}^*_{\tau,\tau'}(\rho)}{\bar{\pi}(\rho)}\left[\log\bar{\pi}^*_{\tau,\tau'}(\rho)-\log\pi_\theta(\rho)\right].
\end{aligned}
\tag{13}
$$

We can then approximate the gradient of $D_{\mathrm{KL}}(p_{\bar{\pi}^*_{\tau,\tau'},f} \| p_{\pi_\theta,f})$ by averaging these K samples according to Eq. (13).

Theorem 4. *Let* $\omega_k = \frac{r(\rho_k) - \tau' \log \bar{\pi}(\rho_k)}{\tau + \tau'}$. *Given* K i.i.d. *samples* $\{\rho_1, \ldots, \rho_K\}$ *from the* reference policy $\bar{\pi}$*, we have the following unbiased gradient estimator,*

$$\nabla_\theta D_{\mathrm{KL}}(p_{\bar{\pi}^*_{\tau,\tau'},f} \| p_{\pi_\theta,f}) \approx -\sum_{k=1}^{K} \frac{\exp\{\omega_k\}}{\sum_{j=1}^{K} \exp\{\omega_j\}} \nabla_\theta \log \pi_\theta(\rho_k), \tag{14}$$

Proof. See Theorem 2. □