Average-Reward Reinforcement Learning with Entropy Regularization

Jacob Adamczyk[1,2], **Volodymyr Makarenko**[3], **Stas Tiomkin**[4], **Rahul V. Kulkarni**[1,2]
[1]Department of Physics, University of Massachusetts Boston
[2]The NSF Institute for Artificial Intelligence and Fundamental Interactions
[3]Department of Computer Engineering, San José State University
[4]Department of Computer Science, Whitacre College of Engineering, Texas Tech University
jacob.adamczyk001@umb.edu, volodymyr.makarenko@sjsu.edu, stas.tiomkin@ttu.edu, rahul.kulkarni@umb.edu

Abstract

The average-reward formulation of reinforcement learning (RL) has drawn increased interest in recent years due to its ability to solve temporally-extended problems without discounting. Independently, RL algorithms have benefited from entropy-regularization: an approach used to make the optimal policy stochastic, thereby more robust to noise. Despite the distinct benefits of the two approaches, the combination of entropy regularization with an average-reward objective is not well-studied in the literature and there has been limited development of algorithms for this setting. To address this gap in the field, we develop algorithms for solving entropy-regularized average-reward RL problems with function approximation. We experimentally validate our method, comparing it with existing algorithms on standard benchmarks for RL.

Introduction

A successful reinforcement learning (RL) agent learns from interacting with its surroundings to achieve desired behaviors, as encoded in a reward function. Thus, in "continuing" tasks, where the amount of interactions is potentially unlimited, the total sum of rewards received by the agent is unbounded. To avoid this divergence, the most popular technique is to *discount* future rewards relative to current rewards. The framework of discounted RL enjoys convergence properties (Sutton and Barto 2018; Kakade 2003; Bertsekas 2012), practical benefits (Schulman et al. 2016; Andrychowicz et al. 2020), and a plethora of useful algorithms (Mnih et al. 2015; Schulman et al. 2015, 2017; Hessel et al. 2018; Haarnoja et al. 2018b) making the discounted objective an obvious choice for the RL practitioner. Despite these benefits, the use of discounting introduces a (typically unphysical) hyperparameter γ which must be tuned for optimal performance. The difficulty in properly tuning γ is illustrated in our motivating example in Figure 1. Furthermore, agents solving the discounted RL problem will fail in optimizing long-term behaviors requiring timescales longer than those allowed by discounting, and it has been argued that the discounted objective is not even a well-defined optimization problem (Naik et al. 2019). Moreover, despite most state-of-the-art algorithms using this discounted framework, their metric for performance is often the total or average reward over trajectories, as opposed to the discounted sum.

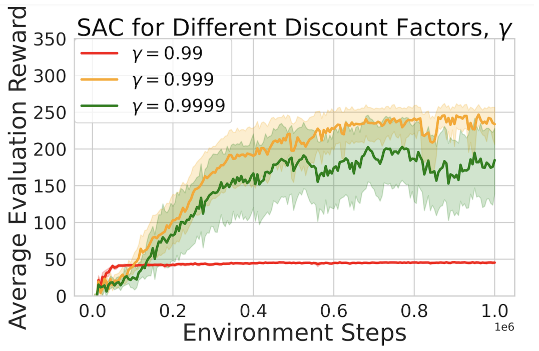

Figure 1: The Swimmer-v4 environment, often not included in Mujoco benchmarks (Franceschetti et al. 2022), is notoriously difficult for discounted methods to solve, as the discount factor is not tuned over and set to its default value of $\gamma = 0.99$. Other discount-sensitive examples of environments have been discussed by (Tessler and Mannor 2020). We find that after carefully tuning the discount factor, SAC can solve the task, but the solution is quite sensitive to the choice of γ. Each curve corresponds to an average over 30 random seeds, with the standard error indicated by the shaded region.

To address these issues, another criterion for solving continuing tasks has been studied (Schwartz 1993; Mahadevan 1996): the average-reward objective. Although it is arguably a more natural choice, it has less obvious convergence properties. Importantly, there are limited average-reward algorithms beyond the tabular setting. Current algorithms beyond tabular settings use policy-gradient methods to develop actor-based models: (Zhang and Ross 2021; Ma et al. 2021; Saxena et al. 2023). While these advancements represent a positive step for average-reward RL algorithms, there remains a need for alternative approaches to develop the field of average-reward deep RL.

In both the discounted and average-reward scenarios, the optimal policy is known to be deterministic (Mahadevan 1996; Sutton and Barto 2018). However, due to observational and control noise, a deterministic policy can fail in important circumstances relevant for real-world usage of RL: e.g. when transferred to other environments or when perturbations arise (Haarnoja et al. 2017, 2018a; Eysenbach and Levine 2022). To address such cases, it would be desirable to have a stochastic optimal policy. Rather than using heuristics (e.g. ε-greedy, Boltzmann) to generate a stochastic policy, the original RL problem can be regularized with an entropy-based term that yields an optimal policy which is naturally stochastic. Implementing this entropy-regularized RL objective corresponds to rewarding the agent (in proportion to a temperature parameter, β^{-1}) for using a policy which has a lower relative entropy (Levine 2018). This formulation of entropy-regularized (often considered in the special case of maximum entropy or "MaxEnt"[1]) RL has led to significant developments in state-of-the-art off-policy algorithms (Haarnoja et al. 2017, 2018b,c).

Despite the desirable features of both the average-reward and entropy-regularized objectives, the combination of these formulations (Neu, Jonsson, and Gómez 2017) is not as well-studied, and no function-approximator algorithms exist yet for this setting. To address this issue, we propose novel algorithms for average-reward RL with entropy regularization which are analogs of the discounted algorithms Soft Actor-Critic (SAC) (Haarnoja et al. 2018b,c) and Soft Q-learning (SQL) (Haarnoja et al. 2017).

Main Contributions

- We extend the policy improvement theorem to the entropy-regularized average-reward case;
- Provide novel algorithms for average-reward deep RL in the entropy-regularized setting;
- Experimentally validate our approach in discrete and continuous control environments.

Notably, our implementation requires minimal changes to common codebases, making it accessible for researchers and allowing for future extensions by the community.

Preliminaries

In this section, we focus on the development of theory for discrete state-action spaces for clarity. Let $\Delta(\mathcal{X})$ denote the probability simplex over the space $\mathcal{X}$. A Markov Decision Process (MDP) is modeled by a state space $\mathcal{S}$, action space $\mathcal{A}$, reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, transition dynamics $p : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ and initial state distribution $\mu \in \Delta(\mathcal{S})$. The state space describes the set of possible configurations in which the agent (and environment) may exist. (This can be juxtaposed with the "observation" which encodes the state information accessible to the agent. We will consider fully observable MDPs where state and observation are synonymous.) The action space is the set of controls available to the agent. Enacting control, the agent may alter its state in the environment. This change is dictated by the (generally stochastic) transition dynamics, p.

At each discrete timestep, an action is taken and the agent receives a reward $r(s, a) \in \mathbb{R}$ from the environment.

[1]MaxEnt refers to using a uniform prior policy. In this work, we consider the case of more general priors.

We will make some of the usual assumptions for average-reward MDPs:

Assumption 1. The Markov chain induced by any stationary policy π is irreducible and aperiodic.

Assumption 2. The reward function is bounded.

In solving an average-reward MDP, one seeks a control policy π which maximizes the expected *reward-rate*, denoted ρ^π. In the average-reward framework, such an objective reads

$$\rho^\pi = \lim_{N\to\infty} \frac{1}{N} \mathop{\mathbb{E}}_{\tau\sim p,\pi,\mu} \left[\sum_{t=0}^{N-1} r(s_t, a_t) \right], \tag{1}$$

where the expectation is taken over trajectories generated by the dynamics p, control policy π, and initial state distribution μ.

The non-scalar (that is, (s, a)-dependent) contribution to the action-value function is called the average-reward differential bias function. Because of its analogy to the Q function in discounted RL, we follow recent work (Zhang and Ross 2021) and similarly denote it as:

$$Q^\pi_\rho(s, a) = \mathop{\mathbb{E}}_{\tau\sim p,\pi} \left[\sum_{t=0}^{\infty} r(s_t, a_t) - \rho^\pi \middle| s_0 = s, a_0 = a \right]. \tag{2}$$

We will now introduce a variation of this MDP framework which includes an entropy regularization term. For notational convenience we refer to entropy-regularized average-reward MDPs as ERAR MDPs. The ERAR MDP constitutes the same ingredients as an average-reward MDP stated above, in addition to a pre-specified prior policy[2] $\pi_0 : \mathcal{S} \to \Delta(\mathcal{A})$ and "inverse temperature", β. The modified objective function for an ERAR MDP now includes a regularization term based on the relative entropy (Kullback-Leibler divergence),

so that the agent now aims to optimize the expected *entropy-regularized reward-rate* , denoted θ^π:

$$\theta^\pi = \lim_{N\to\infty} \frac{1}{N} \mathop{\mathbb{E}}_{\tau\sim p,\pi,\mu} \left[\sum_{t=0}^{N-1} r(s_t, a_t) - \frac{1}{\beta} \log \frac{\pi(a_t|s_t)}{\pi_0(a_t|s_t)} \right], \tag{3}$$

$$\pi^*(a|s) = \operatorname*{argmax}_\pi \theta^\pi. \tag{4}$$

Beyond the mathematical generalization from the MaxEnt formulation, the KL divergence term has also found use in behavior-regularized RL tasks, especially in the offline setting (Wu, Tucker, and Nachum 2019; Zhang and Tan 2024).

Because of Assumption 1, the expression in Equation (3) is independent of the initial state-action. From hereon, we will simply write $\theta = \theta^{\pi^*}$ for the optimal entropy-regularized reward-rate for brevity. Comparing to Equation (1), this rate is seen to include an additional entropic contribution, the relative entropy between the control π and prior π_0.

[2]We will assume for convenience that π_0 has support across $\mathcal{A}$, ensuring the Kullback-Leibler divergence between any policy π and π_0 remains finite.

The differential entropy-regularized action-value function is then given by [3]

$$
\begin{aligned}
&Q_\theta^\pi(s,a) = r(s,a) - \theta^\pi \\
&+ \mathop{\mathbb{E}}_{\tau\sim p,\pi}\left[\sum_{t=1}^{\infty}\left(r(s_t,a_t) - \frac{1}{\beta}\log\frac{\pi(a_t|s_t)}{\pi_0(a_t|s_t)} - \theta^\pi\right)\right]. \quad (5)
\end{aligned}
$$

We have used the subscripts of θ and ρ in this section to distinguish the two value functions. In the following, we will drop the θ subscript as we focus solely on the entropy-regularized objective. Similarly, we will write $Q(s,a) = Q_\theta^{\pi^*}(s,a)$ as a shorthand.

Prior Work

Research on average-reward MDPs has a longstanding history, dating back to seminal contributions by (Blackwell 1962) and later (Mahadevan 1996), laying foundational groundwork for further algorithmic and theoretical investigations (Even-Dar, Kakade, and Mansour 2009; Abbasi-Yadkori et al. 2019; Abounadi, Bertsekas, and Borkar 2001; Neu, Jonsson, and Gómez 2017; Wan, Naik, and Sutton 2021). Due to their theoretical nature, these studies primarily focused on algorithms within tabular settings, possibly explaining why the average-reward approach is not as widely used in the deep RL community. Recent work has begun to focus on this challenge, tackling deep average-reward RL (Zhang and Ross 2021; Ma et al. 2021; Saxena et al. 2023) with policy-based methods. These studies have indicated superior performance of average-reward algorithms in continuous control Mujoco (Todorov, Erez, and Tassa 2012) tasks compared to standard (discounted objective) baselines.

In parallel, the discounted objective has seen the introduction of an entropy-regularization term, discussed in works such as (Todorov 2006, 2009; Ziebart 2010; Rawlik 2013; Haarnoja et al. 2017; Geist, Scherrer, and Pietquin 2019). The included "entropy bonus" term has found considerable use in the development of both theory and algorithms in distinct branches of RL research (Haarnoja et al. 2018a; Eysenbach and Levine 2022; Park et al. 2023). This innovation allows the derivation of optimal policies naturally exhibiting stochasticity in continuous action spaces, which has led SAC (Haarnoja et al. 2018c) and its variants to become state-of-the-art solution methods for addressing the discounted objective.

However, there is limited work on the combination of average-reward and entropy-regularized methods for deep RL. Recent work by (Rawlik 2013; Rose, Mair, and Garrahan 2021; Arriojas et al. 2023b; Li, Wu, and Lan 2022; Wu, Ke, and Wu 2024) has set the groundwork for combining the entropy-regularized and average-reward formulations. We will leverage their results to address the problem of deep average-reward RL with entropy regularization, while also including new theoretical results. In the following, we present average-reward extensions of the two canonical methods of solving for Q: soft actor-critic (SAC) and soft Q-learning (SQL).

[3]We have suppressed the conditioning on the initial state and action for brevity, but the conditioning is identical to that in Equation (2).

Soft Actor-Critic for the Average-Reward Objective

We begin with a discussion of the extension of soft actor-critic (SAC), for which we derive new theoretical results and provide the corresponding average-reward algorithm. First, we draw inspiration from SAC (Haarnoja et al. 2018b) which relies on iteratively calculating a value (critic) of a policy (actor) and improving the actor through soft policy improvement (PI). In the discounted problem formulation, soft PI states that a Boltzmann-form for the policy (π') derived from the value function of a previous policy (π) such that: $\pi' \propto \exp \beta Q^\pi(s,a)$, is guaranteed to outperform the previous policy in the Q-value sense: $Q^{\pi'}(s,a) > Q^\pi(s,a)$ for all s,a (cf. Lemma 2 of (Haarnoja et al. 2018b) for details). We will first show that an analogous result for policy improvement holds in the ERAR setting.

Since the value of a policy is now encoded in the entropy-regularized average reward rate θ^π and *not* in the differential value, the analogue to policy improvement is to establish the bound $\theta^{\pi'} > \theta^\pi$ for some construction of π' from π. Indeed, as we show, the same Boltzmann form over the differential value leads to soft PI in the ERAR objective. After establishing PI and the related theory in this setting we present our algorithm, denoted "ASAC" (for average-reward SAC, and following the naming convention of APO (Ma et al. 2021) and ATRPO (Zhang and Ross 2021)).

Theory

As in the discounted case, it can be shown that the Q function for a fixed policy π satisfies a recursive Bellman backup equation[4]:

Proposition 1 ((Wu, Ke, and Wu 2024)). *Let an ERAR MDP be given with reward function $r(s,a)$, fixed evaluation policy π and prior policy π_0. Then the differential value of π, $Q_\theta^\pi(s,a)$, satisfies*

$$Q_\theta^\pi(s,a) = r(s,a) - \theta^\pi + \mathbb{E}_{s'\sim p} V_\theta^\pi(s'), \quad (6)$$

with the entropy-regularized definition of state-value function

$$V_\theta^\pi(s) = \mathbb{E}_{a\sim\pi}\left[Q^\pi(s,a) - \frac{1}{\beta}\log\frac{\pi(a|s)}{\pi_0(a|s)}\right]. \quad (7)$$

In the average reward formulation, the metric of interest is the reward-rate (θ^π). Our policy improvement result therefore focuses on increases in θ^π, generalizing the recent work of (Zhang and Ross 2021) to the current setting. We find that the gap between any two entropy-regularized reward-rates can be expressed in the following manner.

[4]Equation (7) is an extension of V_{soft}^π in (Haarnoja et al. 2017) to the case of non-uniform prior policy.

Lemma 1 (ERAR Rate Gap). *Consider two policies π, π' absolutely continuous w.r.t. π_0. Then the gap between their corresponding entropy-regularized reward rates is:*

$$\theta^{\pi'} - \theta^{\pi} = \mathop{\mathbb{E}}_{\substack{s\sim d_{\pi'} \\ a\sim\pi'}} \left(A^{\pi}_{\theta}(s,a) - \frac{1}{\beta}\log\frac{\pi'(a|s)}{\pi_0(a|s)} \right), \quad (8)$$

where $A^{\pi}_{\theta}(s,a) = Q^{\pi}_{\theta}(s,a) - V^{\pi}_{\theta}(s)$ is the advantage function of policy π and $d_{\pi'}$ is the steady-state distribution induced by π'.

The proof of this result, and all other results stated in this section, can be found in Appendix A. As a consequence of this result, we find that a proper choice of the policy π' guaranteed to make the right-hand side of Equation (8) positive implies that PI holds. Using the Boltzmann form of a policy (Haarnoja et al. 2018b) with the differential Q-values as the energy function and the appropriate prior distribution (π_0), gives the desired result:

Theorem 1 (ERAR Policy Improvement). *Let a policy π absolutely continuous w.r.t. π_0 and its corresponding differential value $Q^{\pi}_{\theta}(s,a)$ be given. Then, the policy*

$$\pi'(a|s) \doteq \frac{\pi_0(a|s)e^{\beta Q^{\pi}_{\theta}(s,a)}}{\sum_a \pi_0(a|s)e^{\beta Q^{\pi}_{\theta}(s,a)}} \quad (9)$$

achieves a greater entropy-regularized reward-rate. That is, $\theta^{\pi'} \geq \theta^{\pi}$, with equality only at convergence, when $\pi' = \pi = \pi^*$.

It is noteworthy that the corresponding result in Lemma 2 of (Haarnoja et al. 2018b) for SAC (which uses a uniform prior policy), involves the *total* value function. On the other hand, in the ASAC case PI is determined by the *differential* value function. Intuitively, this result can be understood as the $\gamma \to 1$ limit of PI for SAC since the "bulk" contribution to the total value function, which comes from accruing a per-timestep reward θ^{π}, will cancel in the numerator and denominator of Equation (9). In the case of SAC, the bulk contribution is included in the evaluation of the total value function and a discount factor $\gamma < 1$ is required to ensure that the total value function is bounded in the limit of large N (in the sense of Equation (3)). In contrast, for the case of ASAC, the bulk contribution is excluded from the corresponding evaluation (by definition), and the differential value function remains bounded in the limit of large N, obviating the need to introduce a discount factor.

Finally, to ensure the convergence of ASAC, the policy evaluation step must also converge.

Lemma 2 (ERAR Policy Evaluation). *Consider a fixed policy π, for which θ^{π} of Equation* (1) *has been calculated (e.g. with direct rollouts). The iteration of Equations* (2) *and* (7) *converges to the entropy-regularized differential value of π: $Q^{\pi}_{\theta}(s,a)$.*

Proof. The proof follows from the convergence results established in the un-regularized case, e.g. (Wan, Naik, and Sutton 2021). Since the policy π is fixed, the entropic cost $-\beta^{-1}\mathrm{KL}\left(\pi||\pi_0\right)$ can be incorporated into the reward function's definition. □

Algorithm

As in SAC (Haarnoja et al. 2018c), we propose to interleave steps of policy evaluation (PE) and policy improvement (PI) using stochastic approximation to train the actor and critic networks. Let Q_{ψ} denote the online critic network while $Q_{\bar{\psi}}$ denotes the target critic, updated periodically through Polyak averaging of the parameters. To implement a PI step, we use the KL divergence loss to update the parameters of an actor π_{ϕ} based on the policy improvement theorem (Equation (9)):

$$\mathcal{L}_{\phi} = \sum_{s\in\mathcal{B}} \mathrm{KL}\left(\pi_{\phi}(\cdot|s) \middle|\middle| \frac{\pi_0(\cdot|s)e^{\beta Q_{\psi}(s,\cdot)}}{Z(s)}\right) \quad (10)$$

And now we update the critic (differential value) by performing a policy evaluation step with the current actor network. The mean squared error loss is calculated by comparing the expected Q value to the right-hand side of Equation (6):

$$\mathcal{L}_{\psi} = \sum_{s,a,r,s'\sim\mathcal{B}} \left| Q_{\psi}(s,a) - \hat{y} \right|^2, \quad (11)$$

where $\hat{y}$ is the target:

$$\hat{y} = r - \theta + \mathop{\mathbb{E}}_{a'\sim\pi_{\phi}(\cdot|s')} \left[Q_{\bar{\psi}}(s',a') - \frac{1}{\beta}\log\frac{\pi_{\phi}(a'|s')}{\pi_0(a'|s')} \right].$$

Finally, we update θ by considering its definition as in Equation (3):

$$\theta_{k+1} = (1-\tau_{\theta})\theta_k + \tau_{\theta}\left[\sum_{s,a,r} \left(r - \frac{1}{\beta}\log\frac{\pi(a|s)}{\pi_0(a|s)} \right) \right], \quad (12)$$

where τ_{θ} is a learning rate for θ, and samples in the sum are taken uniformly from the replay buffer. We note that this is not exact in the sense that the samples from the replay buffer will possibly mix the θ^{π} from different policies. However, if we assume that the batch is a representative sample of the current policy's steady-state distribution, then the term in brackets represents an unbiased estimate of θ^{π}.

Experimental Validation

For our extension of soft actor-critic we use continuous action environments of various complexity including Pendulum-v1, HalfCheetah-v4, Ant-v4, Swimmer-v4 (see Appendix) from the Gymnasium Mujoco suite (Todorov, Erez, and Tassa 2012; Towers et al. 2024). We compare the performance (average evaluation reward across 10 episodes) for ASAC (ours), SAC, and arDDPG (which was shown to

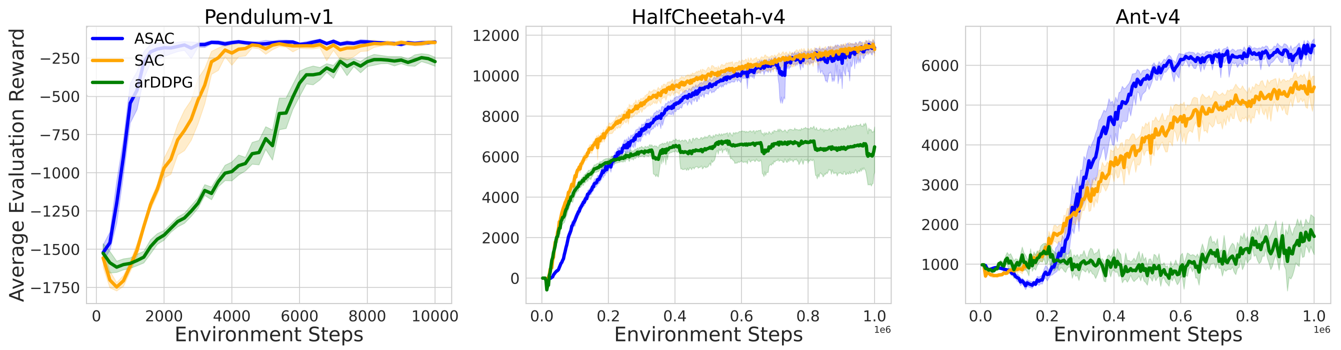

Figure 2: Mujoco benchmark comparing soft actor critic (SAC), average-reward deep deterministic policy gradient (arDDPG) and ours: average-reward soft actor critic ASAC. Each curve corresponds to an average over 20 random seeds, with standard errors indicated by the shaded region.

outperform the other known average-reward algorithms of ATRPO (Zhang and Ross 2021) and APO (Ma et al. 2021)).

The results of these experiments are shown in Figure 2. Broadly, we see an increase in state-of-the-art performance on the average-reward objective (comparing ASAC to arDDPG), and even an improvement against the discounted algorithm SAC. However, we do note that our algorithm did require some further hyperparameter tuning and requires further work for more complex domains. We provide further details on the implementation and hyperparameter sweeps and selection in Appendices B, C. All code for reproducing the experiments is available at https://anonymous.4open.science/r/u-chi-learning-521B/

Soft Q-Learning for the Average-Reward Objective

In this section, we present an extension of the SQL algorithm to the average-reward setting, which we denote ASQL. For simplicity we focus on the case of discrete actions, in which case the necessary action integrals can be calculated exactly.

Theory

In soft Q-learning (Haarnoja et al. 2017), one refines estimates of the optimal (soft) Q function by iterating the Bellman optimality operator which is defined as $\mathcal{T}(\cdot) \doteq r(s,a) + \gamma\beta^{-1}\,\mathbb{E}_{s'} \log \mathbb{E}_{a'} \exp \beta\,(\cdot)$. Since $\gamma \in (0,1)$, this operator is a contraction, and hence converges to its (unique) fixed point, the optimal Q-function: $\mathcal{T}^{\infty}(Q_0) = \mathcal{T}(Q^*) = Q^*$. In the ASQL case, the lack of discount factor poses a challenge in terms of ensuring convergence. However, for the case of deterministic dynamics, it can be shown by mapping to exponential space that the corresponding undiscounted equation (Equation (13)) is simply an eigenvector equation (Arriojas et al. 2023b), which has a unique non-trivial solution corresponding to the entropy-regularized average-reward rate (eigenvalue) and differential Q function (eigenvector). Additional details about this connection are provided in the Appendix.

Furthermore, we obtain the optimality equation for the entropy-regularized differential value, allowing us to solve the ERAR MDP through Q-learning iterations:

Lemma 3. *The optimal ERAR Q-function satisfies the following Bellman equation:*

$$Q(s,a) = r(s,a) - \theta + \frac{1}{\beta} \mathop{\mathbb{E}}_{s'\sim p} \log \mathop{\mathbb{E}}_{a'\sim\pi_0} e^{\beta Q(s',a')}, \quad (13)$$

where θ is the maximum ERAR rate, $\theta = \max_\pi \theta^\pi$ (with θ^π given in Equation (3)*).*

The proof follows from the limit of policy improvement (given in Appendix A). In accordance with SQL's temporal difference update, we propose to iterate Equation (13) until convergence.

Algorithm

To implement this procedure in the spirit of SQL, we use a mean squared error loss between an online network and corresponding estimate of Equation (13) calculated through a target network,

$$\mathcal{J}(\psi) = \frac{1}{2} \mathop{\mathbb{E}}_{s,a,r,s'\sim\mathcal{D}} \left(Q_\psi(s,a) - \hat{y}\right)^2, \quad (14)$$

where the target is

$$\hat{y} = r - \theta + \beta^{-1} \log \mathop{\mathbb{E}}_{a'\sim\pi_0} e^{\beta Q_{\bar{\psi}}(s',a')}, \quad (15)$$

and where $\mathcal{D}$ denotes a replay buffer based on past experience. Note that the temporal difference (TD) target is calculated from a target network (lagging weights relative to the online network), denoted $\bar{\psi}$. Appendix B contains further implementation details. The algorithm's pseudocode is shown in Algorithm 1, with the main differences from SQL highlighted in red. The value of θ must be updated online as well, and we use the same Equation (12) as discussed in the previous section to update its value. The value of θ is updated after averaging its new value over the "gradient steps" loop in Algorithm 1.

The algorithm implements three key components present in many value-based deep RL methods: (1) an estimate of the value function parameterized by two deep neural nets (inspired by (Van Hasselt, Guez, and Silver 2016)),

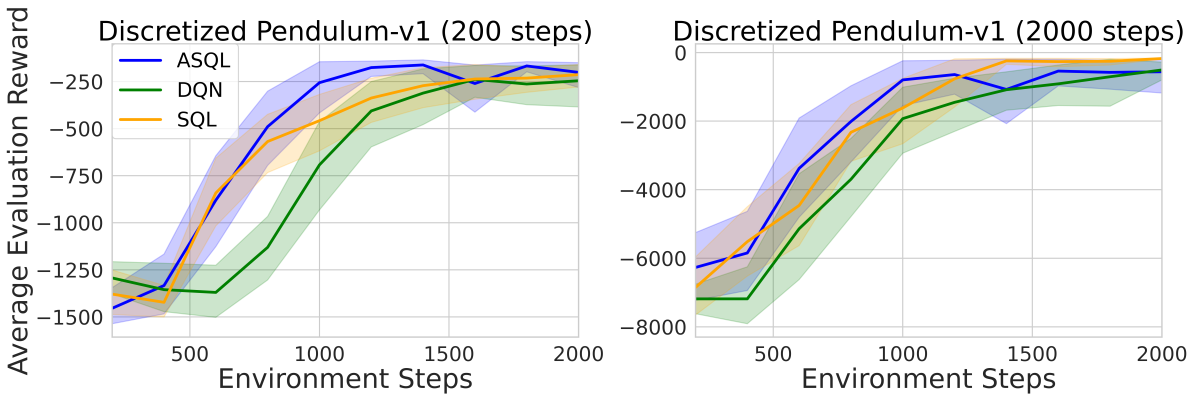

Figure 3: For ASQL, we use a discrete action Pendulum environment and compare against DQN and SQL. We use finetuned hyperparameters, and find that ASQL has a strong performance regardless of episode length, expected based on the discussion in (Saxena et al. 2023).

Algorithm 1: ASQL

```
1:  IN: sample budget, environment, $\beta$, hyperparameters
2:  Initialize:
3:  Online network weights: $\psi_i \sim$ init. distribution
4:  Target network weights: $\bar{\psi}_i = \psi_i$
5:  Entropy-regularized reward-rate: $\theta = 0$
6:  Replay buffer: $\mathcal{D} = \{\}$
7:  while $t <$ sample budget do
8:      Collect experience:
9:      Sample action $a \sim \pi_0$
10:     Take step in environment $s' \sim p(\cdot|s, a)$
11:     Save to replay buffer: $\mathcal{D} \leftarrow \{s, a, r, s'\}$
12:     if train this step then
13:         for each gradient step do
14:             Sample minibatch $\mathcal{B} \subset \mathcal{D}$
15:             Calculate loss via Equation (14)
16:             Do gradient descent on $Q$ network(s)
17:             Update $\theta$ via Equation (12)
18:         end for
19:     end if
20:     if update target parameters then
21:         Update target parameters (Polyak averaging with
            parameters $\tau_\psi$).
22:     end if
23: end while
24: OUT: Optimal policy for ERAR-MDP
```

(2) stochastic gradient descent on a TD error with Adam (Kingma and Ba 2015) and (3) a replay buffer of stored experience. In principle, the replay buffer can be collected by any behavior policy such as an ε-greedy policy (Mnih et al. 2015), but we use the learnt policy (obtained from the exponential in Equation (9)) for more effective exploration as in (Haarnoja et al. 2018b).

Inspired by (Van Hasselt, Guez, and Silver 2016; Fujimoto, Hoof, and Meger 2018), we will train two online networks in parallel. We find this to help considerably improve the evaluation reward. Interestingly, we have found that the popular choice of taking a pessimistic estimate (i.e. $\min$) is not optimal in our experiments. Instead we treat the "aggregation function" as a hyperparameter, and tune it over the choices of $\min, \max, \mathrm{mean}$. Across all environments, we have found $\max$ to be the optimal choice for aggregation. We use two online networks and two corresponding target networks for updates, and upon calculating an aggregated estimate of $Q_{\bar{\psi}}$ we use the $\max$ over the two target networks, calculated (s, a)-wise: $Q_{\bar{\psi}}(s, a) = \max\left\{Q^{(1)}_{\bar{\psi}}(s, a), Q^{(2)}_{\bar{\psi}}(s, a)\right\}$. We train both of the online networks independently to minimize the same loss in Equation (14), measured against the aggregated target value. This is the same technique employed in (Fujimoto, Hoof, and Meger 2018) for discounted un-regularized RL and in (Haarnoja et al. 2018c) for discounted regularized RL. We use the same aggregation method $(\max)$ on the online networks $Q^{(i)}_{\psi}$ to calculate θ across the current batch of data via Equation (3).

Experimental Validation

For experimental benchmarking of ASQL we consider discrete action environments. Specifically, we take a discrete action version of Pendulum-v1 (3 actions: min. torque, zero torque, and max. torque) and the Pong game from the Atari suite. We provide benchmark experiments, comparing ASQL against DQN (Mnih et al. 2015) (as it shares many implementation details) and SQL (Haarnoja et al. 2017) (for its entropy-regularized objective). Although we recognize that these algorithms are not designed to optimize the same objective function, we do not have other benchmarks with which to compare in the discrete-action setting. On the other hand, we emphasize that the metric of interest in discounted RL is often the average evaluation reward. Against this metric, ASQL has stronger performance in terms of sample complexity compared to DQN and SQL.

Using DQN from (Raffin et al. 2021) and our own im-

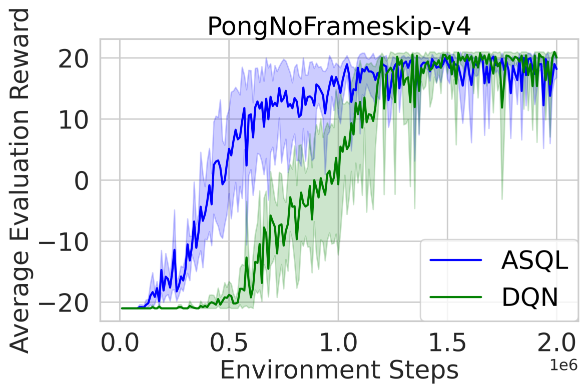

Figure 4: PongNoFrameskip-v4 with DeepMind preprocessing. The main text and Appendix B discusses further implementation details. ASQL outperforms DQN in terms of sample complexity. Each curve corresponds to an average over 10 random seeds, with the standard error plotted by the shaded region.

plementation of discrete-action SQL (Haarnoja et al. 2017), we provide benchmarks in Figures 3 and 4. Hyperparameter tuning has been performed for each algorithm, with the associated values shown in Appendix C. Then, with those tuned hyperparameters, we test the algorithms on a longer episode version (which corresponds to keeping the Pendulum upright for 2000 steps rather than 200), finding that ASQL continues to perform well for both cases.

Note that the evaluation phase for all algorithms uses the greedy policy derived from the learned stochastic policy as is standard. Every one thousand environment steps, we pause the training to perform the evaluation, averaged over ten episodes. The resulting reward curve from each algorithm is then averaged over twenty random initializations.

To test the capability of ASQL on more complex discrete action environments, we turn to the Atari suite (Bellemare et al. 2013). Using the standard pre-processing techniques (downsampling, square cropping, greyscaling, frameskipping, and framestacking), we use a CNN (rather than MLP as in all other experiments) model for the differential Q function architecture. The resulting comparison for Pong is shown in Fig. 4 where ASQL shows a favorable performance against the (discounted) DQN baseline.

Discussion

The motivation for developing novel algorithms for average rewards RL arises from the problems generally associated with discounting. When the RL problem is posed in the discounted framework, a discount factor γ is a required input parameter. However, there is often no principled approach for choosing the value of γ corresponding to the specific problem being addressed. Thus, the experimenter must treat γ as a hyperparameter. This reduces the choice of γ to a trade-off between large values to capture long-term rewards and small values to capture computational efficiency which typically scales polynomially with the horizon, $(1-\gamma)^{-1}$ (Kakade 2003). The horizon introduces a natural timescale to the problem, but this timescale may not be well-aligned with another timescale corresponding to the optimal dynamics: the mixing time of the induced Markov chain. For the discounted approach to accurately estimate the optimal policy, the discounting timescale (horizon) must be larger than the mixing time; however the estimation of the mixing time for the optimal dynamics can be challenging to obtain in the general case, even when the transition dynamics are known. Therefore, an arbitrary "sufficiently large" choice of γ is often made without knowledge of the relevant problem-dependent timescale. This can be problematic from a computational standpoint as evidenced by recent work (Jiang et al. 2015; Schulman et al. 2017; Andrychowicz et al. 2020). These points are illustrated in Figure 1 which shows the performance of SAC for the Swimmer-v4 environment for different choices of γ. For the widely used choice $\gamma = 0.99$ the evaluation rewards are low relative to the optimal case, whereas the average rewards algorithms perform well (cf. Appendix), highlighting the benefits of using the average-rewards framework.

In this work, we have developed a framework for combining the benefits of the average rewards approach with entropy regularization. In particular, we have focused on extensions of the discounted algorithms SAC and SQL to the average rewards domain. By leveraging the connection of the ERAR objective to the soft discounted framework, we have presented the first solution to ERAR MDPs in continuous state and action spaces by use of function approximation. Our experiments suggest that ASQL and ASAC compare favorably in several respects to their discounted counterparts. Our algorithm leverages existing codebases allowing for a straightforward and easily extendable implementation for solving the ERAR objective.

Future Work

The current work suggests multiple extensions which we plan to explore in future research. Following the line of work in (Arriojas et al. 2023b), recent work by the same authors shows a method for the more general case of stochastic transition dynamics, by iteratively learning biases for the dynamics and rewards (Arriojas et al. 2023a). With a model-based algorithm, this seems to be a promising avenue for future exploration for an alternative approach to ASQL in stochastic environments. As a value-based technique, other ideas from the literature such as TD(n), REDQ (Chen et al. 2021), DrQ (Kostrikov, Yarats, and Fergus 2020), combating estimation bias (Hussing et al. 2024), or dueling architectures (Wang et al. 2016) may be included. From the perspective of sampling, the calculation of θ can likely benefit from more complex replay sampling, e.g. PER (Schaul et al. 2015). An important contribution for future work is studying the sample complexity and convergence properties of the proposed algorithms. We believe that the average-reward objective with entropy regularization is a fruitful direction for further research and real-world application, with this work addressing a gap in the existing literature.

Acknowledgements

JA would like to acknowledge the use of the supercomputing facilities managed by the Research Computing Department at UMass Boston; the Unity high-performance computing cluster; and funding support from the Alliance Innovation Lab – Silicon Valley. RVK and JA would like to acknowledge funding support from the NSF through Award No. DMS-1854350 and PHY-2425180. ST was supported in part by NSF Award (2246221), PAZY grant (195-2020), and WCoE, Texas Tech U. This work is supported by the National Science Foundation under Cooperative Agreement PHY-2019786 (The NSF AI Institute for Artificial Intelligence and Fundamental Interactions, http://iaifi.org/).

References

Abbasi-Yadkori, Y.; Bartlett, P.; Bhatia, K.; Lazic, N.; Szepesvári, C.; and Weisz, G. 2019. Politex: Regret bounds for policy iteration using expert prediction. In *International Conference on Machine Learning*, 3692–3702. PMLR.

Abounadi, J.; Bertsekas, D.; and Borkar, V. S. 2001. Learning Algorithms for Markov Decision Processes with Average Cost. *SIAM Journal on Control and Optimization*, 40(3): 681–698.

Andrychowicz, M.; Raichuk, A.; Stańczyk, P.; Orsini, M.; Girgin, S.; Marinier, R.; Hussenot, L.; Geist, M.; Pietquin, O.; Michalski, M.; et al. 2020. What matters in on-policy reinforcement learning? a large-scale empirical study. *arXiv preprint arXiv:2006.05990*.

Arriojas, A.; Adamczyk, J.; Tiomkin, S.; and Kulkarni, R. V. 2023a. Bayesian inference approach for entropy regularized reinforcement learning with stochastic dynamics. In Evans, R. J.; and Shpitser, I., eds., *Proceedings of the Thirty-Ninth Conference on Uncertainty in Artificial Intelligence*, volume 216 of *Proceedings of Machine Learning Research*, 99–109. PMLR.

Arriojas, A.; Adamczyk, J.; Tiomkin, S.; and Kulkarni, R. V. 2023b. Entropy regularized reinforcement learning using large deviation theory. *Phys. Rev. Res.*, 5: 023085.

Bellemare, M. G.; Naddaf, Y.; Veness, J.; and Bowling, M. 2013. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47: 253–279.

Bertsekas, D. 2012. *Dynamic programming and optimal control: Volume I*, volume 4. Athena scientific.

Blackwell, D. 1962. Discrete dynamic programming. *The Annals of Mathematical Statistics*, 719–726.

Chen, X.; Wang, C.; Zhou, Z.; and Ross, K. W. 2021. Randomized Ensembled Double Q-Learning: Learning Fast Without a Model. In *International Conference on Learning Representations*.

Even-Dar, E.; Kakade, S. M.; and Mansour, Y. 2009. Online Markov decision processes. *Mathematics of Operations Research*, 34(3): 726–736.

Eysenbach, B.; and Levine, S. 2022. Maximum Entropy RL (Provably) Solves Some Robust RL Problems. In *International Conference on Learning Representations*.

Franceschetti, M.; Lacoux, C.; Ohouens, R.; Raffin, A.; and Sigaud, O. 2022. Making reinforcement learning work on swimmer. *arXiv preprint arXiv:2208.07587*.

Fujimoto, S.; Hoof, H.; and Meger, D. 2018. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, 1587–1596. PMLR.

Geist, M.; Scherrer, B.; and Pietquin, O. 2019. A theory of regularized Markov decision processes. In *International Conference on Machine Learning*, 2160–2169. PMLR.

Haarnoja, T.; Pong, V.; Zhou, A.; Dalal, M.; Abbeel, P.; and Levine, S. 2018a. Composable deep reinforcement learning for robotic manipulation. In *2018 IEEE international conference on robotics and automation (ICRA)*, 6244–6251. IEEE.

Haarnoja, T.; Tang, H.; Abbeel, P.; and Levine, S. 2017. Reinforcement Learning with Deep Energy-Based Policies. In Precup, D.; and Teh, Y. W., eds., *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, 1352–1361. PMLR.

Haarnoja, T.; Zhou, A.; Abbeel, P.; and Levine, S. 2018b. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In Dy, J.; and Krause, A., eds., *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, 1861–1870. PMLR.

Haarnoja, T.; Zhou, A.; Hartikainen, K.; Tucker, G.; Ha, S.; Tan, J.; Kumar, V.; Zhu, H.; Gupta, A.; Abbeel, P.; et al. 2018c. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*.

Hessel, M.; Modayil, J.; Van Hasselt, H.; Schaul, T.; Ostrovski, G.; Dabney, W.; Horgan, D.; Piot, B.; Azar, M.; and Silver, D. 2018. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Hussing, M.; Voelcker, C.; Gilitschenski, I.; Farahmand, A.-m.; and Eaton, E. 2024. Dissecting Deep RL with High Update Ratios: Combatting Value Overestimation and Divergence. *arXiv preprint arXiv:2403.05996*.

Jiang, N.; Kulesza, A.; Singh, S.; and Lewis, R. 2015. The dependence of effective planning horizon on model accuracy. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, 1181–1189.

Kakade, S. M. 2003. *On the sample complexity of reinforcement learning*. Ph.D. thesis, University College London.

Kingma, D.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR)*. San Diego, CA, USA.

Kostrikov, I.; Yarats, D.; and Fergus, R. 2020. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. *arXiv preprint arXiv:2004.13649*.

Levine, S. 2018. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*.

Li, T.; Wu, F.; and Lan, G. 2022. Stochastic first-order methods for average-reward markov decision processes. *arXiv preprint arXiv:2205.05800*.

Ma, X.; Tang, X.; Xia, L.; Yang, J.; and Zhao, Q. 2021. Average-Reward Reinforcement Learning with Trust Region Methods. In Zhou, Z.-H., ed., *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, 2797–2803. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Mahadevan, S. 1996. Average reward reinforcement learning: Foundations, algorithms, and empirical results. *Machine learning*, 22: 159–195.

Mitter, S. K.; and Newton, N. 2000. The duality between estimation and control. *Published in Festschrift for A. Bennoussan*.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533.

Naik, A.; Shariff, R.; Yasui, N.; Yao, H.; and Sutton, R. S. 2019. Discounted reinforcement learning is not an optimization problem. *arXiv preprint arXiv:1910.02140*.

Neu, G.; Jonsson, A.; and Gómez, V. 2017. A unified view of entropy-regularized Markov decision processes. *arXiv preprint arXiv:1705.07798*.

Park, S.; Lee, K.; Lee, Y.; and Abbeel, P. 2023. Controllability-Aware Unsupervised Skill Discovery. *arXiv preprint arXiv:2302.05103*.

Raffin, A.; Hill, A.; Gleave, A.; Kanervisto, A.; Ernestus, M.; and Dormann, N. 2021. Stable-Baselines3: Reliable Reinforcement Learning Implementations. *Journal of Machine Learning Research*, 22(268): 1–8.

Rawlik, K. C. 2013. *On probabilistic inference approaches to stochastic optimal control*. Ph.D. thesis, The University of Edinburgh.

Rose, D. C.; Mair, J. F.; and Garrahan, J. P. 2021. A reinforcement learning approach to rare trajectory sampling. *New Journal of Physics*, 23(1): 013013.

Saxena, N.; Khastagir, S.; Shishir, N.; and Bhatnagar, S. 2023. Off-policy average reward actor-critic with deterministic policy search. In *International Conference on Machine Learning*, 30130–30203. PMLR.

Schaul, T.; Quan, J.; Antonoglou, I.; and Silver, D. 2015. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*.

Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; and Moritz, P. 2015. Trust region policy optimization. In *International conference on machine learning*, 1889–1897. PMLR.

Schulman, J.; Moritz, P.; Levine, S.; Jordan, M.; and Abbeel, P. 2016. High-Dimensional Continuous Control Using Generalized Advantage Estimation. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Schwartz, A. 1993. A reinforcement learning method for maximizing undiscounted rewards. In *Proceedings of the tenth international conference on machine learning*, volume 298, 298–305.

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

Tessler, C.; and Mannor, S. 2020. Reward tweaking: Maximizing the total reward while planning for short horizons. *arXiv preprint arXiv:2002.03327*.

Theodorou, E. A.; and Todorov, E. 2012. Relative entropy and free energy dualities: Connections to path integral and kl control. In *2012 ieee 51st ieee conference on decision and control (cdc)*, 1466–1473. IEEE.

Todorov, E. 2006. Linearly-solvable Markov decision problems. In Schölkopf, B.; Platt, J.; and Hoffman, T., eds., *Advances in Neural Information Processing Systems*, volume 19. MIT Press.

Todorov, E. 2009. Efficient computation of optimal actions. *Proceedings of the national academy of sciences*, 106(28): 11478–11483.

Todorov, E.; Erez, T.; and Tassa, Y. 2012. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, 5026–5033. IEEE.

Towers, M.; Kwiatkowski, A.; Terry, J.; Balis, J. U.; Cola, G. D.; Deleu, T.; Goulão, M.; Kallinteris, A.; Krimmel, M.; KG, A.; Perez-Vicente, R.; Pierré, A.; Schulhoff, S.; Tai, J. J.; Tan, H.; and Younis, O. G. 2024. Gymnasium: A Standard Interface for Reinforcement Learning Environments. arXiv:2407.17032.

Van Hasselt, H.; Guez, A.; and Silver, D. 2016. Deep reinforcement learning with double Q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.

Wan, Y.; Naik, A.; and Sutton, R. S. 2021. Learning and planning in average-reward Markov decision processes. In *International Conference on Machine Learning*, 10653–10662. PMLR.

Wang, Z.; Schaul, T.; Hessel, M.; Hasselt, H.; Lanctot, M.; and Freitas, N. 2016. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, 1995–2003. PMLR.

Wu, F.; Ke, J.; and Wu, A. 2024. Inverse reinforcement learning with the average reward criterion. *Advances in Neural Information Processing Systems*, 36.

Wu, Y.; Tucker, G.; and Nachum, O. 2019. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*.

Zhang, Y.; and Ross, K. W. 2021. On-policy deep reinforcement learning for the average-reward criterion. In *International Conference on Machine Learning*, 12535–12545. PMLR.

Zhang, Z.; and Tan, X. 2024. An Implicit Trust Region Approach to Behavior Regularized Offline Reinforcement Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 16944–16952.

Ziebart, B. D. 2010. *Modeling purposeful adaptive behavior with the principle of maximum causal entropy*. Carnegie Mellon University.

Technical Appendix

A Proofs

Lemma 1 (ERAR Backup Equation)**.** *Let an ERAR MDP be given with reward function $r(s,a)$, fixed evaluation policy π and prior policy π_0. Then the differential value of π, $Q^\pi_\theta(s,a)$, satisfies*

$$Q^\pi_\theta(s,a) = r(s,a) - \theta^\pi + \mathbb{E}_{s'\sim p} V^\pi_\theta(s'), \tag{16}$$

with the entropy-regularized definition[5] *of state-value function*

$$V^\pi_\theta(s) = \mathbb{E}_{a\sim\pi}\left[Q^\pi(s,a) - \frac{1}{\beta}\log\frac{\pi(a|s)}{\pi_0(a|s)}\right]. \tag{17}$$

Proof. We begin with the definitions

$$Q^\pi_\theta(s,a) = r(s,a) - \theta^\pi + \mathop{\mathbb{E}}_{s'\sim p,\pi}\left[\sum_{t=1}^{\infty}\left(r(s_t,a_t) - \frac{1}{\beta}\log\frac{\pi(a_t|s_t)}{\pi_0(a_t|s_t)} - \theta^\pi\right)\right], \tag{18}$$

for the current state-action and

$$Q^\pi_\theta(s',a') = r(s',a') - \theta^\pi + \mathop{\mathbb{E}}_{s''\sim p,\pi}\left[\sum_{t=2}^{\infty}\left(r(s_t,a_t) - \frac{1}{\beta}\log\frac{\pi(a_t|s_t)}{\pi_0(a_t|s_t)} - \theta^\pi\right)\right], \tag{19}$$

for the next state-action.

Re-writing $Q^\pi_\theta(s,a)$ and highlighting the terms of $Q^\pi_\theta(s',a')$ in blue,

$$Q^\pi_\theta(s,a) = r(s,a) - \theta^\pi + \mathop{\mathbb{E}}_{\tau\sim p,\pi}\left[\sum_{t=2}^{\infty}\left(r(s_t,a_t) - \frac{1}{\beta}\log\frac{\pi(a_t|s_t)}{\pi_0(a_t|s_t)} - \theta^\pi\right) + r(s',a') - \frac{1}{\beta}\log\frac{\pi(a'|s')}{\pi_0(a'|s')} - \theta^\pi\right], \tag{20}$$

$$Q^\pi_\theta(s,a) = r(s,a) - \theta^\pi + \mathop{\mathbb{E}}_{s'\sim p,a'\sim\pi}\left[Q^\pi_\theta(s',a') - \frac{1}{\beta}\log\frac{\pi(a'|s')}{\pi_0(a'|s')}\right], \tag{21}$$

and identifying the entropy-regularized state value function as $V(s) = \mathbb{E}_{a\sim\pi}\left[Q^\pi_\theta(s,a) - \frac{1}{\beta}\log\frac{\pi(a|s)}{\pi_0(a|s)}\right]$. □

Lemma 1 (ERAR Rate Gap)**.** *Consider two policies π, π' absolutely continuous w.r.t. π_0. Then the gap between their corresponding entropy-regularized reward rates is:*

$$\theta^{\pi'} - \theta^\pi = \mathop{\mathbb{E}}_{s\sim d_{\pi'},a\sim\pi'}\left(A^\pi_\theta(s,a) - \frac{1}{\beta}\log\frac{\pi'(a|s)}{\pi_0(a|s)}\right), \tag{22}$$

where $A^\pi_\theta(s,a) = Q^\pi_\theta(s,a) - V^\pi_\theta(s)$ is the advantage function of policy π and $d_{\pi'}$ is the steady-state distribution induced by π'.

Proof. Working from the right-hand side of the equation,

$$\begin{aligned}
\mathop{\mathbb{E}}_{s\sim d_{\pi'},a\sim\pi'}\left(A^\pi_\theta(s,a) - \frac{1}{\beta}\log\frac{\pi(a|s)}{\pi_0(a|s)}\right) &= \mathop{\mathbb{E}}_{s\sim d_{\pi'},a\sim\pi'}\left(Q^\pi_\theta(s,a) - V^\pi_\theta(s) - \frac{1}{\beta}\log\frac{\pi'(a|s)}{\pi_0(a|s)}\right)\\
&= \mathop{\mathbb{E}}_{s\sim d_{\pi'},a\sim\pi'}\left(r(s,a) - \theta^\pi + \mathop{\mathbb{E}}_{s'\sim p} V^\pi_\theta(s') - V^\pi_\theta(s) - \frac{1}{\beta}\log\frac{\pi'(a|s)}{\pi_0(a|s)}\right)\\
&= \theta^{\pi'} - \theta^\pi + \mathop{\mathbb{E}}_{s\sim d_{\pi'},a\sim\pi'}\left(\mathop{\mathbb{E}}_{s'\sim p(\cdot|s,a)} V^\pi_\theta(s') - V^\pi_\theta(s)\right)\\
&= \theta^{\pi'} - \theta^\pi.
\end{aligned}$$

where we have used the definition

$$\theta^{\pi'} = \mathop{\mathbb{E}}_{s\sim d_{\pi'},a\sim\pi'}\left(r(s,a) - \frac{1}{\beta}\log\frac{\pi'(a|s)}{\pi_0(a|s)}\right) \tag{23}$$

and

$$\mathop{\mathbb{E}}_{s\sim d_{\pi'}}\mathop{\mathbb{E}}_{a\sim\pi'}\mathop{\mathbb{E}}_{s'\sim p} V^\pi_\theta(s') = \mathop{\mathbb{E}}_{s\sim d_{\pi'}} V^\pi_\theta(s), \tag{24}$$

which follows given that $d_{\pi'}$ is the stationary distribution. In other words, $d_{\pi'}$ is an eigenvector of the transition operator $p(s'|s,a)\cdot\pi'(a'|s')$. □

[5] Equation (17) is an extension of V^π_{soft} in (Haarnoja et al. 2017) to the case of a non-uniform prior policy.

Theorem 1 (ERAR Policy Improvement)**.** *Let a policy π absolutely continuous w.r.t. π_0 and its corresponding differential value $Q^\pi_\theta(s,a)$ be given. Then, the policy*

$$\pi'(a|s) \doteq \frac{\pi_0(a|s)e^{\beta Q^\pi_\theta(s,a)}}{\sum_a \pi_0(a|s)e^{\beta Q^\pi_\theta(s,a)}} \tag{25}$$

achieves a greater entropy-regularized reward-rate. That is, $\theta^{\pi'} \geq \theta^\pi$, with equality only at convergence, when $\pi' = \pi = \pi^*$.

Proof. Let π' be defined as above. Then

$$\frac{1}{\beta}\log\frac{\pi'(a|s)}{\pi_0(a|s)} = Q^\pi_\theta(s,a) - \frac{1}{\beta}\log \mathop{\mathbb{E}}_{a\sim\pi_0} e^{\beta Q^\pi_\theta(s,a)}. \tag{26}$$

Using Lemma 1,

$$\begin{aligned}
\theta^{\pi'} - \theta^\pi &= \mathop{\mathbb{E}}_{s\sim d_{\pi'},a\sim\pi'} \left(A^\pi_\theta(s,a) - \frac{1}{\beta}\log\frac{\pi'(a|s)}{\pi_0(a|s)}\right) \\
&= \mathop{\mathbb{E}}_{s\sim d_{\pi'},a\sim\pi'} \left(Q^\pi_\theta(s,a) - V^\pi_\theta(s) - \frac{1}{\beta}\log\frac{\pi'(a|s)}{\pi_0(a|s)}\right) \\
&= \mathop{\mathbb{E}}_{s\sim d_{\pi'},a\sim\pi'} \left(\frac{1}{\beta}\log \mathop{\mathbb{E}}_{a\sim\pi_0} e^{\beta Q^\pi_\theta(s,a)} - V^\pi_\theta(s)\right) \geq 0
\end{aligned}$$

where the last line follows from the variational formula (Mitter and Newton 2000; Theodorou and Todorov 2012),

$$\frac{1}{\beta}\log \mathop{\mathbb{E}}_{a\sim\pi_0} e^{\beta Q^\pi_\theta(s,a)} = \inf_\pi \mathop{\mathbb{E}}_{a\sim\pi} \left(Q^\pi(s,a) - \frac{1}{\beta}\log\frac{\pi(a|s)}{\pi_0(a|s)}\right). \tag{27}$$

□

A.1 Soft Q-Learning Proofs

Lemma 4. *The optimal ERAR Q function satisfies*

$$Q(s,a) = r(s,a) - \theta + \frac{1}{\beta}\mathop{\mathbb{E}}_{s'\sim p}\log \mathop{\mathbb{E}}_{a'\sim\pi_0} e^{\beta Q(s',a')}, \tag{28}$$

Proof. Policy iteration monotonically improves the ERAR rate until $\theta' = \theta$ at convergence. So by the case of equality seen in the previous proof, we have a relation between the optimal state and action value functions:

$$V(s) = \frac{1}{\beta}\log \mathop{\mathbb{E}}_{a\sim\pi_0} e^{\beta Q(s,a)} \tag{29}$$

from which the desired result follows immediately via Lemma 1. □

A.2 Eigenvector Equation

This update equation is in fact the same (in exponential space) as derived by Arriojas et al. (2023b) for deterministic dynamics. Since in the exponential space, this is an eigenvalue equation for a primitive matrix, the unique solution (eigenvector) exists and can be obtained e.g. by the power method. The eigenvector given by (Arriojas et al. 2023b) is

$$u(s,a) = e^{\beta(r(s,a)-\theta)} \mathop{\mathbb{E}}_{s'\sim p,a'\sim\pi_0} u(s',a'), \tag{30}$$

which in the case of deterministic dynamics can be written in log-space as:

$$\log u(s,a) = \beta\left(r(s,a)-\theta\right) + \mathop{\mathbb{E}}_{s'\sim p}\log \mathop{\mathbb{E}}_{a'\sim\pi_0} u(s',a'), \tag{31}$$

Upon dividing by β and defining $q = \beta^{-1}\log u$:

$$q(s,a) = r(s,a) - \theta + \frac{1}{\beta}\mathop{\mathbb{E}}_{s'\sim p}\log \mathop{\mathbb{E}}_{a'\sim\pi_0} e^{\beta q(s',a')}, \tag{32}$$

which is identical to our Equation (13).

For the case of stochastic dynamics, (Arriojas et al. 2023a) has shown that the average-reward RL problem can be mapped onto another eigenvector equation with a different (learnable) choice of dynamics and reward function. So for both cases (i.e. for deterministic dynamics and for stochastic dynamics) the update equation can be mapped on to an eigenvalue equation (in exponential space).

B Implementation Details

B.1 Atari

For the Atari environments, we use a (standard, (Mnih et al. 2015)) CNN architecture with $32, 64, 64$ channels and kernel sizes of $8, 4, 3$ and strides of $4, 2, 1$ respectively in the three convolutional layers. Finally, there is a fully connected layer of width 512 before the $|\mathcal{A}|$-dimensional output. As in (Mnih et al. 2015) we use the center and square cropping (84×84) and stack 4 frames while skipping 4. This ensures each frame renders the moving sprite (e.g. ball, enemies) and gives important information about velocities. We grey-scale the images to reduce channel dimension from 12 to 4 and use uint8 encoding of images to reduce memory size. A "FireReset" wrapper is used to ensure that the game is started at the first step of each episode, as some games require pressing the "Fire" button for initialization. Whenever a life is lost, the environment emits a termination signal. Many of these wrappers can be found in Gymnasium's wrapper subpackage ("AtariPreprocessing"). The Pong finetuned runs (shown in Fig. 4) cost ~ 40 GPU hours (A100 workstation), and we swept the ASQL hyperparameters for ~ 100 GPU hours. Each run requires roughly ~ 7 GB of RAM.

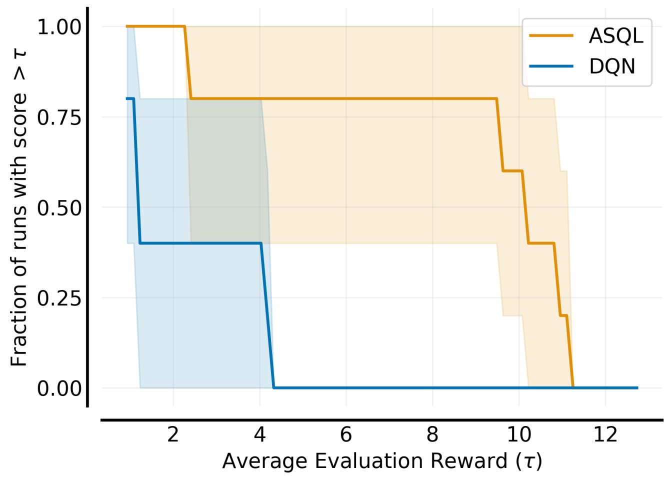

Figure 5: Performance profile of τ comparison test for the Pong environment.

B.2 MuJoCo

For all SAC runs, we used (Raffin et al. 2021) implementation of SAC with hyperparameters (beyond the default values) shown below in Section C. The finetuned runs here took $\sim$ 200 GPU hours for all environments, and sweeping the ASAC hyperparameter ranges cost $\sim$ 1000 GPU hours. Each run requires roughly $\sim$ 1 GB of RAM.

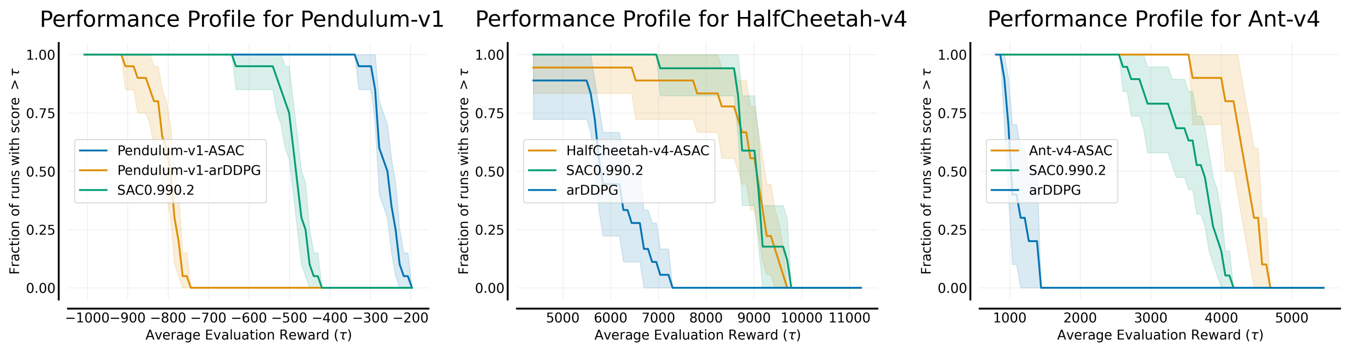

Figure 6: Performance profile of τ comparison test for the Mujoco suite.

C Hyperparameters

C.1 Atari

We use the default Atari Suite hyperparameters for DQN in (Mnih et al. 2015).

We finetuned the hyperparameters for Pong on ASQL (shown in Table 2, using the sweep ranges shown below in Table 1.

Table 1: Atari Sweep

Environment	Sweep Range
τ_θ (uniform)	$0.00 - 1.00$
batch size	16, 32
learning rate (log uniform values)	$1.00 \times 10^{-6} - 1.00 \times 10^{-3}$
β (uniform)	$1.00 \times 10^{-2} - 10.0$
target update interval	10000
τ	1
grad steps	4
train freq	4
hidden dim	512
number of networks	2
aggregation function	min, mean, max
learning starts	50000

Table 2: Finetuned Hyperparameter Values for ASQL, PongNoFrameskip-v4

Environment	Finetuned Value
τ_θ	5.00×10^{-2}
learning rate	3.50×10^{-5}
τ	1.00

C.2 MuJoCo

Table 3: MuJoCo sweep

Environment	Sweep Range
aggregation	max, min, mean
number networks	2
τ_θ (uniform)	$0.00 - 1.00$
actor learning rate	1.00×10^{-5}, 5.00×10^{-5}, 1.00×10^{-4}, 2.00×10^{-4}, 5.00×10^{-4}
learning rate	1.00×10^{-4}, 2.00×10^{-4} ,3.00×10^{-4}, 5.00×10^{-4}, 8.00×10^{-4}, 1.00×10^{-3}
buffer_size	100000, 1000000
hidden dim	256
batch size	256
target update interval	1
τ	0.0003, 0.0005, 0.0008, 0.001, 0.003, 0.005, 0.007, 0.01, 0.03, 0.05
max gradient norm	0, 10, 1000

Table 4: Finetuned Hyperparameter Values for SAC

Environment	Pendulum-v1	Ant-v4	Swimmer-v4	HalfCheetah-v4
ent coef, β^{-1}	2.00×10^{-1}	2.00×10^{-1}	5.00×10^{-2}	2.00×10^{-1}

For SAC, we set the temperature parameter fixed (shown in the table below). We found this to perform better than the automatically adjusted temperature parameter used by default. However, future work can explore the use for a learned temperature parameter for ASAC and compare these methods as well.

Table 5: Finetuned Hyperparameter Values for ASAC

Environment	Pendulum-v1	Ant-v4	Swimmer-v4	HalfCheetah-v4
learning rate	2.00×10^{-3}	2.00×10^{-4}	3.00×10^{-4}	5.00×10^{-4}
τ	5.00×10^{-2}	5.00×10^{-4}	5.00×10^{-3}	8.00×10^{-4}
τ_θ	5.00×10^{-1}	2.84×10^{-1}	6.84×10^{-1}	7.30×10^{-2}
actor learning rate	1.00×10^{-4}	1.00×10^{-5}	1.00×10^{-4}	1.00×10^{-4}

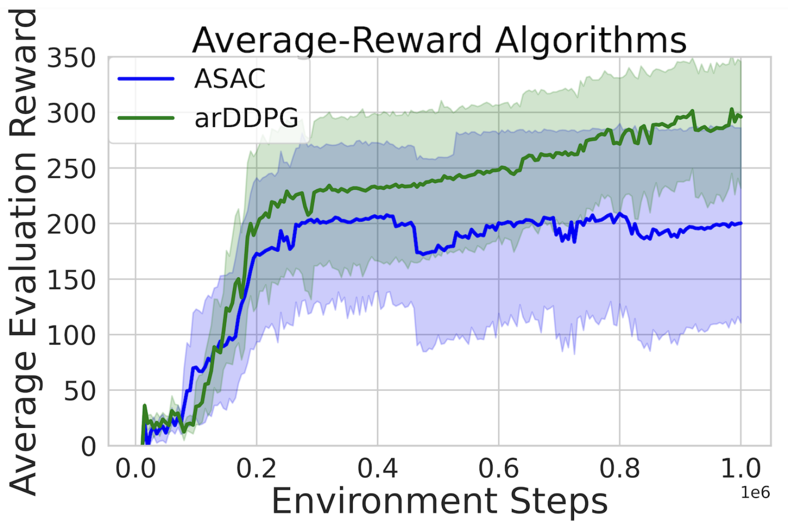

Figure 7: Results on the Swimmer-v4 environment for the undiscounted objective.

C.3 Pendulum

Table 6: Pendulum sweep

Hyperparameter	Sweep Range
τ_θ (uniform)	$0.00 - 1.00$
batch size	$32, 64, 128, 256$
learning rate (log uniform values)	$1.00 \times 10^{-4} - 1.00 \times 10^{-2}$
β (log uniform values)	$1.00 \times 10^{-2} - 10.0$
target update interval	$2, 10, 50, 100, 200, 500, 1000, 5000$
τ	1.00
max gradient norm	$10.0, 0.00$
train freq	1
hidden dim	$64, 128, 256$
learning starts	0

Table 7: Finetuned Hyperparameter Values for ASQL

Environment	Pendulum-v1
β	1.50×10^{-1}
τ_θ	2.50×10^{-2}
learning rate	3.20×10^{-3}
τ	1.00

Table 8: Finetuned Hyperparameter Values for DQN

Environment	Pendulum-v1
exploration final ε	1.00×10^{-1}
exploration fraction	1.20×10^{-1}
γ	0.99
learning rate	1.00×10^{-3}
τ	1.00

Table 9: Finetuned Hyperparameter Values for SQL

Environment	Pendulum-v1
β	1.00
γ	0.99
learning rate	1.00×10^{-3}
τ	1.00