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 proposes 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 its successes in solving a wide range of difficult sequential decision making problems (Schulman et al., 2015; Mnih et al., 2015; Silver et al., 2016). Among various deep RL methods one of the core ideas is the policy optimization, which presents the policy search problem as an optimization problem, e.g. the so-called policy gradient methods (Williams & Peng, 1991; Williams, 1992; Sutton et al., 1998). Different from a traditional optimization problem, policy optimization needs to collect its training data from the environment based on its policy, the argument also to be optimized on. Such interaction may make the performance of the algorithm very sensitive to the policy with high variance, which results in the need of an extensive hyperparameter tuning. It may also create many local optima, where a lack-of-exploration policy would fail to collect high-reward training data which in turns prevents any further improvement of the policy from useful signals. Indeed, training a RL model usually requires an extensive hyperparameter tuning and a huge, if not impractical, number of sampled trajectories. It has been a popular topic how to design an algorithm that has the ability to improve the policy steadily, and explore the policy space properly and efficiently, so that it can avoid getting trapped in a poor local optimum and discover a good policy quickly.

Many improvements to policy optimization have been proposed in the literature, to both the stability and the data efficiency of the algorithm (Peters et al., 2010; Van Hoof et al., 2015; Fox et al., 2015; Schulman et al., 2015; Montgomery & Levine, 2016; Nachum et al., 2017b;c; Tangkaratt et al., 2017; Abdolmaleki et al., 2018; Haarnoja et al., 2018). However, a clear gap between their great success in practice and their theoretical analyses remains. In particular, existing theoretical results typically assume a fairly simple setting, which either ignores the parametrization of the model or only considers linear models. However, it is hard to justify that such assumptions would still hold when using complex approximators with finite representation capacity (restricted by the parameter space constraint), e.g. neural network models. Such discrepancy between theory and practice has become a hurdle to the wider application of model-free policy gradient methods.

In this paper, we focus on a more realistic setting where the policy is parametrized as a *non-convex* function in its parameter space. We start with the entropy-regularized expected reward as our objective (Williams & Peng, 1991; Fox et al., 2015; Schulman et al., 2017; Nachum et al., 2017b; Haarnoja et al., 2017). We first reformulate this objective as a lift-and-project procedure following the idea of Mirror Descent. On the one hand, such reformulation makes it easier to analyze the algorithm in the parameter space. Based on such reformulation, a monotonic improvement guarantee in performance can be proved with a fairly simple proof, even in the non-convex setting. We also studies its fixed

points properties. On the other hand, such formulation suggests to perform multiple steps of gradient descent on the project step, which leads to our first algorithm, Policy Mirror Descent (PMD). PMD first lifts the policy in the entire policy-simplex, ignoring the constraint induced by its parametrization, then solving a project step by multiple steps of gradient descent to update the policy in the parameter space.

We then propose additional modifications to mitigate the potential deficiencies of PMD. Our main algorithm, Reversed Entropy PMD (REPMD), takes both the entropy and the relative entropy regularizers, and uses a mean seeking direction of KL divergence for projection. The benefit of this is twofold: (1) Using the mean seeking KL divergence in the project step, as well as the additional entropy regularizer, helps avoids some poor local optima; (2) The problem can now be globally optimally solved in some specific *non-convex* cases, e.g. one-layer-softmax networks. Similar guarantees can also be proved for REPMD but only on a surrogate reward $\mathrm{SR}(\pi)$ rather than the expected reward. We further study the properties of $\mathrm{SR}(\pi)$ and provide theoretical and empirical evidences that SR could serve as a good guidance for learning the policy. Lastly, we also show how our algorithm can be extended to cooperate with value function approximation, and present its actor-critic version.

1.1 Notations and Problem Setting

For simplicity, we only 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 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 $r(\rho) = \sum_{t=1}^{T-1} r(s_t, a_t)$. Given a set of parametrized policy functions $\pi_\theta \in \Pi$, policy optimization aims to search the optimal policy π_θ^* by maximizing the expected reward,

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

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. Without loss of generality, we also assume that the state transition function is deterministic, and the discount factor $\gamma = 1$. The same simplification is also assumed in Nachum et al. (2017a). Results for stochastic state transition function are presented in Appendix B.

2 Reversed Entropy Policy Mirror Descent

In this section, we first present our algorithm PMD. As mentioned in Section 1, we focus on analyzing its properties in the *non-convex* setting. Two additional modifications are then proposed in Section 2.2, leading to our main algorithm REPMD. Lastly, although our algorithm is developed purely policy-based, it can be easily extended to cooperate with value function approximation. We briefly discussed the actor-critic version of our method in Section 2.4.

2.1 Policy Mirror Descent

Given a *reference policy* $\bar{\pi}$ (usually the current policy), an algorithm that maximizes the relative entropy regularized expected reward typically learns π_θ by

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

Besides reinforcement learning, such regularizer has been widely used in the literature of online learning and constrained optimization problems (Nemirovskii et al., 1983; Beck & Teboulle, 2003), mostly in the mirror descent algorithm. Note that when $\bar{\pi}$ is a uniform distribution, Eq. (2) also ensembles the entropy regularizer.

To better analyze its behavior in the parameter space, in this paper we follow the idea of mirror descent to reformulate the problem as a lift-and-project procedure.

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

Proposition 1 shows that the problems of Eq. (3) and Eq. (2) are in fact equivalent, even when π is non-convex in θ. Note that although it is non-convex in π for the lift step, we can actually solve the problem analytically, where

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

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

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

The lift-and-project reformulation suggests an alternative algorithm: Lift the current policy π_{θ_t} to $\bar{\pi}_\tau^*$, then perform multiple steps of gradient descent on the project step to update $\pi_{\theta_{t+1}}$. [1] We name this method by Policy Mirror Descent (PMD). One can show that PMD asymptotically converges to the optimal policy when Π is a convex set (Nemirovskii et al., 1983; Beck & Teboulle, 2003). The next proposition shows that despite of the non-convexity of Π, PMD still has some desirable properties.

Ruitong: May wanna add the implementation details in Appendix.

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

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 PMD.*

Despite of the desirable properties, Proposition 2 relies on the condition that the projection step of PMD needs to be globally optimally solved. Oftentimes in practice this is not true when π_θ is non-convex in θ, which hinders the applicability of Proposition 2.

Another problem is that PMD is observed to get trapped in some poor local optima in practice. 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 PMD for further exploration.

2.2 Reversed Entropy Policy Mirror Descent

In this section, we propose two modifications to PMD to overcome its aforementioned potential deficiencies. 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, $D_{\mathrm{KL}}(\bar{\pi}_{\tau,\tau'}^* \| \pi_\theta)$, for the project step. The new algorithm, called Reversed Entropy Policy Mirror Descent (REPMD), solves the following optimization problem to update the policy $\pi_{\theta_{t+1}}$:

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

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

As we will see in Section 3, REPMD outperforms PMD significantly in our experiments. Theorem 1 further shows that REPMD also enjoys similar desirable properties, but on a surrogate reward $\mathrm{SR}(\pi_\theta)$.

Theorem 1. *Given the project step $D_{\mathrm{KL}}(\bar{\pi}^*_{\tau,\tau'}\|\pi_\theta)$ can be solved globally optimally, REPMD satisfies the following properties for an arbitrary parametrization of π.*

1. **(Monotonic Improvement Guarantee)** *Assume that π_{θ_t} is the update sequence, then*

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

where

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

2. **(Global optimum inclusion)** *Every stationary point of the expected reward $SR(\pi_\theta)$, including the globally optimal policy π_{θ^*}, is a fixed point of REPMD.*

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

Ruitong: Double check

Jincheng: Add proof in A.3

Ruitong: Why?

Furthermore, as shown in Proposition 3, for the one-layer-softmax neural network π, the condition in Theorem 1, that $D_{\mathrm{KL}}(\bar{\pi}^*_{\tau,\tau'}\|\pi_\theta)$ can be solved globally optimally, can now be satisfied in practice.

Proposition 3. *Assume $\pi_\theta(s) = \mathrm{softmax}(\phi_s^\top\theta)$. 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.*

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

Although $\mathrm{SR}(\pi_\theta)$ is different from 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 1. *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 $\tau' = 0$.*

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

Algorithm 1 The REPMD algorithm

Input: τ, τ', K
Output: Policy π_θ
1: Random initialized π_{θ_1};
2: **For** $t = 1, 2, \ldots, T$ **do**
3: Set $\bar{\pi} = \pi_{\theta_t}$;
4: **Repeat**
5: Sample a mini-batch of K trajectories from $\bar{\pi}$;
6: Compute the gradient according to Eq. (8);
7: Update $\pi_{\theta_{t+1}}$ by the gradient;
8: **Until** converged or reach max_iter;
9: Return π_{θ_T}.

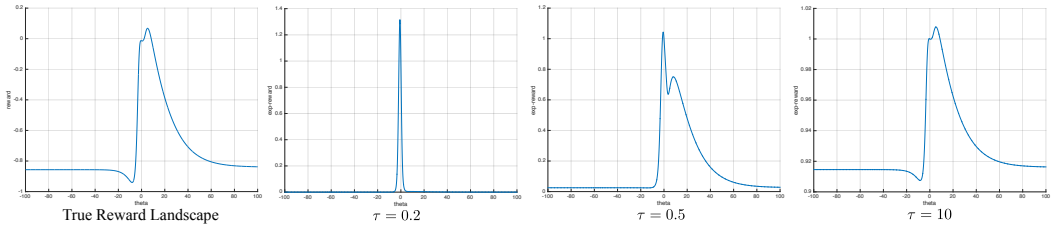


Figure 1: Simulation result for true reward landscape and $\mathrm{SR}(\pi)$ with different value of τ.

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.

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). We empirical investigate the behavior of SRπ on a simulation setting, as shown in Fig. 1. Note that there is a poor local maximum for $\theta < 0$, where naive gradient method will converge to if θ is initialized on the left. Picking $\tau = 0.2$, the reward landscape of $SR(\pi)$ guides θ to converges the neighbourhood of 0, avoiding such poor local maximum. Later in the training when τ increases to 10, $SR(\pi)$ recovers the true expected reward landscape, where θ will converge to good local , if not the global, maximum. While a simulation result may be still preliminary, it suggests that $\mathrm{SR}(\pi)$ could be a good guidance for maximizing the true expected reward in some cases. Further investigation and a principled way to find such an ideal sequence of (τ, τ') is left for future work.

Ruitong: Why?

Ruitong: Add the simulation setting in the caption of the figure.

Ruitong: Add the simulation result for true loss and SR(π) with different value of τ.

2.3 Learning

We now discuss the implementation details of REPMD. The full algorithm is presented in Algorithm 2. Despite of its non-convexity, it can be verified that the lift step has an analytic solution

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

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

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

2.4 Cooperate with Value Function Approximation

Given a reference policy $\bar{\pi}$ and an initial state s, recall that the objective in the lift step is

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

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, as follows.

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

Let $\bar{\pi}^*_{\tau,\tau'}(\cdot|s) = \arg\max_\pi \mathcal{O}_{\text{REPMD}}(\pi, s)$ denote the optimal policy on state s. Further denote the soft optimal state value function $\mathcal{O}_{\text{REPMD}}(\bar{\pi}^*_{\tau,\tau'}(\cdot|s), s)$ by $\bar{V}^*_{\tau,\tau'}(s)$, and let $\bar{Q}^*_{\tau,\tau'}(s,a) = r(s,a) + \gamma \bar{V}^*_{\tau,\tau'}(s')$ be the soft-Q function. It can be verified that

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

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 Eq. (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

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

This suggests a Monte-Carlo estimation of $\bar{V}^*_{\tau,\tau'}(s)$: by sampling one single 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,

$$L(\phi) = \mathbb{E}_{s\sim\mathcal{D}}\left[\frac{1}{2}\left(V_\phi(s) - [Q_\psi(s,a) - \tau'\log\bar{\pi}(a|s)]\right)^2\right] \tag{10}$$

where $\mathcal{D}$ is a reply buffer.

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 help stabilize the training (Haarnoja et al., 2018). 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. The soft Q-function parameters ψ is then trained to minimize the soft Bellman error using the target state value network,

$$L(\psi) = \mathbb{E}_{(s,a,s')\sim\mathcal{D}}\left[\frac{1}{2}\left(Q_\psi(s,a) - \left[r(s,a) + \gamma V_{\bar{\phi}}(s')\right]\right)^2\right] \tag{11}$$

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

$$L(\theta) = \mathbb{E}_{s\sim\mathcal{D}}\left[D_{\text{KL}}\left(\exp\left\{\frac{Q_\psi(s,\cdot) + \tau\log\bar{\pi}(\cdot|s) - V_\phi(s)}{\tau+\tau'}\right\}\middle\|\,\pi_\theta(\cdot|s)\right)\right] \tag{12}$$

where we approximate $\bar{\pi}^*_{\tau,\tau'}$ by the soft-Q and state value function approximations. The next theorem shows that the gradient of Eq. (12) can be computed by importance sampling using the reference policy,

Theorem 3. *The gradient of Eq.* (12) *is,*

$$\nabla_\theta L(\theta) = \nabla_\theta \mathbb{E}_{s\sim\mathcal{D}}\left[\mathbb{E}_{a\sim\bar{\pi}}\left[\exp\left\{\frac{Q_\psi(s,a) - \tau'\log\bar{\pi}(a|s) - V_\phi(s)}{\tau+\tau'}\right\}\log\pi_\theta(a|s)\right]\right]. \tag{13}$$

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.

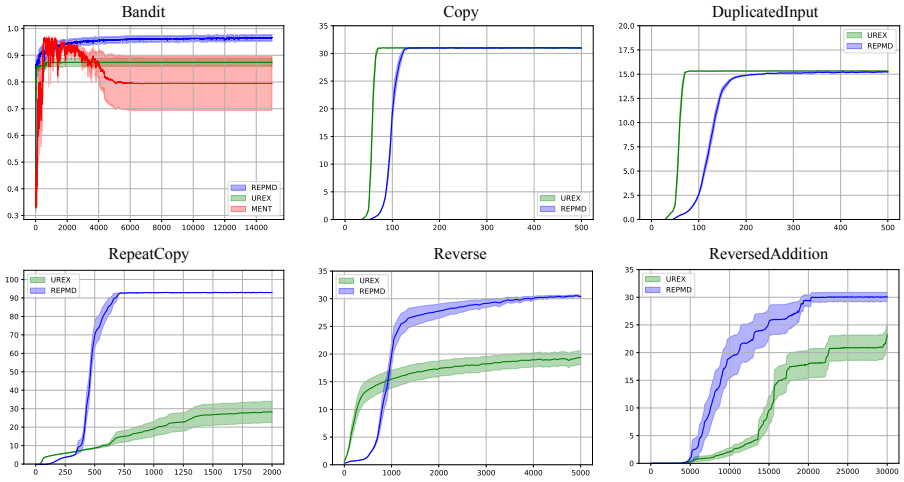


Figure 2: Results using the best hyper-parameters for each method: MENT (red), UREX (green), and REPMD (blue). Plots show average reward with standard error during training. Synthetic bandit results averaged over 5 runs. Algorithmic task results averaged over 25 random training runs (5 runs $\times$ 5 random seeds for neural network initialization). X-axis is number of sampled trajectories.

3 EXPERIMENTS

We evaluate REPMD and **** on a number of benchmark tasks. We first introduce the tasks used for testing REPMD and *****, then present the experiment results. Implementation details are provided in Appendix.

3.1 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.

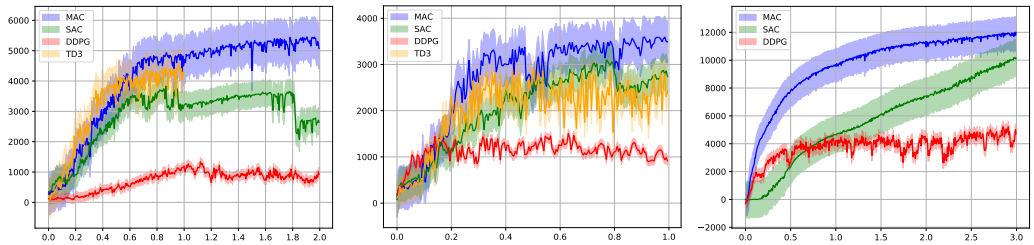
We further test our method on five algorithmic tasks from the OpenAI gym library, in rough order of difficulty: Copy, DuplicatedInput, RepeatCopy, Reverse, and ReversedAddition (Brockman et al., 2016). In each problem, the agent operates on a tape of characters or digits. At each time step, the agent 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 details of the tasks are provided in the Appendix.

Lastly, we test ***** using standard continuous-control benchmarks from OpenAI Gym utilizing the Mujoco environment (Brockman et al., 2016; Todorov et al., 2012), including Hopper, Walker2d, HalfCheetah, Ant and Humanoid.

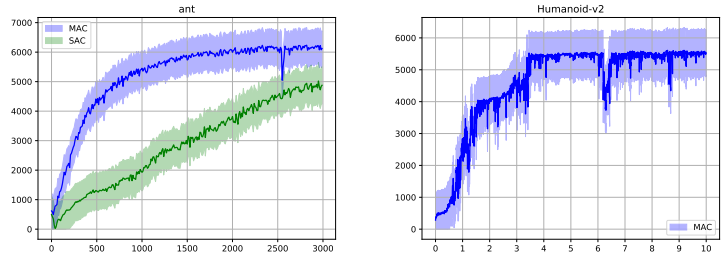
3.2 IMPLEMENTATION DETAILS

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 2, 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



(a) Walker2d (b) Hopper (c) Halfcheetah



(e) Ant (f) Humanoid

Figure 3: Learning curves on continuous control benchmarks.

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 [2].

3.3 Comparative Evaluation

3.3.1 Experiments of REPMD

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 (2). 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

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

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[3]. Note that the performance of REPMD is sill significantly better than UREX even compared with the results reported in Nachum et al. (2017a).

3.3.2 EXPERIMENTS OF *****

For continuous control tasks, we compare ***** to deep deterministic policy gradient (DDPG) (), an efficient off-policy deep RL methods, twin delayed deep deterministic policy gradient algorithm (TD3), a recent extension to DDPG by applying the double Q-learning trick to address over-estimation problem when function approximation is used, and Soft-Actor-Critic (SAC), a recent state-of-the-art off-policy algorithm on a number of continuous control benchmarks. Figure **??** presents the total average return of evaluation rollouts during training of all algorithms on continuous control benchmarks. Results of each algorithm are averaged over five different instances with different random seeds. The solid curves corresponds to the mean and the shaded region to the standard errors over the five trials. The results show that...

3.4 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.

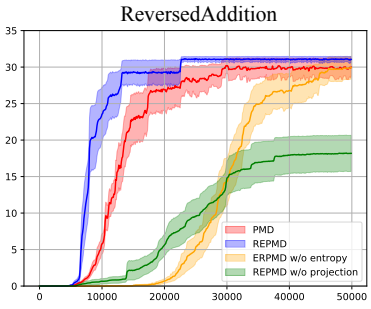


Figure 4: Ablation Study.

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 $\arg\max_{\pi_\theta \in \Pi} \mathbb{E}_{\rho \sim \pi_\theta} r(\rho) - \tau \mathrm{KL}(\pi_\theta \| \bar{\pi}) + \tau' \mathcal{H}(\pi_\theta)$.

Results on ReversedAddition are reported in Figure (4). It clearly shows that optimizing policy by performing the *mean seeking* KL divergence projection is very important as suggested in REPMD.

4 RELATED WORKS

The lift-and-project formulation differs our method from most of the previous policy search methods. To our best knowledge, four methods in the literature also use such lift-and-project formulation: Mirror Descent Guided Policy Search (MDGPS) (Montgomery & Levine, 2016), Guide Actor-Critic (GAC) (Tangkaratt et al., 2017), Maxmimum aposteriori (MPO) (Abdolmaleki et al., 2018), and Soft Actor-Critic (SAC) (Haarnoja et al., 2018). MDGPS (Montgomery & Levine, 2016) has a fundamentally different learning principle 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. MPO (Abdolmaleki et al., 2018) does not have this entropy regularizer neither. As shown in Section 3.4, such entropy term with an annealing τ' could significantly improve the efficiency of the algorithm. Both GAC and SAC use the mode seeking direction of KL divergence for the project step, different from the mean seeking direction in our method (Tangkaratt et al., 2017; Haarnoja et al., 2018). Furthermore, compared to our method, SAC uses only the entropy regularizer in the lift step (no relative entropy regularizer). The benefit of the

[3]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).

relative entropy has been discussed in TRPO (Schulman et al., 2015) and MPO (Abdolmaleki et al., 2018) that it could significantly help stabilize the learning. GAC tends to match the mean of Gaussian policies in the project step instead of directly minimizing the KL divergence with gradient descent. For other "one-step" methods, although they could or could not be reformulated into a lift-and-project procedure, they would still obliviously differs from REPMD, 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 2.1, 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, with a mean seeking KL divergence (Schulman et al., 2015). The monotonical improvement guarantee only exists for an impractical formulation of TRPO [4]. 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 3, 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 3, 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.

5 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.

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

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.

Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*. MIT press, 1998.

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.

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*, pp. 5026–5033. IEEE, 2012.

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 PROOFS FOR SECTION 2

This section is devoted to the omitted proofs in Section 2.

A.1 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 $\bar{\pi}$. □

A.2 PROOF OF PROPOSITION 2

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

(Global optimum inclusion) The stationary point of $\mathbb{E}_{\rho\sim\pi_\theta} r(\rho)$ is the π_θ which satisfies,

$$\begin{aligned} &\sum_\rho r(\rho) \cdot \frac{d\pi_\theta(\rho)}{d\theta} = 0 \\ \iff &\sum_\rho \left[\log \pi_\theta(\rho) - \log \pi_\theta(\rho) - \frac{r(\rho)}{\tau}\right] \cdot \frac{d\pi_\theta(\rho)}{d\theta} = 0 \qquad (\tau > 0) \\ \iff &\sum_\rho \left[\log \pi_\theta(\rho) - \log\{\pi_\theta(\rho) \exp\{r(\rho)/\tau\}\}\right] \cdot \frac{d\pi_\theta(\rho)}{d\theta} = 0. \end{aligned} \tag{14}$$

On the other hand, the fixed point of PMD indicates at some iteration t,

$$\begin{aligned} &\pi_{\theta_t} = \pi_{\theta_{t+1}}, \\ &\text{where } \pi_{\theta_t} \xrightarrow{\text{Lift Step}} \bar{\pi}_\tau^* \xrightarrow{\text{Project Step}} \pi_{\theta_{t+1}} \text{ in Eq. (3)}, \end{aligned} \tag{15}$$

which means π_{θ_t} is the solution of the Project Step,

$$\begin{aligned} &\left.\frac{dD_{\mathrm{KL}}(\pi_\theta \| \bar{\pi}_\tau^*)}{d\theta}\right|_{\theta=\theta_t} = 0 \\ \iff &\left.\sum_\rho \left[\log \pi_\theta(\rho) - \log \bar{\pi}_\tau^*(\rho) + 1\right] \cdot \frac{d\pi_\theta(\rho)}{d\theta}\right|_{\theta=\theta_t} = 0 \\ \iff &\left.\sum_\rho \left[\log \pi_\theta(\rho) - \log\left\{\frac{\pi_{\theta_t}(\rho) \exp\{r(\rho)/\tau\}}{\sum_{\rho'} \pi_{\theta_t}(\rho') \exp\{r(\rho')/\tau\}}\right\} + 1\right] \cdot \frac{d\pi_\theta(\rho)}{d\theta}\right|_{\theta=\theta_t} = 0 \\ &\qquad\qquad \text{(by Lift Step in Eq. (15))} \\ \iff &\left.\sum_\rho \left[\log \pi_\theta(\rho) - \log\{\pi_{\theta_t}(\rho) \exp\{r(\rho)/\tau\}\} + c\right] \cdot \frac{d\pi_\theta(\rho)}{d\theta}\right|_{\theta=\theta_t} = 0, \end{aligned} \tag{16}$$

where we denote $c = 1 + \log \sum_{\rho'} \pi_{\theta_t}(\rho') \exp\{r(\rho')/\tau\}$. Note that for c independent of ρ, we have,

$$\left.\sum_\rho c \cdot \frac{d\pi_\theta(\rho)}{d\theta}\right|_{\theta=\theta_t} = c \cdot \left.\frac{d\sum_\rho \pi_\theta(\rho)}{d\theta}\right|_{\theta=\theta_t} = c \cdot \left.\frac{d1}{d\theta}\right|_{\theta=\theta_t} = 0.$$

Therefore, Eq. (16) is equivalent with,

$$\iff \left.\sum_\rho \left[\log \pi_\theta(\rho) - \log\{\pi_{\theta_t}(\rho) \exp\{r(\rho)/\tau\}\}\right] \cdot \frac{d\pi_\theta(\rho)}{d\theta}\right|_{\theta=\theta_t} = 0. \tag{17}$$

Comparing Eq. (17) with Eq. (14), we have the fixed point condition of PMD is the same as the definition of the stationary point of $\mathbb{E}_{\rho\sim\pi_\theta} r(\rho)$. □

A.3 PROOF OF THEOREM 1

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

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

(Global optimum inclusion) The stationary point of $\mathrm{SR}(\pi_\theta)$ is the π_θ which satisfies,

$$
\begin{aligned}
&\frac{d\mathrm{SR}(\pi_\theta)}{d\theta} = 0 \\
\iff&(\tau+\tau')\cdot\frac{\sum_\rho \exp\left\{\frac{r(\rho)+\tau\log\pi_\theta(\rho)}{\tau+\tau'}\right\}\cdot\frac{\tau}{\tau+\tau'}\cdot\frac{1}{\pi_\theta(\rho)}\cdot\frac{d\pi_\theta(\rho)}{d\theta}}{\sum_{\rho'}\exp\left\{\frac{r(\rho')+\tau\log\pi_\theta(\rho')}{\tau+\tau'}\right\}} = 0 \\
\iff&-\sum_\rho \frac{\pi_\theta(\rho)\exp\left\{\frac{r(\rho)-\tau'\log\pi_\theta(\rho)}{\tau+\tau'}\right\}}{\sum_{\rho'}\pi_\theta(\rho')\exp\left\{\frac{r(\rho')-\tau'\log\pi_\theta(\rho')}{\tau+\tau'}\right\}}\cdot\frac{1}{\pi_\theta(\rho)}\cdot\frac{d\pi_\theta(\rho)}{d\theta} = 0. \qquad (\tau>0)
\end{aligned}
\tag{18}
$$

On the other hand, the fixed point of REPMD indicates at some iteration t,

$$
\begin{aligned}
&\pi_{\theta_t} = \pi_{\theta_{t+1}}, \\
&\text{where } \pi_{\theta_t} \xrightarrow{\text{Lift Step}} \bar{\pi}^*_{\tau,\tau'} \xrightarrow{\text{Project Step}} \pi_{\theta_{t+1}} \text{ in Eq. (5)},
\end{aligned}
\tag{19}
$$

which means π_{θ_t} is the solution of the Project Step,

$$
\begin{aligned}
&\frac{dD_{\mathrm{KL}}(\bar{\pi}^*_{\tau,\tau'}\|\pi_\theta)}{d\theta}\bigg|_{\theta=\theta_t} = 0 \\
\iff&-\sum_\rho \bar{\pi}^*_{\tau,\tau'}(\rho)\cdot\frac{1}{\pi_\theta(\rho)}\cdot\frac{d\pi_\theta(\rho)}{d\theta}\bigg|_{\theta=\theta_t} = 0 \\
\iff&-\sum_\rho \frac{\pi_{\theta_t}(\rho)\exp\left\{\frac{r(\rho)-\tau'\log\pi_{\theta_t}(\rho)}{\tau+\tau'}\right\}}{\sum_{\rho'}\pi_{\theta_t}(\rho')\exp\left\{\frac{r(\rho')-\tau'\log\pi_{\theta_t}(\rho')}{\tau+\tau'}\right\}}\cdot\frac{1}{\pi_\theta(\rho)}\cdot\frac{d\pi_\theta(\rho)}{d\theta}\bigg|_{\theta=\theta_t} = 0 \\
&\qquad\qquad\text{(by Lift Step in Eq. (19))}
\end{aligned}
\tag{20}
$$

Comparing Eq. (20) with Eq. (18), we have the fixed point condition of REPMD is the same as the definition of the stationary point of $\mathrm{SR}(\pi_\theta)$. □

A.4 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,

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

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

A.5 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 $\tau' \to 0$, $\mathrm{SR}(\pi_\theta) \to \mathbb{E}_{\rho\sim\pi_\theta} r(\rho)$. □

A.6 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),

$$\mathop{\mathbb{E}}_{\rho\sim\pi} r(\rho) - \tau D_{\mathrm{KL}}(\pi\|\bar{\pi}) + \tau'\mathcal{H}(\pi) = \mathop{\mathbb{E}}_{\rho\sim\pi}[r(\rho) + \tau\log\bar{\pi}(\rho)] + (\tau+\tau')\mathcal{H}(\pi), \tag{21}$$

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.

$$\bar{\pi}^*_{\tau,\tau'}(\rho) \propto \exp\left\{\frac{r(\rho)+\tau\log\bar{\pi}(\rho)}{\tau+\tau'}\right\} = \bar{\pi}(\rho)\exp\left\{\frac{r(\rho)-\tau'\log\bar{\pi}(\rho)}{\tau+\tau'}\right\}. \tag{22}$$

□

A.7 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,

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

$$= -\frac{1}{K}\sum_{k=1}^K \frac{\bar{\pi}(\rho_k)\exp\left\{\frac{r(\rho_k)-\tau'\log\bar{\pi}(\rho_k)}{\tau+\tau'}\right\}}{\bar{\pi}(\rho_k)\sum_{\rho'}\bar{\pi}(\rho')\exp\left\{\frac{r(\rho')-\tau'\log\bar{\pi}(\rho')}{\tau+\tau'}\right\}}\nabla_\theta\log\pi_\theta(\rho_k) \qquad \text{by Eq. (7)}$$

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

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

□

A.8 PROOF OF THEOREM 3

Proof. Let $\pi(a|s) = \exp\left\{\frac{Q_\psi(s,a)+\tau\log\bar{\pi}(a|s)-V_\phi(s)}{\tau+\tau'}\right\}$, then we have,

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

□

B STOCHASTIC TRANSITION SETTING

In Section 1.1, we assume that the state transition function is deterministic for simplicity. For completeness, we consider the general stochastic transition setting here.

B.1 NOTATIONS AND SETTINGS

Recall in Section 1.1, 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., $f(\rho) = 1, \forall\rho$.

B.2 REPMD OPTIMIZATION PROBLEM

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

$$\begin{aligned}
&\textbf{(Project Step)} \quad \arg\min_{\pi_\theta\in\Pi} D_{\mathrm{KL}}(p_{\bar{\pi}^*_{\tau,\tau'},f}\|p_{\pi_\theta,f}), \\
&\textbf{(Lift Step)} \quad \text{where } \bar{\pi}^*_{\tau,\tau'} = \arg\max_{\pi\in\Delta_f} \mathbb{E}_{\rho\sim p_{\pi,f}}[r(\rho) - \tau'\log\pi(\rho)] - \tau D_{\mathrm{KL}}(p_{\pi,f}\|p_{\pi_{\theta_t},f}).
\end{aligned} \tag{23}$$

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

Like Eq. (5), $\bar{\pi}^*_{\tau,\tau'}$ in Eq. (23) also has a closed form expression,

Lemma 2. *The unconstrained optimal policy of Eq.* (23) *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. (23) 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,

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

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 $f(\rho) = 1, \forall\rho$.

B.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 4. *Assume that π_{θ_t} is the update sequence of the REPMD algorithm in Eq.* (23)*, then*

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

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

$$\mathrm{SR}_f(\pi_{\theta_{t+1}}) - \mathrm{SR}_f(\pi_{\theta_t}) = (\tau+\tau')\log\sum_{\rho}\frac{f(\rho)\exp\left\{\frac{r(\rho)+\tau\log\pi_{\theta_{t+1}}(\rho)}{\tau+\tau'}\right\}}{\sum_{\rho} f(\rho)\exp\left\{\frac{r(\rho)+\tau\log\pi_{\theta_t}(\rho)}{\tau+\tau'}\right\}}$$

$$=(\tau+\tau')\log\sum_{\rho}\frac{f(\rho)\exp\left\{\frac{r(\rho)+\tau\log\pi_{\theta_t}(\rho)}{\tau+\tau'}\right\}}{\sum_{\rho} f(\rho)\exp\left\{\frac{r(\rho)+\tau\log\pi_{\theta_t}(\rho)}{\tau+\tau'}\right\}}\cdot\exp\left\{\frac{\tau\log\pi_{\theta_{t+1}}(\rho) - \tau\log\pi_{\theta_t}(\rho)}{\tau+\tau'}\right\}$$

$$=(\tau+\tau')\log\sum_{\rho}\bar{\pi}^*_{\tau,\tau'}(\rho)f(\rho)\cdot\exp\left\{\frac{\tau\log\pi_{\theta_{t+1}}(\rho) - \tau\log\pi_{\theta_t}(\rho)}{\tau+\tau'}\right\}$$

$$\geq(\tau+\tau')\sum_{\rho}\bar{\pi}^*_{\tau,\tau'}(\rho)f(\rho)\cdot\log\exp\left\{\frac{\tau\log\pi_{\theta_{t+1}}(\rho) - \tau\log\pi_{\theta_t}(\rho)}{\tau+\tau'}\right\}$$

$$=\tau\sum_{\rho}\bar{\pi}^*_{\tau,\tau'}(\rho)f(\rho)\cdot\log\frac{\pi_{\theta_{t+1}}(\rho)}{\pi_{\theta_t}(\rho)}$$

$$=\tau\sum_{\rho}\bar{\pi}^*_{\tau,\tau'}(\rho)f(\rho)\cdot\log\frac{\pi_{\theta_{t+1}}(\rho)f(\rho)}{\pi_{\theta_t}(\rho)f(\rho)}$$

$$=\tau\left[D_{\mathrm{KL}}(p_{\bar{\pi}^*_{\tau,\tau'},f}\|p_{\pi_{\theta_t},f}) - D_{\mathrm{KL}}(p_{\bar{\pi}^*_{\tau,\tau'},f}\|p_{\pi_{\theta_{t+1}},f})\right] \geq 0.$$ □

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

Proposition 5. *$SR_f(\pi_\theta)$ satisfies the following properties:*

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

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

B.4 Learning

The REPMD learning process is intact under the stochastic transition setting. Similar with Appendix A.7, we can estimate the KL divergence in the projection step of Eq. (23) by drawing K *i.i.d.* samples $\{\rho_1,\dots,\rho_K\}$ 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,

$$\begin{aligned}
D_{\mathrm{KL}}(p_{\bar\pi^*_{\tau,\tau'},f}\|p_{\pi_\theta,f})&=\mathop{\mathbb{E}}_{\rho\sim p_{\bar\pi^*_{\tau,\tau'},f}}\left[\log\bar\pi^*_{\tau,\tau'}(\rho)-\log\pi_\theta(\rho)\right]\\
&=\mathop{\mathbb{E}}_{\rho\sim p_{\bar\pi,f}}\frac{\bar\pi^*_{\tau,\tau'}(\rho)}{\bar\pi(\rho)}\left[\log\bar\pi^*_{\tau,\tau'}(\rho)-\log\pi_\theta(\rho)\right].
\end{aligned}\tag{24}$$

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. (24).

Theorem 5. *Let* $\omega_k=\frac{r(\rho_k)-\tau'\log\bar\pi(\rho_k)}{\tau+\tau'}$. *Given* K i.i.d. *samples* $\{\rho_1,\dots,\rho_K\}$ *from the* reference policy $\bar\pi$, *we have the following unbiased gradient estimator,*

$$\nabla_\theta D_{\mathrm{KL}}(p_{\bar\pi^*_{\tau,\tau'},f}\|p_{\pi_\theta,f})\approx-\sum_{k=1}^{K}\frac{\exp\{\omega_k\}}{\sum_{j=1}^{K}\exp\{\omega_j\}}\nabla_\theta\log\pi_\theta(\rho_k),\tag{25}$$

Proof. See Appendix A.7 for the proof of Theorem 2. □

C Details of Learning Algorithm

Details of the learning algorithms of REPMD and *** are provided in this section.

C.1 Details of REPMD

Pseudocode for REPMD is presented in Algorithm 2.

Algorithm 2 The REPMD algorithm

Input: temperature parameters τ and τ', number of samples for computing gradient K
1: Random initialized π_{θ_1};
2: **For** $t=1,2,\dots$ **do**
3: Set $\bar\pi=\pi_\theta$
4: **Repeat**
5: Sample a mini-batch of K trajectories from $\bar\pi$
6: Compute the gradient according to Eq. (8)
7: Update π_{θ_t} by gradient descent
8: **Until** converged or reach maximum of training steps
9: Return π_{θ_T}.

C.2 DETAILS OF *****

Pseudocode for *** is presented in Algorithm 3. The major difference between *** and repmd in the lift step is that instead of sampling K actions as described in Algorithm 2, **** only samples one action to construct $\bar{\pi}^*_{\tau,\tau'}$, due to the fact both the soft-Q and state value function approximations are adopted. The value function approximations also make *** capable of using off-policy data from a reply buffer.

Algorithm 3 The **** algorithm

Input: temperature parameters τ and τ', lag on target value network α

Initialize $\pi_\theta, V_\phi, V_{\bar{\phi}}, Q_{\psi_1}, Q_{\psi_2}$ and replay buffer $\mathcal{D}$
For $t = 1, 2, \ldots$ **do**
  For each environment step **do**
    $a \sim \pi_\theta(\cdot|s)$
    Observe s' and r from environment
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s, a, r, s')\}$
  Set $\bar{\pi} = \pi_\theta$
  For each training step **do**
    Sample a mini-batch of data $\mathcal{M} = \{(s_i, a_i, r_i, s'_i)\}_{i=1}^B$ from $\mathcal{D}$
    Compute gradient $\nabla_\theta L(\theta), \nabla_\phi L(\phi), \nabla_{\psi_1} L(\psi_1), \nabla_{\psi_2} L(\psi_2)$according to Eqs. (10) to (12)
    Update parameters $\theta, \phi, \psi_1, \psi_2$ by gradient descent
    Update $\bar{\phi}$ by $\alpha\phi + (1-\alpha)\bar{\phi}$