Off-Policy Actor-Critic with Emphatic Weightings

Eric Graves   graves@ualberta.ca
Ehsan Imani   imani@ualberta.ca
Raksha Kumaraswamy   kumarasw@ualberta.ca
Martha White   whitem@ualberta.ca
Reinforcement Learning and Artificial Intelligence Laboratory
Department of Computing Science, University of Alberta
Edmonton, Alberta, Canada T6G 2E8

Editor: Joelle Pineau

Abstract

A variety of theoretically-sound policy gradient algorithms exist for the on-policy setting due to the policy gradient theorem, which provides a simplified form for the gradient. The off-policy setting, however, has been less clear due to the existence of multiple objectives and the lack of an explicit off-policy policy gradient theorem. In this work, we unify these objectives into one off-policy objective, and provide a policy gradient theorem for this unified objective. The derivation involves *emphatic weightings* and *interest functions*. We show multiple strategies to approximate the gradients, in an algorithm called Actor Critic with Emphatic weightings (ACE). We prove in a counterexample that previous (semi-gradient) off-policy actor-critic methods—particularly Off-Policy Actor-Critic (OffPAC) and Deterministic Policy Gradient (DPG)—converge to the wrong solution whereas ACE finds the optimal solution. We also highlight why these semi-gradient approaches can still perform well in practice, suggesting strategies for variance reduction in ACE. We empirically study several variants of ACE on two classic control environments and an image-based environment designed to illustrate the tradeoffs made by each gradient approximation. We find that by approximating the emphatic weightings directly, ACE performs as well as or better than OffPAC in all settings tested.

Keywords: off-policy learning, policy gradient, actor-critic, reinforcement learning

1. Introduction

Policy gradient methods are a general class of algorithms for learning optimal policies for both the on and off-policy settings. In policy gradient methods, a parameterized policy is improved using gradient ascent (Williams, 1992), with seminal work in actor-critic algorithms (Witten, 1977; Barto et al., 1983) and many techniques since proposed to reduce variance of the estimates of this gradient (Konda and Tsitsiklis, 2000; Weaver and Tao, 2001; Greensmith et al., 2004; Peters et al., 2005; Bhatnagar et al., 2007, 2009; Grondman et al., 2012; Gu et al., 2017b). These algorithms rely on a fundamental theoretical result: the *policy gradient theorem.* This theorem (Sutton et al., 1999a; Marbach and Tsitsiklis, 2001) simplifies estimation of the gradient, which would otherwise require difficult-to-estimate gradients with respect to the stationary distribution of the policy and potentially of the action-values.

These gradients can be sampled *on-policy* or *off-policy*. On-policy methods are limited to learning about the agent's current policy: the policy must be executed to obtain a sample of the gradient. Conversely, in off-policy learning the agent can learn about many policies that are different from the policy being executed. Sampling these gradients is most straightforward in the on-policy setting, and so most policy gradient methods are on-policy.

Off-policy methods, however, are critical in an online setting, where an agent generates a single stream of interaction with its environment. Off-policy methods can learn from data generated by older versions of a policy, known as experience replay, a critical factor in the recent success of deep reinforcement learning (Lin, 1992; Mnih et al., 2015; Schaul et al., 2016). They also enable learning from other forms of suboptimal data, including data generated by human demonstration, non-learning controllers, and even random behaviour. Off-policy methods also enable learning about the optimal policy while executing an exploratory policy (Watkins and Dayan, 1992), thereby addressing the exploration-exploitation tradeoff. Finally, off-policy methods allow an agent to learn about many different policies at once, forming the basis for a predictive understanding of an agent's environment (Sutton et al., 2011; White, 2015) and enabling the learning of options (Sutton et al., 1999b; Precup, 2000; Klissarov and Precup, 2021). With options, an agent can determine optimal (short) behaviours from its current state.

Off-policy policy gradient methods have been developed, particularly in recent years where the need for data efficiency and decorrelated samples in deep reinforcement learning require the use of experience replay and so off-policy learning. This work began with the Off-Policy Actor-Critic algorithm (OffPAC) (Degris et al., 2012a), where an off-policy policy gradient theorem was provided that parallels the on-policy policy gradient theorem, but only for tabular policy representations.[1] This motivated further development, including a recent actor-critic algorithm proven to converge when the critic uses linear function approximation (Maei, 2018), as well as several methods using the approximate off-policy gradient such as Deterministic Policy Gradient (DPG) (Silver et al., 2014; Lillicrap et al., 2015), Actor-Critic with Experience Replay (ACER) (Wang et al., 2016), and Interpolated Policy Gradient (IPG) (Gu et al., 2017a). However, it remained an open question whether the foundational theorem that underlies these algorithms can be generalized beyond tabular representations.

This question was resolved with the development of an off-policy policy gradient theorem, for general policy parametrization (Imani et al., 2018). The key insight is that the gradient can be simplified if the gradient in each state is weighted with an emphatic weighting. This result, combined with previous methods for incrementally estimating emphatic weightings (Yu, 2015; Sutton et al., 2016), allowed for the design of a new off-policy actor-critic algorithm, called Actor-Critic with Emphatic Weightings (ACE). Afterwards, an algorithm was proposed that directly estimates the emphatic weightings using a gradient temporal difference update, with an associated proof of convergence using the standard two-timescale analysis (Zhang et al., 2020).

However, ACE and the underlying off-policy policy gradient theorem do not obviously resolve the dilemma of computing the off-policy gradient, because ACE—and previously OffPAC—were introduced under what is called the *excursions* objective. This objective weights states by the visitation distribution of the behaviour policy. This is sensible in the

---

1. See B. Errata in Degris et al. (2012b) that the theorem only applies to tabular policy representations.

parallel off-policy setting, where many policies are learned in parallel from one stream of experience. The agent reasons about the outcomes of those policies when taking excursions from its current behaviour. In the episodic setting, however, weighting the states by the visitation distribution of the behaviour policy is not appropriate. Instead, an ideal episodic objective should be weighted by the start-state distribution, and this objective should be optimized from off-policy data without changing this weighting.

Potentially because of this mismatch, most recent methods have pursued strategies that directly correct state-weightings,[2] to obtain gradients of either the episodic objective (Liu et al., 2020b) or a finite-horizon objective (Kallus and Uehara, 2020). This approach involves estimating the distribution of state visitation under the behaviour policy and the discounted state visitation under the target policy to obtain an importance sampling ratio to reweight the updates. One disadvantage to this approach is that the state visitation distribution must be estimated, and these estimates can be extremely biased—both by the initialization and by the limitations of the parametric function class used for the estimates.

In this work, we revisit estimating the off-policy gradient with emphatic weightings. First, we propose a unifying objective for the episodic setting and the excursions objective. This objective facilitates the design of a single, unifying off-policy algorithm pertinent to both settings. Then, we extend the off-policy policy gradient theorem both to this objective and to objectives that incorporate entropy regularization. We further show that the on-policy episodic objective can also be seen as a special case, through the use of interest functions to specify interest in the set of start states. We then highlight the difference to previous off-policy approaches including OffPAC and DPG, which use a biased gradient that we call the semi-gradient because it omits a component of the gradient. We prove that in a simple three state counterexample, with two states aliased, the stationary point under semi-gradients is suboptimal, unlike ACE which converges to the optimal solution.

Though such state-aliasing is not pathological, it does seem to contradict the success of methods built on OffPAC in the literature. We show that, under a condition called the *strong growth condition* (Schmidt and Roux, 2013), which is obtained if there is sufficient representational capacity, the semi-gradient solution is equivalent to that of ACE. Further, we highlight that even outside this setting, in some cases the semi-gradient can be seen as using the sound update underlying ACE, but with a different state weighting. These two insights help explain why OffPAC has been surprisingly effective and suggest promising directions for a lower-variance version of ACE.

Finally, we discuss several improvements to the algorithm, including using direct estimates of the emphatic weightings to reduce variance, incorporating experience replay, and balancing between the higher-variance full gradient update and the lower-variance semi-gradient update. We empirically investigate variants of ACE in several benchmark environments, and find that those with direct estimates of the emphatic weighting consistently perform as well as or better than OffPAC.

Remark on Contributions: This paper builds on our conference paper (Imani et al., 2018). The conference paper presented a policy gradient theorem for one off-policy objective

---

2. A notable exception is an approach that uses a nonparametric Bellman equation—essentially using nonparametric estimates of the environment model (Tosatto et al., 2020). This approach allows direct estimation of the gradient by taking derivatives through this nonparametric Bellman equation.

function, and a counterexample showing that OffPAC and DPG can converge to the wrong solution, with experiments limited to the simple three-state counterexample.

An important limitation of that work is that it only applies to the excursions objective, which does not obviously relate to the standard policy gradient objective. Further, the work primarily investigated the true gradient, and did not provide much insight into how to practically use the algorithm. This paper builds on the conference paper in several substantive ways.

1. We first clarify the confusion surrounding objectives by providing a more general objective that includes both the standard objective and the excursions objective. We provide all the derivations for this more general objective (their Theorems 1 and 2 and Proposition 1, which are our Theorems 2 and 3 and Proposition 6). The result is nearly identical, but reiterated with the more general definition, and slightly different terminology. We also highlight that we can recover a sound on-policy algorithm using interest functions.

2. We prove that the semi-gradient from OffPAC has suboptimal solutions on the counterexample. The conference paper only showed this empirically. To obtain this result, we needed to extend the derivation to include entropy regularization, so that the optimal policy is stochastic and its parameters are bounded. Then we proved that the stationary points for OffPAC on the counterexample do not include the optimal point.

3. We provide insight into why semi-gradient methods can perform well, by showing that in some cases the weighting does not affect the solution and that locally we can characterize the semi-gradient as a gradient update with a different state weighting.

4. We provide several algorithmic improvements to ACE, particularly by directly estimating the emphatic weightings to give a lower variance update. In our experiments, this algorithm outperforms or performs comparably to all others, including OffPAC.

5. We conduct a more thorough empirical investigation of ACE in more settings, including two classic benchmarks and a new partially-observable image-based domain. The experiments are aimed at deeply investigating the role of different components, including (a) the objective used for learning versus evaluation, (b) the algorithm for estimating the emphatic weightings, and (c) the algorithm used for the critic.

2. Problem Formulation

Throughout the paper, we use bolded symbols to represent vectors and matrices, and uppercase italicized symbols to represent random variables. We consider a Markov decision process $(\mathcal{S}, \mathcal{A}, P, r)$, where $\mathcal{S}$ denotes the set of states, $\mathcal{A}$ denotes the set of actions, $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ denotes the one-step state transition dynamics, and $r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ denotes the transition-based reward function. At each timestep $t = 1, 2, \ldots$, the agent selects an action A_t according to its behaviour policy μ, where $\mu : \mathcal{S} \to \Delta(\mathcal{A})$. The environment responds by transitioning into a new state S_{t+1} according to P, and emits a scalar reward $R_{t+1} = r(S_t, A_t, S_{t+1})$.

The discounted sum of future rewards given actions are selected according to some target policy π is called the return, and defined as:

$$G_t \stackrel{\text{def}}{=} R_{t+1} + \gamma_{t+1} R_{t+2} + \gamma_{t+1} \gamma_{t+2} R_{t+3} + \ldots \tag{1}$$

$$= R_{t+1} + \gamma_{t+1} G_{t+1}.$$

We use transition-based discounting $\gamma : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$, as it facilitates specifying different tasks (termination) on top of the same MDP (White, 2017). This generality is useful in our setting, where we may be interested in learning multiple (option) policies, off-policy. The state value function for π and γ is defined as

$$v_\pi(s) \stackrel{\text{def}}{=} \mathbb{E}_\pi[G_t | S_t = s] \quad \forall s \in \mathcal{S}, \tag{2}$$

where under discrete states and actions this corresponds to

$$v_\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \sum_{s' \in \mathcal{S}} \mathrm{P}(s'|s, a)[r(s, a, s') + \gamma(s, a, s') v_\pi(s')] \quad \forall s \in \mathcal{S}$$

and under continuous states and actions this corresponds to

$$v_\pi(s) = \int_{\mathcal{A}} \pi(a|s) \int_{\mathcal{S}} \mathrm{P}(s'|s, a)[r(s, a, s') + \gamma(s, a, s') v_\pi(s')] \, \mathrm{d}a \, \mathrm{d}s' \quad \forall s \in \mathcal{S}.$$

When possible, we will opt for the more generic expectation notation, to address both the finite and continuous cases.

In off-policy control, the agent's goal is to learn a target policy π while following the behaviour policy μ. The target policy $\pi_{\boldsymbol{\theta}}$ is a differentiable function of a weight vector $\boldsymbol{\theta} \in \mathbb{R}^{n_{\boldsymbol{\theta}}}, n_{\boldsymbol{\theta}} \in \mathbb{N}$. The goal of the agent is to learn a policy that maximizes total reward. Depending on the setting, however, this goal is specified with different objectives, as we discuss in the next section.

Throughout, we assume that the limiting distribution $d_\mu(s)$ under μ exists, where

$$d_\mu(s) \stackrel{\text{def}}{=} \lim_{t \to \infty} \mathrm{P}(S_t = s | s_0, \mu) \tag{3}$$

and $\mathrm{P}(S_t = s | s_0, \mu)$ is the probability—or density—that $S_t = s$ when starting in state s_0 and executing μ. Similarly, d_π denotes the limiting distribution[3] of states under π. Note that this limiting distribution exists in either the episodic or continuing settings (White, 2017; Huang, 2020). One way to see this is that in the episodic setting under transition-based discounting, there are no explicit terminal states. Rather, the agent transitions between the state right before termination, back to a start state, as it would for a continuing problem. The transition-based discount truncates returns, to specify that the goal of the agent from each state is to maximize return for the episode.

3. The Weighted-Excursions Objective for Off-Policy Control

In this section we introduce the weighted-excursions objective that unifies two objectives commonly considered in off-policy control. The objective encompasses the standard on-policy episodic objective as well as the excursions objective that allows option policies to be

---

3. This term is sometimes overloaded to mean the discounted state visitation starting from a single start state s_0: $\sum_{t=0}^{\infty} \gamma^t \mathrm{P}(S_t = s | s_0, \pi)$. This overloading comes from the definition given in the original policy gradient theorem (Sutton et al., 1999a). We only use it to mean state visitation (limiting distribution).

learned off-policy with different termination conditions. Throughout, we will define both the finite state and continuous state versions.

The standard episodic objective is

$$J_0(\boldsymbol{\theta}) \stackrel{\text{def}}{=} \sum_{s \in \mathcal{S}} d_0(s) v_{\pi_{\boldsymbol{\theta}}}(s) \qquad \text{or} \qquad J_0(\boldsymbol{\theta}) \stackrel{\text{def}}{=} \int_{\mathcal{S}} d_0(s) v_{\pi_{\boldsymbol{\theta}}}(s) \, \mathrm{d}s \qquad (4)$$

where $d_0 \in \Delta(\mathcal{S})$ is the start state distribution. This objective can be optimized in either the on-policy or off-policy settings, with different updates to account for the fact that data is gathered according to μ in the off-policy setting.

Another objective that has been considered in the off-policy setting is the *excursions objective* (Degris et al., 2012b)

$$J_{\text{exc}}(\boldsymbol{\theta}) \stackrel{\text{def}}{=} \sum_{s \in \mathcal{S}} d_{\mu}(s) v_{\pi_{\boldsymbol{\theta}}}(s), \qquad (5)$$

where the state weighting is determined by the behaviour policy's limiting state distribution d_{μ}. This objective assumes that the target policy $\pi_{\boldsymbol{\theta}}$ will be executed as an excursion from the current behaviour—or that the agent will use the target policy in planning under the current state visitation. In the options framework (Sutton et al., 1999b), for example, the agent learns the optimal policies for a variety of different options, each with different termination conditions. The behaviour policy itself μ could be learning in a continuing environment, even though the objective for the option policy π_{θ} is episodic, in that it has termination conditions.

The excursions objective, however, is not necessarily appropriate for learning an alternative optimal policy. For example, consider an episodic setting where a reasonable behaviour policy is currently being used, and we would like to learn an improved policy. The improved policy will be executed from the set of start states, not from the visitation distribution under the behaviour policy. In this setting, we would like to optimize the standard episodic objective, using off-policy data generated by the current behaviour policy.

These two settings can be unified by considering the *weighted-excursions* objective

$$J_{\mu}(\boldsymbol{\theta}) \stackrel{\text{def}}{=} \sum_{s \in \mathcal{S}} d_{\mu}(s) i(s) v_{\pi_{\boldsymbol{\theta}}}(s) \qquad \text{or} \qquad J_{\mu}(\boldsymbol{\theta}) \stackrel{\text{def}}{=} \int_{\mathcal{S}} d_{\mu}(s) i(s) v_{\pi_{\boldsymbol{\theta}}}(s) \, \mathrm{d}s, \qquad (6)$$

where the weighting $i : \mathcal{S} \to [0, \infty)$ represents the interest in a state (Sutton et al., 2016). This objective reduces to the episodic (start-state) formulation by setting $i(s) = d_0(s)/d_{\mu}(s)$. This choice is not just hypothetical; we will show how our algorithm designed for the excursions objective can also be used for the start-state formulation by using an easy-to-compute setting of the interest $i(s)$ for each $s \in \mathcal{S}$. This generalization goes beyond unifying the previous two objectives, and also naturally allows us to reweight states based on their importance for the target policy. For example, when learning an option, the interest function could be set to 1 in states from which the policy will be executed (i.e., the initiation set), and zero elsewhere.

More generally, we could consider the generic objective $\sum_{s \in \mathcal{S}} d(s) v_{\pi_{\boldsymbol{\theta}}}(s)$ for any state weighting d. Then the weighted-excursions objective is simply an instance of this more generic objective, with $d = d_{\mu}(s) i(s)$. The reason that we opt for the weighted-excursions

6

objective is because it is sufficiently general to incorporate our two settings of interest but sufficiently restricted to facilitate development of a practical off-policy algorithm. In fact, the idea of interest comes from the same work on emphatic weightings (Sutton et al., 2016) on which we build our algorithm. This work also provides insight into how to easily incorporate such interest functions in the off-policy setting.

Remark: In this work, we focus on episodic objectives—those with termination—but it is worthwhile to contrast this to the objective used for continuing problems. For continuing problems, there is a compelling case for the average reward objective

$$J_{\text{ave}}(\boldsymbol{\theta}) \stackrel{\text{def}}{=} \sum_{s \in \mathcal{S}} d_{\pi_{\boldsymbol{\theta}}}(s) r_{\pi_{\boldsymbol{\theta}}}(s) \qquad \text{or} \qquad J_{\text{ave}}(\boldsymbol{\theta}) \stackrel{\text{def}}{=} \int_{\mathcal{S}} d_{\pi_{\boldsymbol{\theta}}}(s) r_{\pi_{\boldsymbol{\theta}}}(s) \, \mathrm{d}s \qquad (7)$$

for $r_\pi(s) = \mathbb{E}_\pi[r(s, A, S')]$. Optimizing J_{ave} is equivalent to optimizing $\sum_{s \in \mathcal{S}} d_\pi(s) v_\pi(s)$ for a constant $0 \le \gamma < 1$ (Sutton and Barto, 2018). Intuitively, the agent seeks to maximize state values while shifting its state distribution towards high-valued states.

A sharp distinction has been previously made between *alternative life* objectives and excursions objectives in policy evaluation (Patterson et al., 2022). In that work, alternative life objectives use the state-weighting d_π that specifies: learn the best value function as if data had been gathered under the target policy π. Such a weighting is obtained by using prior corrections—namely reweighting entire trajectories with products of importance sampling ratios (Precup et al., 2000; Meuleau et al., 2000)—or estimating d_π/d_μ (Hallak and Mannor, 2017; Liu et al., 2018; Gelada and Bellemare, 2019). The excursions objective, on the other hand, uses a weighting of d_μ. This distinction is not clearly relevant to control. Instead, typically *on-policy* average reward algorithms are developed for the continuing objective—which has weighting d_π. The few off-policy algorithms designed for the continuing, average reward setting directly estimate d_π and d_μ (Liu et al., 2019). The episodic objective for control does not include d_π, and so is different from the alternative life objective in policy evaluation.

In the remainder of this paper we develop an algorithm for optimizing the weighted-excursions objective in Equation (6) from off-policy data.

4. An Off-Policy Policy Gradient Theorem using Emphatic Weightings

In this section, we introduce the off-policy policy gradient theorem for the discrete and continuous settings as well as for deterministic policies. We additionally compare this gradient to the gradient underlying OffPAC, and provide a generalized gradient that interpolates between this gradient—called the semi-gradient—and the true gradient.

4.1 The Off-Policy Policy Gradient Theorem

The policy gradient theorem with function approximation in the on-policy setting was a seminal result (Sutton et al., 1999a, Theorem 1). The first attempt to extend the policy gradient theorem to the off-policy excursion objective was limited to the setting where the policy is tabular (Degris et al., 2012b, Theorem 2).[4] In this section, we show that the policy

---

4. Note that the statement in the paper is stronger, but in an errata published by the authors, they highlight an error in the proof. Consequently, the result is only correct for the tabular setting.

gradient theorem does hold in the off-policy setting for the weighted excursions objective, for arbitrary smooth policy parametrizations. The resulting gradient resembles the standard policy gradient, but with emphatic weightings to reweight the states.

Definition 1 (Emphatic Weighting) *The emphatic weighting* $m : \mathcal{S} \to [0, \infty)$ *for a behaviour policy* μ *and target policy* π *can be recursively defined as*

$$m(s') \overset{\text{def}}{=} d_\mu(s')i(s') + \mathbb{E}[\gamma(S, A, S')m(S) \mid S' = s'], \tag{8}$$

where the distribution over S, A *from next state* S' *is under* P *and* π.

Under finite states and actions, this expectation is $\sum_{s \in \mathcal{S}} \sum_{a \in \mathcal{A}} \pi(a|s)\mathrm{P}(s'|s, a)\gamma(s, a, s')m(s)$. Under continuous states and actions, it is $\int_\mathcal{S} \int_\mathcal{A} \pi(a|s)\mathrm{P}(s'|s, a)\gamma(s, a, s')m(s) \, \mathrm{d}a \, \mathrm{d}s$. If a deterministic policy $\pi : \mathcal{S} \to \mathcal{A}$ is used, then it is $\int_\mathcal{S} \mathrm{P}(s'|s, \pi(s))\gamma(s, \pi(s), s')m(s) \, \mathrm{d}s$.

Notice that these emphatic weightings involve states leading into s'; bootstrapping is back-in-time, rather than forward. The emphatic weighting reflects the relative importance of a state s', based on its own interest and weighting—the first term $d_\mu(s')i(s')$—as well as the discounted importance of states that lead into s'. For value estimation, this is sensible because it reflects how much the agent bootstraps off of the values in s', and hence how much it relies on accurate estimates in s'. For policy gradients, we have a related role: the importance of a state is due both to its own value, as well as how much other states that lead into it depend on the value obtained from that state.

Theorem 2 (Off-Policy Policy Gradient Theorem) *For finite states and actions,*

$$\frac{\partial J_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \sum_{s \in \mathcal{S}} m(s) \sum_a \frac{\partial \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, a) = \sum_{s \in \mathcal{S}} m(s)g(s, \boldsymbol{\theta}) \tag{9}$$

where

$$g(s, \boldsymbol{\theta}) \overset{\text{def}}{=} \mathbb{E}_{\pi_{\boldsymbol{\theta}}} \left[\frac{\partial \log \pi(A|S; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(S, A) \; \middle| \; S = s \right] \tag{10}$$

is the gradient for a given state, rewritten using the log-likelihood ratio. For continuous states, and either continuous or discrete actions,

$$\frac{\partial J_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \int_\mathcal{S} m(s)g(s, \boldsymbol{\theta}) \, \mathrm{d}s. \tag{9 for continuous states}$$

Proof We first start with the finite state setting. The proof relies on the vector form of the emphatic weighting

$$\mathbf{m}^\top \overset{\text{def}}{=} \mathbf{i}^\top (\mathbf{I} - \mathbf{P}_{\pi, \gamma})^{-1},$$

where the vector $\mathbf{i} \in \mathbb{R}^{|\mathcal{S}|}$ has entries $\mathbf{i}(s) \overset{\text{def}}{=} d_\mu(s)i(s)$ and $\mathbf{P}_{\pi, \gamma} \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{S}|}$ is the matrix with entries $\mathbf{P}_{\pi, \gamma}(s, s') \overset{\text{def}}{=} \sum_{a \in \mathcal{A}} \pi(a|s; \boldsymbol{\theta})\mathrm{P}(s'|s, a)\gamma(s, a, s')$. First notice that

$$\frac{\partial J_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \frac{\partial \sum_{s \in \mathcal{S}} \mathbf{i}(s)v_\pi(s)}{\partial \boldsymbol{\theta}} = \sum_{s \in \mathcal{S}} \mathbf{i}(s) \frac{\partial v_\pi(s)}{\partial \boldsymbol{\theta}}.$$

Therefore, to compute the gradient of J_μ, we need to compute the gradient of the value function with respect to the policy parameters. A recursive form of the gradient of the

value function can be derived, as we show below. Before starting, for simplicity of notation we will use

$$\mathbf{g}(s) = \sum_a \frac{\partial \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, a),$$

where $\mathbf{g} : \mathcal{S} \to \mathbb{R}^d$. Now let us compute the gradient of the value function:

$$
\begin{aligned}
\frac{\partial v_\pi(s)}{\partial \boldsymbol{\theta}} &= \frac{\partial}{\partial \boldsymbol{\theta}} \sum_a \pi(a|s; \boldsymbol{\theta}) q_\pi(s, a) \\
&= \sum_a \frac{\partial \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, a) + \sum_a \pi(a|s; \boldsymbol{\theta}) \frac{\partial q_\pi(s, a)}{\partial \boldsymbol{\theta}} \\
&= \mathbf{g}(s) + \sum_a \pi(a|s; \boldsymbol{\theta}) \frac{\partial \sum_{s'} \mathrm{P}(s'|s, a)(r(s, a, s') + \gamma(s, a, s') v_\pi(s'))}{\partial \boldsymbol{\theta}} \\
&= \mathbf{g}(s) + \sum_a \pi(a|s; \boldsymbol{\theta}) \sum_{s'} \mathrm{P}(s'|s, a) \gamma(s, a, s') \frac{\partial v_\pi(s')}{\partial \boldsymbol{\theta}}.
\end{aligned}
\tag{11}
$$

We can simplify this more easily using vector form. Let $\dot{\mathbf{v}}_\pi \in \mathbb{R}^{|\mathcal{S}| \times d}$ be the matrix of gradients (with respect to the policy parameters $\boldsymbol{\theta}$) of v_π for each state s, and $\mathbf{G} \in \mathbb{R}^{|\mathcal{S}| \times d}$ the matrix where each row corresponding to state s is the vector $\mathbf{g}(s)$. Then

$$\dot{\mathbf{v}}_\pi = \mathbf{G} + \mathbf{P}_{\pi, \gamma} \dot{\mathbf{v}}_\pi \qquad \implies \quad \dot{\mathbf{v}}_\pi = (\mathbf{I} - \mathbf{P}_{\pi, \gamma})^{-1} \mathbf{G}. \tag{12}$$

Therefore, we obtain

$$
\begin{aligned}
\sum_{s \in \mathcal{S}} \mathbf{i}(s) \frac{\partial v_\pi(s)}{\partial \boldsymbol{\theta}} = \mathbf{i}^\top \dot{\mathbf{v}}_\pi &= \mathbf{i}^\top (\mathbf{I} - \mathbf{P}_{\pi, \gamma})^{-1} \mathbf{G} \\
&= \mathbf{m}^\top \mathbf{G} \\
&= \sum_{s \in \mathcal{S}} m(s) \sum_a \frac{\partial \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, a).
\end{aligned}
$$

The proof is similar for the continuous case; we include it in Appendix B. ∎

We additionally prove the deterministic policy gradient objective, for a deterministic policy, $\pi : \mathcal{S} \to \mathcal{A}$. The objective remains the same, but the space of possible policies is constrained, resulting in a slightly different gradient.

Theorem 3 (Deterministic Off-policy Policy Gradient Theorem)

$$\frac{\partial J_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \int_{\mathcal{S}} m(s) \frac{\partial \pi(s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \left. \frac{\partial q_\pi(s, a)}{\partial a} \right|_{a = \pi(s; \boldsymbol{\theta})} \mathrm{d}s \tag{13}$$

where

$$m(s') = d_\mu(s') i(s') + \int_{\mathcal{S}} \mathrm{P}(s'|s, \pi(s; \boldsymbol{\theta})) \gamma(s, \pi(s; \boldsymbol{\theta}), s') m(s) \, \mathrm{d}s.$$

The proof is presented in Appendix A.

4.2 Connection to OffPAC and the Semi-gradient

The off-policy policy gradient above contrasts with the one used by OffPAC

$$\sum_{s \in \mathcal{S}} d_\mu(s) \mathbb{E}_{\pi_{\boldsymbol{\theta}}} \left[\frac{\partial \log \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(S, A) \mid S = s \right]. \tag{14}$$

The only difference is in the state weighting d_μ. This small difference, however, is key for getting the correct gradient. In fact, the OffPAC gradient can be considered a *semi-gradient*, because it drops a part of the gradient when using the product rule:

$$\begin{aligned} \nabla_{\boldsymbol{\theta}} J_{\mathrm{exc}}(\boldsymbol{\theta}) &= \sum_{s \in \mathcal{S}} d_\mu(s) \nabla_{\boldsymbol{\theta}} \sum_a \pi_{\boldsymbol{\theta}}(a|s) q_\pi(s, a) \\ &= \sum_{s \in \mathcal{S}} d_\mu(s) \sum_a \left[q_\pi(s, a) \nabla_{\boldsymbol{\theta}} \pi_{\boldsymbol{\theta}}(a|s) + \pi_{\boldsymbol{\theta}}(a|s) \nabla_{\boldsymbol{\theta}} q_\pi(s, a) \right]. \end{aligned}$$

The second term with $\nabla_{\boldsymbol{\theta}} q_\pi(s, a)$ is not obviously easy to compute, because it requires reasoning about how changes to the policy parameters affect the action-values. OffPAC opted to drop this term, and justify why this approximation was acceptable. The resulting semi-gradient has proven to be surprisingly effective, though we provide several examples in this work where it performs poorly.

Note that using this correct weighting is only critical due to state aliasing. If the policy representation is tabular, for example, the weighting on the gradients does not affect the asymptotic solution as long as it is non-zero.[5] The gradient $\sum_{s \in \mathcal{S}} m(s) g(\boldsymbol{\theta}, s) = 0$ if and only if $g(\boldsymbol{\theta}, s) = 0$, because the gradient vector has an independent entry for each state. Therefore, even if the state weighting in the gradient is changed to some other weighting d, then we still get $\sum_{s \in \mathcal{S}} d(s) g(\boldsymbol{\theta}, s) = 0$. This is the reason that the semi-gradient underlying OffPAC converges to a stationary point of this objective in the tabular setting. More generally, for any sufficiently powerful function approximator that can obtain $g(\boldsymbol{\theta}, s) = 0$ for all states, this result holds. In optimization, this property has been termed the *strong growth condition* (Schmidt and Roux, 2013). We state this simple result for the irrelevance of the weighting formally in the following proposition, both to highlight it and because to the best of our knowledge it has not been formally stated.

Proposition 4 (Irrelevance of Weighting under the Strong Growth Condition)
If $g(\boldsymbol{\theta}, s) = 0$ for all states, then $\boldsymbol{\theta}$ is a stationary point under any state weighting $d : \mathcal{S} \to \mathbb{R}$.

This result states that with a function approximator that can perfectly maximize value in each state, the choice of state weighting in the gradient computation is not relevant. Both the off-policy gradient with emphatic weighting, and the semi-gradient with weighting d_μ, can converge to this stationary point.

Alternatively, we can consider the more generic emphatic weighting that lets us sweep between the gradient and the semi-gradient. The emphatic weightings were introduced for temporal difference (TD) learning with eligibility traces (Mahmood et al., 2015). The weightings depend on the eligibility trace parameter, λ. For $\lambda = 1$, the weightings effectively

---

5. Note that the weighting on the gradients could still affect the rate of convergence (Laroche and Tachet des Combes, 2021).

become 1 everywhere—implicitly resulting in state weightings of d_μ—because there is no bootstrapping. For $\lambda = 0$, the weightings are exactly as defined above.

Definition 5 (Generalized Emphatic Weighting) *For $\eta \in [0, 1]$, the generalized emphatic weighting $m_\eta : \mathcal{S} \to [0, \infty)$ for a behaviour policy μ and target policy π is defined as*

$$m_\eta(s') \stackrel{\text{def}}{=} (1 - \eta)d_\mu(s')i(s') + \eta m(s') \tag{15}$$

For $\eta = 1$, $m_\eta(s') = m(s')$.

Note that in the original work, $m(s')$ was called the follow-on weighting, and $m_\eta(s')$ the emphatic weighting. Because m_η is a strict generalization of m, we simply call m an emphatic weighting as well.[6]

The parameter η can be interpreted as trading off bias and variance in the emphatic weightings. Notice that for $\eta = 0$, and assuming an interest of 1 everywhere, we get that $m_\eta(s) = d_\mu(s)$. Such a choice would give exactly the semi-gradient weighting. As we increase η, we interpolate between the semi-gradient and the full gradient. For $\eta = 1$, we get $m_\eta(s) = m(s)$ and so we have the full gradient. For $\eta = 0$, therefore, we obtain a biased gradient estimate, but the emphatic weightings themselves are easy to estimate—they are myopic estimates of interest—which could significantly reduce variance when estimating the gradient. Selecting η between 0 and 1 could provide a reasonable balance, obtaining a nearly unbiased gradient to enable convergence to a valid stationary point but potentially reducing some variability when estimating the emphatic weighting. We will develop our algorithm for the general emphatic weighting, as this will give us more flexibility in the weighting, and will also allow us to incorporate OffPAC as one extreme (biased) version of the approach.

4.3 Summary

In this section we proved the off-policy policy gradient theorem for the weighted-excursions objective, for both stochastic (Theorem 2) and deterministic (Theorem 3) policies. These gradients are similar to the standard policy gradient, but with states weighted by emphatic weightings. We contrasted this to OffPAC, the (unsound) algorithm on which many off-policy policy gradient algorithms are built, highlighting that the only distinction is in this state weighting. We argued that this state weighting does not always affect the solution (Proposition 4), potentially partially explaining why OffPAC often performs well—especially in deep RL with large neural networks. Finally, we show how we can use generalized emphatic weightings, with a parameter η (Definition 5), that allows us to interpolate between the sound, but harder-to-estimate emphatic weighting (at $\eta = 1$) and the unsound, but easy-to-obtain weighting used in OffPAC (at $\eta = 0$).

---

6. Note that the original emphatic weightings (Sutton et al., 2016) use $\lambda = 1 - \eta$. This is because their emphatic weightings are designed to balance bias introduced from using λ for estimating value functions: larger λ means the emphatic weighting plays less of a role. For this setting, we want larger η to correspond to the full emphatic weighting (the unbiased emphatic weighting), and smaller η to correspond to a more biased estimate, to better match the typical meaning of such trace parameters.

5. Actor-Critic with Emphatic Weightings

In this section, we develop an incremental actor-critic algorithm with emphatic weightings that uses the above off-policy policy gradient theorem. To perform a gradient ascent update on the policy parameters, the goal is to obtain a sample of the gradient in Equation 9. As discussed above, the only difference to the semi-gradient used by OffPAC, in Equation 14, is in the weighting of the states: the true gradient uses $m(s)$ and the semi-gradient uses $d_\mu(s)$. Therefore, we can use standard solutions developed for other actor-critic algorithms to obtain a sample of $g(s, \boldsymbol{\theta})$. The key difficulty is in estimating $m(s)$ to reweight this gradient.

5.1 Sampling the Gradient from a State

Before addressing this key difficulty, we provide a brief reminder on how to obtain a sample of $g(s, \boldsymbol{\theta})$, with more details in Appendix C. The simplest approach to compute the gradient for a given state is to use what is sometimes called the *all-actions* gradient $\sum_a \frac{\partial \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, a)$. To avoid summing over all actions, we instead obtain an unbiased sample of the gradient under the action taken by the behaviour policy $A \sim \mu(s, \cdot)$: $\rho(s, A; \boldsymbol{\theta}) \frac{\partial \log \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, A)$ where the importance sampling ratio $\rho(s, a; \boldsymbol{\theta}) \stackrel{\text{def}}{=} \frac{\pi(a|s;\boldsymbol{\theta})}{\mu(a|s)}$ corrects the distribution over actions. The estimated gradient has more variance when we only use a sampled action rather than all actions. The standard strategy to reduce this variance is to subtract a baseline, such as an estimate of the value function $v(s)$. The resulting update is

$$\rho(s, a; \boldsymbol{\theta}) \frac{\partial \log \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} [q_\pi(s, a) - v(s)].$$

In practice, it is difficult to obtain $q_\pi(s, a)$ to compute this gradient exactly. Instead, we obtain (a likely biased) sample of the advantage, $\delta_t \approx q_\pi(S_t, A_t) - v(s_t)$. An unbiased but high variance estimate uses a sample of the return G_t from (s_t, a_t), and uses $\delta_t = G_t - v(s_t)$. Then $\mathbb{E}[\delta_t | S_t = s, A_t = a] = q_\pi(s, a) - v(s)$. On the other extreme, we can directly approximate the action-values, $\hat{q}(s, a)$. In-between, we can use n-step returns, or λ-returns, to balance between using observed rewards and value estimates. In this work, we opt for the simplest choice: $n = 1$ with $\delta_t = R_{t+1} + \gamma_{t+1} v(S_{t+1}) - v(S_t)$.

Finally, we need to estimate the values themselves. Any value approximation algorithm can be used to estimate v, such as TD, gradient TD or even emphatic TD. Given we already compute the emphasis weighting, it is straightforward to use emphatic TD. We investigate both a gradient TD critic and an emphatic TD critic in the experiments.

5.2 Estimating the Gradient with Emphatic Weightings

The previous section outlined how to sample the gradient for a given state. We now need to ensure that these updates across states are weighted by the emphatic weighting. Intuitively, if we could estimate $m_\eta(s)$ and $d_\mu(s)$ for all s, then we could simply premultiply the update with $m_\eta(s)/d_\mu(s)$. However, estimating the state distribution is nontrivial, and actually unnecessary. In this section, we first explain how to obtain a Monte Carlo estimate of this pre-multiplier, and then how to directly estimate it with function approximation.

5.2.1 An Unbiased Estimate of the Gradient with Emphatic Weightings

We can rely on the original work introducing the emphatic weightings to obtain a Monte Carlo estimate with the update

$$M_t = (1 - \eta)i_t + \eta F_t \qquad\qquad F_t = \gamma_t \rho_{t-1} F_{t-1} + i_t \qquad\qquad (16)$$

where $F_0 = 0$ and i_t is the interest observed in S_t, with $i(s) = \mathbb{E}[i_t | S_t = s]$. Putting this together with the previous section, the gradient can be sampled by (1) sampling states according to d_μ, (2) taking actions according to μ, (3) obtaining an estimate δ_t of the advantage $q_\pi(s, a) - v(s)$, and then (4) weighting the update with M_t, to get

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \rho_t M_t \frac{\partial \log \pi(A_t | S_t; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \delta_t.$$

Note that the all-actions gradient update with this emphatic weighting would be

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha M_t \sum_{b \in \mathcal{A}} \pi(b | S_t; \boldsymbol{\theta}) \frac{\partial \log \pi(b | S_t; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(S_t, b),$$

but as before we will assume that we use one sampled action. We prove that our update is an unbiased estimate of the gradient for a fixed policy in Proposition 6. We assume access to an unbiased estimate δ_t of the advantage for this result, though in practice δ_t is likely to have some bias due to using approximate values.

Proposition 6 *For a fixed policy π, with the conditions on the MDP from (Yu, 2015) and assuming unbiased δ_t, namely that $\mathbb{E}[\delta_t | S_t = s, A_t = a] = q_\pi(s, a) - v(s)$,*

$$\mathbb{E}_{S_t \sim d_\mu, A_t \sim \mu}[\rho_t M_t \delta_t \nabla_{\boldsymbol{\theta}} \log \pi(A_t | S_t; \boldsymbol{\theta})] = \sum_{s \in \mathcal{S}} m_\eta(s) \sum_a \frac{\partial \pi(a | s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, a).$$

Proof Emphatic traces M_t have been shown to provide an unbiased estimate of the true emphatic weighting for policy evaluation in Emphatic TD. We use the emphatic weighting differently, but can rely on the proof from (Sutton et al., 2016) to ensure that (a) $d_\mu(s) \lim_{t \to \infty} \mathbb{E}_\mu[M_t | S_t = s] = m_\eta(s)$. Note also that, given S_t, the next action does not impact the expectation: (b) $\mathbb{E}_\mu[M_t | S_t = s] = \mathbb{E}_\mu[M_t | S_t = s, A_t]$. Using these equalities, we obtain

$$\mathbb{E}_{S_t \sim d_\mu, A_t \sim \mu}[\rho_t M_t \delta_t \nabla_{\boldsymbol{\theta}} \log \pi(A_t | S_t; \boldsymbol{\theta})]$$

$$= \sum_{s \in \mathcal{S}} d_\mu(s) \mathbb{E}_\mu[\rho_t M_t \delta_t \nabla_{\boldsymbol{\theta}} \log \pi(A_t | S_t; \boldsymbol{\theta}) | S_t = s]$$

$$= \sum_{s \in \mathcal{S}} d_\mu(s) \mathbb{E}_\mu \Big[\mathbb{E}_\mu[\rho_t M_t \delta_t \nabla_{\boldsymbol{\theta}} \log \pi(A_t | S_t; \boldsymbol{\theta}) | S_t = s, A_t] \Big] \qquad \triangleright \text{law of total expectation}$$

$$= \sum_{s \in \mathcal{S}} d_\mu(s) \mathbb{E}_\mu \Big[\mathbb{E}_\mu[M_t | S_t = s, A_t] \ \mathbb{E}_\mu[\rho_t \delta_t \nabla_{\boldsymbol{\theta}} \log \pi(A_t | S_t; \boldsymbol{\theta}) | S_t = s, A_t] \Big]$$

$$= \sum_{s \in \mathcal{S}} d_\mu(s) \mathbb{E}_\mu[M_t | S_t = s] \mathbb{E}_\mu \Big[\mathbb{E}_\mu[\rho_t \delta_t \nabla_{\boldsymbol{\theta}} \log \pi(A_t | S_t; \boldsymbol{\theta}) | S_t = s, A_t] \Big] \qquad \triangleright \text{ using (b)}$$

$$= \sum_{s \in \mathcal{S}} m_\eta(s) \mathbb{E}_\mu[\rho_t \delta_t \nabla_{\boldsymbol{\theta}} \log \pi(A_t|S_t; \boldsymbol{\theta})|S_t = s] \qquad \triangleright \text{ using (a)}$$

$$= \sum_{s \in \mathcal{S}} m_\eta(s) \sum_a \frac{\partial \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, a),$$

where the last line follows from the fact that

$$\mathbb{E}_\mu[\rho_t \delta_t \nabla_{\boldsymbol{\theta}} \log \pi(A_t|S_t; \boldsymbol{\theta})|S_t = s]$$

$$= \sum_a \mu(a|s)\rho(s, a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s; \boldsymbol{\theta}) \frac{\partial \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \mathbb{E}_\mu[\delta_t|S_t = s, A_t = a]$$

$$= \sum_a \mu(a|s)\rho(s, a) \nabla_{\boldsymbol{\theta}} \log \pi(a|s; \boldsymbol{\theta}) \frac{\partial \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} [q_\pi(s, a) - v(s)] \qquad \triangleright \text{ by assumption}$$

$$= \sum_a \frac{\partial \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, a). \qquad \blacksquare$$

5.2.2 A BIASED ESTIMATE OF THE GRADIENT WITH EMPHATIC WEIGHTINGS

For a fixed policy, the emphatic trace provides an unbiased way to reweight the gradient. However, this is no longer true when the target policy is updated online in an actor-critic algorithm like ACE; the emphatic trace will contain importance sampling ratios for older versions of the policy. If the target policy changes substantially during the learning process—as one would hope—the older importance sampling ratios could bias the emphatic trace's estimates. On the other hand, a constant discount rate $\gamma < 1$ would give exponentially less weight to older importance sampling ratios in the emphatic trace, potentially mitigating the bias. We empirically investigate the impact of this type of bias in section 9.3.

As a Monte Carlo estimator of the emphatic weightings, the emphatic trace can also yield high-variance estimates. Furthermore, the product of importance sampling ratios in the emphatic trace can lead to even higher-variance estimates, which can slow learning (Precup et al., 2001; Liu et al., 2020a; Ghiassian et al., 2018). As such, several algorithms have been proposed that use parametrized functions to estimate the factors required for reweighting the updates (Hallak and Mannor, 2017; Gelada and Bellemare, 2019; Liu et al., 2018; Zhang et al., 2019). We can similarly approximate the emphatic weighting directly.

For our setting, this involves estimating $\mathbb{E}_\mu[M_t|S_t = s] = m_\eta(s)/d_\mu(s)$. Because M_t is a function of F_t, we directly approximate $\mathbb{E}_\mu[F_t|S_t = s]$ by learning a parametrized function $f_\phi(s)$ and use it in place of F_t to compute M_t. Notice that the resulting $f_\phi(s) \approx \mathbb{E}_\mu[F_t|S_t = s] = m(s)/d_\mu(s)$, and so $(1 - \eta)i(s) + \eta f_\phi(s) \approx \mathbb{E}_\mu[M_t|S_t = s] = m_\eta(s)/d_\mu(s)$. Because we sample $s \sim d_\mu$, weighting the update with an estimate of $m_\eta(s)/d_\mu(s)$ is effectively an importance sampling ratio that ensures the updates are weighted according to $m_\eta(s)$.

The direct approximation relies on the recursive equation for the emphatic weighting that allows for a temporal-difference update. Unlike the usual TD update, this update bootstraps off the estimate from the previous step rather than the next step, because the emphatic weightings accumulate interest back in time, leading into the state. Recall the

recursive equation

$$m(s') = d_\mu(s')i(s') + \sum_{\mathcal{S},\mathcal{A}} \pi(a|s;\boldsymbol{\theta})p(s'|s,a)\gamma(s,a,s')m(s).$$

This equation is a Bellman equation, that specifies the emphatic values for state s' in terms of the immediate interest and the states leading into s. We approximate $f_\phi(s) \approx m(s)/d_\mu(s)$, rather than $m(s)$. The recursive formula for an $f(s)$ that equals $m(s)/d_\mu(s)$ is

$$f(s') = i(s') + \tfrac{1}{d_\mu(s')} \sum_{\mathcal{S},\mathcal{A}} \pi(a|s;\boldsymbol{\theta})p(s'|s,a)\gamma(s,a,s')d_\mu(s)f(s'),$$

where $i_t + \rho_{t-1}\gamma_t f_\phi(S_{t-1})$ is a sample of this target, as we show in the next proposition.

Proposition 7 *Under the conditions stated by Sutton et al. (2016),*

$$\lim_{t\to\infty} \mathbb{E}_\mu[i_t + \rho_{t-1}\gamma_t f_\phi(S_{t-1})|S_t = s'] = i(s') + \tfrac{1}{d_\mu(s')} \sum_{s,a} \pi(a|s;\boldsymbol{\theta})p(s'|s,a)\gamma(s,a,s')d_\mu(s)f_\phi(s).$$

Proof The proof follows the strategy showing the unbiasedness of F_t in Sutton et al. (2016).

$$\begin{aligned}
&\lim_{t\to\infty} \mathbb{E}_\mu[i_t + \rho_{t-1}\gamma_t f_\phi(S_{t-1})|S_t = s'] \\
&= i(s') + \lim_{t\to\infty} \mathbb{E}_\mu[\rho_{t-1}\gamma_t f_\phi(S_{t-1})|S_t = s'] \\
&= i(s') + \sum_{s,a} \lim_{t\to\infty} \Pr\{S_{t-1} = s, A_{t-1} = a|S_t = s'\} \frac{\pi(a|s;\boldsymbol{\theta})}{\mu(a|s)}\gamma(s,a,s')f_\phi(s) \\
&= i(s') + \sum_{s,a} \frac{d_\mu(s)\mu(a|s)p(s'|s,a)}{d_\mu(s')} \frac{\pi(a|s;\boldsymbol{\theta})}{\mu(a|s)}\gamma(s,a,s')f_\phi(s) \qquad \triangleright \text{ Bayes rule} \\
&= i(s') + \tfrac{1}{d_\mu(s')} \sum_{s,a} \pi(a|s;\boldsymbol{\theta})p(s'|s,a)\gamma(s,a,s')d_\mu(s)f_\phi(s)
\end{aligned}$$

∎

The utility of this result is that it provides a way to sample the target for this Bellman equation. The following is a sample update whose target is equal to the above in expectation, when sampling $s \sim d_\mu$:

$$\phi \leftarrow \phi + \beta_t\big[i_t + \rho_{t-1}\gamma_t f_\phi(S_{t-1}) - f_\phi(S_t)\big]\nabla_\phi f_\phi(S_t), \tag{17}$$

where β_t is a step-size. The update aims to minimize the difference between the current estimate of the emphatic weightings and the target, using a standard semi-gradient TD update. We could alternatively use a gradient TD (Sutton et al., 2009; Sutton and Barto, 2018) update

$$\phi \leftarrow \phi + \beta_t\left(\big[i_t + \rho_{t-1}\gamma_t f_\phi(S_{t-1}) - f_\phi(S_t)\big]\nabla_\phi f_\phi(S_t) - \gamma_t h(S_{t-1})\nabla_\phi f_\phi(S_{t-1})\right), \tag{18}$$

where the auxiliary function $h(s)$ provides an estimate of $\mathbb{E}_\mu[i_t+\rho_{t-1}\gamma_t f_\phi(S_{t-1})-f_\phi(S_t)|S_t = s]$ using a regression update.

Once we use an approximation to the emphatic weighting, we may introduce bias into the gradient estimate. As an extreme case, imagine the function is restricted to predict the same value for all states. Using these estimates means replacing M_t in the actor updates with a constant factor that can be subsumed in the step-size and therefore following the semi-gradient updates in Equation 14. However, the advantage is that these lower-variance estimates do not have to be computed online, unlike M_t. The function f_ϕ can be trained by sampling transitions from a replay buffer (Lin, 1993; Mnih et al., 2015) and applying the update in Equation 17.

5.3 Summary

In this section, we showed how to operationalize the off-policy policy gradient theorem, by discussing how to sample the gradient. The gradient involves a standard actor-critic update from a state s, but additionally weighted by the emphatic weighting $m_\eta(s)$. We leverage existing results for sampling the emphatic weighting in Section 5.2.1, and proved that using this unbiased emphatic weighting in our update results in an unbiased update (Proposition 6). These sampled emphatic weightings, however, can be high variance and become biased if the policy changes on each step. We develop a parameterized estimator in Section 5.2.2, by developing a recursive formula for $f_\phi(s) \approx m(s)/d_\mu(s)$ in Proposition 7. Though a similar form was used to show unbiasedness of a sampled weighting in the original work on emphasis (Sutton et al., 2016), they did not recognize nor leverage this recursive form to develop a TD algorithm to learn a parameterized estimator. We need this $f_\phi(s)$, instead of directly estimating $m(s)$, because states are sampled $s \sim d_\mu$, so weighting by $m(s)$ would be incorrect.

We summarize the final algorithm in Algorithm 1, in Section 7. This pseudocode is provided later, after incorporating entropy regularization and showing how to use the algorithm for episodic problems. The algorithm shows both how to use the emphatic trace and directly-estimated emphatic weightings.

6. Incorporating Entropy Regularization

In this section, we discuss how to extend the off-policy policy gradient theorem to incorporate entropy regularization. We focus on discrete states and actions in the main body, and include results for continuous states and actions in the appendix. Entropy regularization is commonly used with policy gradient methods, as it encodes a preference for more stochasticity in the policy. Recall that at each state, the policy induces a probability distribution over actions whose entropy is defined as

$$\mathcal{H}\big(\pi(.|s)\big) \stackrel{\text{def}}{=} - \sum_{a \in \mathcal{A}} \pi(a|s) \log \pi(a|s).$$

This value captures the stochasticity in the policy. A uniform random policy will maximize entropy, while the entropy of a nearly deterministic policy will be a large negative value.

Entropy regularization is believed to promote exploration, improve the loss surface, and promote faster convergence rates. For exploration, entropy regularization can help make the policy more stochastic and diversify the set of states and actions that the agent learns about (Williams and Peng, 1991; Haarnoja et al., 2018). To find a good policy, the agent

needs accurate estimates of values of different actions in different parts of the state space. A greedy policy only takes actions that are deemed optimal at the current point which results in learning only about a limited number of trajectories. Entropy regularization has also shown to help policy optimization by modifying the landscape. The resulting objective is smoother, which allows the use of larger step-sizes and also reduces the chance of getting stuck in bad local maxima (Ahmed et al., 2019).

Finally, the use of entropy regularization facilitates convergence analysis to stationary points. Mei et al. (2020) showed that entropy regularization can ensure existence of stationary points and also improve the rate of convergence in tabular policy optimization. The idea is similar to convex optimization, where ℓ_2 regularization makes the loss function strongly convex and improves the convergence rate from sublinear to linear. In Appendix E, we extend the proof of existence of a stationary point for entropy regularized policy optimization to state aggregation. Therefore, in addition to extending our algorithm to allow for entropy regularization, we also use this extension to formally prove a counterexample for the semi-gradient updates.

Extending the results to entropy regularization is relatively straightforward, as it relies on the already developed machinery of soft action-values that include entropy in the rewards. Entropy regularization augments each reward with a sample of the entropy at the current state so that the return is modified to

$$\tilde{G}_t \overset{\text{def}}{=} (R_{t+1} - \tau \log \pi(A_t|S_t)) + \gamma_{t+1}(R_{t+2} - \tau \log \pi(A_{t+1}|S_{t+1})) + \dots$$
$$= R_{t+1} - \tau \log \pi(A_t|S_t) + \gamma_{t+1}\tilde{G}_{t+1},$$

where τ is a parameter that controls the amount of regularization. Entropy regularized state values and action values can be defined as (Geist et al., 2019)

$$\tilde{v}_\pi(s) \overset{\text{def}}{=} \mathbb{E}_\pi[\tilde{G}_t|S_t = s] \quad \forall s \in \mathcal{S}$$
$$\tilde{q}_\pi(s, a) \overset{\text{def}}{=} \mathbb{E}_\pi[R_{t+1} + \gamma_{t+1}\tilde{G}_{t+1}|S_t = s, A_t = a] \quad \forall s \in \mathcal{S} \text{ and } a \in \mathcal{A}$$
$$= \sum_{s' \in \mathcal{S}} \mathrm{P}(s'|s, a)[r(s, a, s') + \gamma(s, a, s')\tilde{v}_\pi(s')] \quad \forall a \in \mathcal{A}, \forall s \in \mathcal{S}.$$

Here, $\tilde{v}_\pi(s) = \mathbb{E}_\pi[\tilde{q}_\pi(S, A)|S = s] + \tau\mathcal{H}(\pi(\cdot|s))$ for entropy $\mathcal{H}(\pi(\cdot|s)) = \mathbb{E}_\pi[-\log \pi(A|s)]$. The entropy-regularized objective function simply uses these entropy regularized values,

$$\tilde{J}_\mu(\boldsymbol{\theta}) \overset{\text{def}}{=} \sum_{s \in \mathcal{S}} d_\mu(s)i(s)\tilde{v}_{\pi_{\boldsymbol{\theta}}}(s). \tag{19}$$

The off-policy policy gradient theorem can be extended to the entropy regularized objective. The result looks a little different, because the relationship between the soft values and soft action-values is a little different than in the unregularized setting. Notice that $-\tau \log \pi(a|s)$ is not included in the first reward for $\tilde{q}_\pi(s, a)$, because this first action is given—not random—and entropy is an expectation over all actions. Further, $\log \pi(a|s)$ can be arbitrarily large, though the entropy itself remains nicely bounded. For this reason, we do not define the soft action-values to be $\mathbb{E}_\pi[\tilde{G}_t|S_t = s, A_t = a]$, as we typically would for the unregularized setting. Nonetheless, deriving the policy gradient theorem for these soft action-values uses exactly the same steps. Further, we do recover the unregularized formulation when $\tau = 0$.

Theorem 8 (Off-policy Policy Gradient Theorem for Entropy Regularization)

$$\frac{\partial \tilde{J}_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \sum_{s\in\mathcal{S}} m(s) \left[\sum_a \frac{\partial \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \tilde{q}_\pi(s,a) + \tau \frac{\partial \mathcal{H}(\pi(\cdot|s;\boldsymbol{\theta}))}{\partial \boldsymbol{\theta}} \right]$$

$$= \sum_{s\in\mathcal{S}} m(s) \sum_a \frac{\partial \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \left[\tilde{q}_\pi(s,a) - \tau \log \pi(a|s;\boldsymbol{\theta})) \right]$$

The proof is in Appendix D, as is the extension to continuous states, continuous actions, and other regularizers.

This theorem shows that we can use a similar update as in the unregularized setting, simply by adding $-\tau \log \pi(a|s;\boldsymbol{\theta}))$. Given the true soft action-values $\tilde{q}_\pi(s,a)$, an update with an unbiased sample of the gradient is

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha\rho_t M_t \frac{\partial \log \pi(A_t|S_t;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \left[\tilde{q}_\pi(S_t, A_t) - \tau \log \pi(A_t|S_t;\boldsymbol{\theta})) \right].$$

In practice, we subtract a baseline and use an estimate δ_t of the advantage $\tilde{q}_\pi(s,a) - \tau \log \pi(a|s;\boldsymbol{\theta}) - \tilde{v}_\pi(s)$, by only directly estimating the soft values $v(s) \approx \tilde{v}_\pi(s)$. The 1-step return sample is again the TD error, but with entropy regularization added to the rewards: $\delta_t = R_{t+1} - \tau \log \pi(A_t|S_t) + \gamma_{t+1}v(S_{t+1}) - v(S_t)$.

7. Formulating Episodic Problems as a Special Case

The goal in an episodic problem is to maximize the expected episodic return. Mismatch between the algorithm's objective and this goal, even in the on-policy setting, can lead the agent to suboptimal policies (Thomas, 2014; Nota and Thomas, 2020). This section describes how we can use the interest function to maximize the right objective.

The objective function in Equation 4 that weights state values by the start state distribution is equal to expected episodic return. With limited function approximation resources this weighting matters. If the parametrized function is incapable of representing a policy that maximizes all state values, the agent has to settle for lower values in some states to maximize the values of more important ones. For example, if the weighting is d_μ, the agent may incorrectly prioritize states that appear less often at the start of an episode and more frequently at other points in the behaviour policy's trajectories, and fail to optimize the episode return.

The interest function in Equation 6 allows us to focus the parametrized function's resources on states of interest. Recall that we assume that the agent observes interest i_t in state S_t, with $i(s) = \mathbb{E}[i_t \mid S_t = s]$. So far, we have not used the additional flexibility that the interest itself can be random, but for this unification we will use this property. If we could set $i(S_t) = d_0(S_t)/d_\mu(S_t)$ then the objective function would correspond to the episodic objective. However, neither d_μ nor d_0 is available to the agent, and we would like to avoid estimating those distributions.

Fortunately, we can show that if the signal is set to one at the beginning of an episode and set to zero thereafter, its expectation in the limit of time will be proportional to the correct ratio. Under transition-based discounting, the agent is informed that an episode has begun whenever it receives a discount factor of zero: namely, that the transition (S_t, A_t, S_{t+1})

resulted in termination. Therefore, we can obtain this result by allowing for the interest to be a function of the whole transition.

Proposition 9 *If the signal i_t is set to*

$$i_t \overset{\text{def}}{=} \begin{cases} 1, & \text{if } \gamma_t = 0 \text{ (i.e. the beginning of an episode)} \\ 0, & \text{otherwise} \end{cases} \tag{20}$$

then its expected value under the behaviour policy is

$$\mathbb{E}_\mu[i_t \mid S_t = s] \propto \frac{d_0(s)}{d_\mu(s)}.$$

Proof

$$\begin{aligned} \mathbb{E}_\mu[i_t \mid S_t = s] &= \Pr(\gamma_t = 0 \mid S_t = s) \\ &= \frac{\Pr(S_t = s \mid \gamma_t = 0)\Pr(\gamma_t = 0)}{\Pr(S_t = s)} \\ &= \frac{d_0(s)\Pr(\gamma_t = 0)}{d_\mu(s)} \\ &\propto \frac{d_0(s))}{d_\mu(s)}, \end{aligned}$$

where we used the fact that $\Pr(S_t = s \mid \gamma_t = 0) = d_0(s)$ and $\Pr(S_t = s) = d_\mu(s)$. ∎

The constant term is the same for all states, $\Pr(\gamma_t = 0)$. It simply reflects the probability of termination under the behaviour policy. This constant term does not change the relative ordering between policies, since all weight is still on the start states. Therefore, the resulting objective is equivalent to the episodic objective.

We can relate the resulting update to the on-policy and off-policy updates used for the episodic setting. In the on-policy setting, if we use the interest function given by Equation 20 and a constant discount factor γ during the episode, then the update reduces to the unbiased episodic actor-critic algorithm of Sutton and Barto (2018), originally proposed by Thomas (2014).

Sutton et al. (1999a) proved the policy gradient theorem for the episodic setting

$$\frac{\partial J_0(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \sum_{s \in \mathcal{S}} \bar{d}_\pi(s) \sum_a \frac{\partial \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, a) \qquad \text{for} \quad \bar{d}_\pi(s) = \sum_{t=0}^{\infty} \gamma^t \Pr(s; t, \pi),$$

where $\Pr(s; t, \pi)$ is the probability of going from a start state to state s in t steps under π, the weighting in the gradient $\bar{d}_\pi$ is the discounted state visit distribution, and the agent should discount updates that occur later in the episode to account for this weighting. An algorithm that does not discount later updates—and thus samples according to d_π—results in a biased update.

To fix this problem, Thomas (2014) proposed the unbiased actor-critic algorithm with the updates rules

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha I \delta \nabla_{\boldsymbol{\theta}} \log \pi(A|s; \boldsymbol{\theta})$$

19

$$I \leftarrow \gamma I.$$

Since I is initialized to 1 and updated after the update to $\boldsymbol{\theta}$, I will be 1 during the first weight update, and will decay by γ each timestep thereafter.

The on-policy ACE update rule with $\rho = 1$ and $\eta = 1$ is

$$F_t \leftarrow \gamma_t F_{t-1} + i_t \qquad \text{for } F_0 = 0$$
$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha F_t \delta \nabla_{\boldsymbol{\theta}} \log \pi(A|s; \boldsymbol{\theta}).$$

Because $i_t = 1$ on the first time step and F is initialized to 0, F will be 1 on the first weight update and will decay by γ each time step thereafter. From this inspection it is clear that the ACE update reduces to the unbiased actor-critic update in the on-policy episodic setting. The final ACE algorithm with all the discussed techniques is in Algorithm 1.

Algorithm 1 Online Actor Critic with Emphatic weightings (ACE)

Initialize weights for actor $\boldsymbol{\theta}$ and critic $\mathbf{w}$
For emphatic trace, initialize $F_0 = 0$; for directly-estimated emphatic weightings, initialize weights $\boldsymbol{\phi}$ for approximate emphatic weightings $f_{\boldsymbol{\phi}}$
Suggested (default) settings of parameters: $\eta = 0.1$, $\tau = 0.01$
Obtain initial feature vector $\mathbf{x}_0$ and set $i_0 \leftarrow 1$
repeat
    Choose an action a_t according to $\mu(\cdot|\mathbf{x}_t)$
    Observe reward r_{t+1}, next state vector $\mathbf{x}_{t+1}$ and γ_{t+1}
    For episodic setting: Set $i_{t+1} = 1$ if a new episode has begun; else $i_{t+1} = 0$
    For excursions setting: If no preferences between states, set $i_{t+1} = 1$
    $\tilde{r}_{t+1} \leftarrow r_{t+1} - \tau \log \pi(a_t|\mathbf{x}_t; \boldsymbol{\theta})$
    $\rho_t \leftarrow \frac{\pi(a_t|\mathbf{x}_t; \boldsymbol{\theta})}{\mu(a_t|\mathbf{x}_t)}$
    Update (entropy-regularized) critic $v_{\mathbf{w}}$
    if using emphatic trace **then**
        $M_t \leftarrow (1 - \eta)i_t + \eta F_t$
        $F_{t+1} \leftarrow \rho_t \gamma_{t+1} F_t + i_{t+1}$
    else
        $M_t \leftarrow (1 - \eta)i_t + \eta f_{\boldsymbol{\phi}}(\mathbf{x}_t)$
        $\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} + \beta_t \big[i_{t+1} + \rho_t \gamma_{t+1} f_{\boldsymbol{\phi}}(\mathbf{x}_t) - f_{\boldsymbol{\phi}}(\mathbf{x}_{t+1})\big] \nabla_{\boldsymbol{\phi}} f_{\boldsymbol{\phi}}(\mathbf{x}_{t+1})$
    $\delta_t \leftarrow \tilde{r}_{t+1} + \gamma_{t+1} v_{\mathbf{w}}(s_{t+1}) - v_{\mathbf{w}}(s_t)$
    $\boldsymbol{\psi} \leftarrow \nabla_{\boldsymbol{\theta}} \log \pi(b|\mathbf{x}; \boldsymbol{\theta})$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha_t \rho_t M_t \delta_t \boldsymbol{\psi}$
until agent done interaction with environment

---

8. Stationary Points under the Semi-gradient and Gradient Updates

In this section, we provide more insight into the difference between the stationary points of the weighted excursions objective—obtained using the true gradient in Equation 9—versus those obtain with the semi-gradient update underlying OffPAC, in Equation 14. We first

provide a counterexample showing that all the stationary points for the semi-gradient have poor performance. In fact, initializing at the optimal solution, and then updating with the semi-gradient, moves the solution away to one of these highly suboptimal stationary points. We then discuss that the semi-gradient update can actually be seen as a full gradient update, albeit with a different implicit state weighting. This connection illuminates why the semi-gradient does so poorly on the counterexample, but also potentially sheds light on why OffPAC has generally performed reasonably in practice.

8.1 A Counter-example for the Semi-Gradient Update

Recall that, in the derivation of the policy gradient theorem, the product rule breaks down the gradient of the value function into the two terms shown in Equation 11. The policy gradient theorem in OffPAC only considers the first term, i.e. the gradient of the policy given a fixed value function. The approximated gradient is

$$\frac{\partial J_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \approx \sum_{s \in \mathcal{S}} \mathbf{d}_\mu(s) \sum_a \frac{\partial \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, a).$$

The approximation above weights the states by the behaviour policy's state distribution instead of emphatic weightings.

To see the difference between these weightings, consider the MDP in Figure 1a. For the actor, s_0 has a feature vector $[1, 0]$, and the feature vector for both s_1 and s_2 is $[0, 1]$. This aliased representation forces the actor to take a similar action in s_1 and s_2. The behaviour policy takes actions a_0 and a_1 with probabilities 0.25 and 0.75 in all non-terminal states, so that s_0, s_1, and s_2 will have probabilities 0.5, 0.125, and 0.375 under $\mathbf{d}_\mu$. The target policy is initialized to take a_0 and a_1 with probabilities 0.9 and 0.1 in all non-terminal states, which is close to optimal.

We trained two actors on this MDP, one with semi-gradient updates and one with gradient updates. The actors are initialized to the target policy above and the updates use exact values rather than critic estimates. States and actions are sampled from the behaviour policy. As shown in Figures 1b and 1c, while both methods start close to the highest attainable value of the objective function (that of a deterministic policy that takes a_0 everywhere), semi-gradient updates move the actor towards a suboptimal policy and reduce the objective function along the way. Gradient updates, however, increase the probability of taking a_0 in all states and increase the value of the objective function.

The problem with semi-gradient updates boils down to the weighting. In an expected semi-gradient update, each state tries to increase the probability of the action with the highest action-value. There will be a conflict between the aliased states s_1 and s_2 because their highest-valued actions differ. If the states are weighted by d_μ in the expected update, s_1 will appear insignificant to the actor, and the update will increase the probability of a_1 in the aliased states. The ratio between $q_\pi(s_1, a_0)$ and $q_\pi(s_2, a_1)$ is not enough to counterbalance this weighting.

However, s_1 has an importance that the semi-gradient update overlooks. Taking a suboptimal action at s_1 will also reduce $q(s_0, a_0)$, and after many updates the actor will gradually prefer to take a_1 in s_0. Eventually, the target policy will be to take a_1 at all states, which has a lower value under the objective function than the initial target policy.

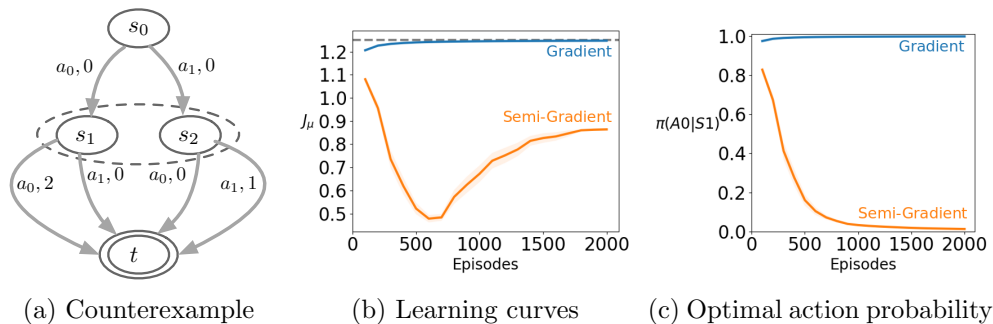(a) Counterexample   (b) Learning curves   (c) Optimal action probability

Figure 1: (a) A counterexample that demonstrates the suboptimal behaviour of semi-gradient updates. The semi-gradients converge for the tabular setting (Degris et al., 2012b), but not necessarily under function approximation—such as with the state aliasing in this MDP. The start state is denoted s_0 and the terminal state is denoted t. States s_1 and s_2 are aliased to the actor. The interest $i(s)$ is set to one for all states. (b) Learning curves comparing semi-gradient updates and gradient updates, averaged over 30 runs with negligible standard error bars. The actor has a softmax output on a linear transformation of features and is trained with a step-size of 0.1 (though results were similar across all the stepsizes tested). The dashed line shows the highest attainable objective function under the aliased representation. (c) The probability of taking the optimal action (a_0) in the aliased states.

This experiment highlights why the weight of a state should depend not only on its own share of d_μ, but also on its predecessors. The behaviour policy's state distribution is not the proper deciding factor in the competition between s_1 and s_2, even when optimizing the excursions objective.

Proposition 10 formalizes the problem with semi-gradient updates by showing that, under any $\tau > 0$, semi-gradient updates will not converge to a stationary point of the objective function in the counterexample. The requirement $\tau > 0$ is only needed to ensure existence of a stationary point; this result holds for τ arbitrarily close to zero. The proof is presented in Appendix E.

Proposition 10 *For any $\tau > 0$ where $\tau \neq \tau_i \approx 0.2779$ in the three-state counterexample, semi-gradient updates do not converge to a stationary point of the objective function.*

8.2 The Semi-Gradient as a True Gradient with a Different Weighting

In this section, we show that the semi-gradient can locally be seen as a gradient update, with a different state weighting in the objective. Locally for the current weights $\boldsymbol{\theta}_t$ with corresponding policy π, that state weighting is $\mathbf{d} = \mathbf{d}_\mu(\mathbf{I} - \mathbf{P}_{\pi,\gamma})$, with objective $J_d(\boldsymbol{\theta}) \stackrel{\text{def}}{=} \sum_{s \in \mathcal{S}} d(s) v_{\pi_{\boldsymbol{\theta}}}(s)$. The resulting $\mathbf{d}$ may not be a distribution. In fact, it may not even be positive! If it is negative, the objective tells the agent to minimize the value for that state. This is precisely what occurs in the above counterexample.

First, to see why this is the implicit state weighting, notice that for the semi-gradient update to be a gradient update, the weighting d_μ used in the update has to correspond to an emphatic weighting m_d for some state weighting d in the objective. In other words, $d_\mu(s) = m_d(s)$ where $\mathbf{m}_d = \mathbf{d}(\mathbf{I} - \mathbf{P}_{\pi,\gamma})^{-1}$. This requirement implies that $\mathbf{d} = \mathbf{m}_d(\mathbf{I} - \mathbf{P}_{\pi,\gamma})$. Therefore, by using the weighting d_μ in the expected gradient, the semi-gradient update locally around $\boldsymbol{\theta}_t$ can be seen as a gradient update on J_d. This weighting d actually changes as $\boldsymbol{\theta}_t$ changes. In fact, we know that the semi-gradient update cannot be seen as the gradient of a fixed objective function, from our counterexample for OffPAC and the result for the on-policy setting showing that omitting the discount factor results in an update that is not a gradient (Nota and Thomas, 2020). However, even if this interpretation is only valid locally at each update, such a negative weighting can be problematic.

We can compute the implicit weighting d, in our counter-example. Let $b = \mu(a_0|s_0)$ and $p = \pi(a_0|s_0)$. Then we know that $d_\mu(s_0) = 0.5$, $d_\mu(s_1) = 0.5b = 0.125$ and $d_\mu(s_0) = 0.5(1 - b) = 0.375$. Further, we know that $m(s_0) = d_\mu(s_0)$ and

$$m(s_1) = d_\mu(s_1) + \pi(a_0|s_0)m(s_0) = bd_\mu(s_0) + pm(s_0) = d_\mu(s_0)(b + p) = 0.5(b + p)$$
$$m(s_2) = d_\mu(s_2) + (1 - p)m(s_0) = d_\mu(s_0)((1 - b) + (1 - p)) = 0.5(2 - b - p).$$

The implicit d for the semi-gradient update, locally around this π, has $\tilde{m}(s) = d_\mu(s)$ where $\tilde{m}(s_0) = d(s_0)$ and so $d(s_0) = d_\mu(s_0) = 0.5$ and

$$\tilde{m}(s_1) = d(s_1) + p\tilde{m}(s_0) = d(s_1) + pd_\mu(s_0) = d(s_1) + 0.5p$$
$$\tilde{m}(s_2) = d(s_2) + (1 - p)\tilde{m}(s_0) = d(s_2) + (1 - p)d_\mu(s_0) = d(s_2) + 0.5(1 - p).$$

Using $\tilde{m}(s_1) = d_\mu(s_1) = 0.5b$ and $\tilde{m}(s_2) = d_\mu(s_2) = 0.5(1 - b)$, we get that

$$d(s_1) = \tilde{m}(s_1) - 0.5p = 0.5(b - p)$$
$$d(s_2) = \tilde{m}(s_2) - 0.5(1 - p) = 0.5((1 - b) - (1 - p)) = 0.5(p - b).$$

If $b > p$, then $d(s_2) < 0$; if $b < p$, then $d(s_1) < 0$; and if $b = p$ then both are zero. In our counterexample, we set $b < p$, making the update move away from increasing the value in s_1, namely preferring to increase the value in s_2 and decrease the value in s_1. The iterative updates maintain the condition $b < p$, even as the policy changes—which changes p—so the implicit weighting systematically causes convergence to a suboptimal policy.

We note that the implicit weighting is independent of the representation. However, we know that the semi-gradient update converges to a stationary point of the excursions objective for the tabular setting. This might seem odd, given that the implicit weighting is negative for a state in this counterexample. However, this is not contradictory. Recall in Section 2 we discussed that in the tabular setting, the condition for the stationary point is that the gradient must be zero at every state, independently. Summing up zeros, even when weighting those zeros with a negative weighting, still results in zero.

There has been other work that has noted that optimizing under a different state weighting can still be effective. Proposition 6 of Ghosh et al. (2020) shows that a lower bound can be optimized, using data generated under the behaviour policy. The action-values of the behaviour policy is used, and only the log likelihood terms of the target policy are differentiated. This lower bound, however, is only an approximation to the true objective.

Our result differs, in that it highlights that equivalent solutions can be obtained, even under different state weightings.

This view suggests a direction to reduce variance in the ACE updates: consider appropriate implicit weightings d, that allow for low variance emphatic updates. One potentially promising approach is to only consider emphatic weightings a small number of steps back-in-time. Another is to err on the side of smaller η in the emphatic trace in ACE, knowing that the implicit weighting d may remain reasonable for a broad range of η.

9. Experiments: Studying Properties of ACE

In this section we investigate key properties of the ACE algorithm empirically.[7] Specifically, we study the effects of the trade-off parameter and the choice of estimator for the emphatic weightings on the learned policy, as these choices are central to the ACE algorithm. First, we revisit the simple counterexample from Section 8.1 to examine how the trade-off parameter can affect the learned policy. Next, we illustrate some issues with using the emphatic trace in the ACE algorithm by testing it on a modified version of the counterexample. We then move to two classic control environments to study the two estimators discussed in Section 5.2 and their effects on the learned policy. Finally, we test several variants of ACE on a challenging environment designed to illustrate the issues associated with both estimators. Please see Table 1 in Appendix F for a description of each of the algorithms compared.

9.1 The Impact of the Trade-Off Parameter

The parameter η can be interpreted as trading off bias and variance. For $\eta = 0$, the bias can be significant, as shown in the previous section. A natural question to ask is how the bias changes as η ranges from 0 to 1.

To answer this question, we repeated the experiment in Section 8.1, but this time with η taking values in $\{0, 0.25, 0.5, 0.75, 1\}$ and the actor's step-size taking values in $\{0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1\}$, with the best-performing step-size (by total area under the learning curve, averaged over 30 runs) for each value of η used in Figures 2b and 2c. To highlight the rate of learning, the actor's policy was initialized to take each action with equal probability. The results are displayed in Figure 2 with shaded regions depicting standard error.

Figure 2a shows the performance of ACE with each value of η over the range of step-sizes tested. ACE performed well for a range of step-sizes, and even small values of η significantly improved over $\eta = 0$ (OffPAC). However, the performance of ACE with $\eta > 0$ was more sensitive to the choice of step-size, with performance decreasing as the step-size grew large, while the performance of $\eta = 0$ was lower overall but remained steady with larger step-sizes.

Figure 2b plots the learning curves for ACE with each value of η. For $\eta = 0$ (OffPAC), the algorithm decreased the objective function during learning to get to a suboptimal fixed point, while $\eta > 0$ always improved the objective function relative to the starting point. For a surprisingly small choice of $\eta = 0.5$, the actor converged to the optimal solution, and even $\eta = 0.25$ produced a much more reasonable solution than $\eta = 0$.

---

7. Code is available at: `https://github.com/gravesec/actor-critic-with-emphatic-weightings`.

Figure 2c shows the probability of taking the optimal action in the aliased states. The optimal policy is to take a_0 in the aliased states with probability 1. ACE with $\eta = 0$ (OffPAC) quickly converged to the suboptimal solution of choosing the best action for s_2 instead of s_1. Even with η just a bit higher than 0, convergence is to a more reasonable solution, choosing the optimal action the majority of the time.
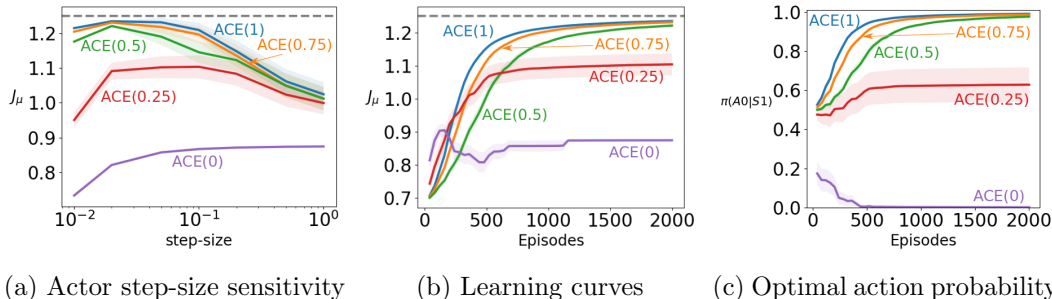

(a) Actor step-size sensitivity   (b) Learning curves   (c) Optimal action probability

Figure 2: Performance of ACE with different values of η in the counterexample.

To determine whether the η parameter has similar effects when ACE is used with a learned critic, we repeated the previous experiment but this time using value estimates from a critic trained with GTD(λ) (Maei, 2011). We checked all combinations of the following critic step-sizes: $\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}\}$, the following critic trace decay rates: $\{0, 0.5, 1.0\}$, and the following actor step-sizes: $\{10^{-10}, 10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}\}$, and used the best-performing combination (by area under the learning curve) for each value of η in Figures 3b and 3c. We averaged the results over 10 runs, and plotted the standard error as shaded regions. Figure 3 shows that, as before, even relatively small values of η can achieve close to the optimal solution. However, $\eta = 0$ (OffPAC) still finds a suboptimal policy. Overall, the outcomes are similar to the previous experiment, although noisier due to the use of a learned critic rather than the true value function.

(a) Actor step-size sensitivity   (b) Learning curves   (c) Optimal action probability
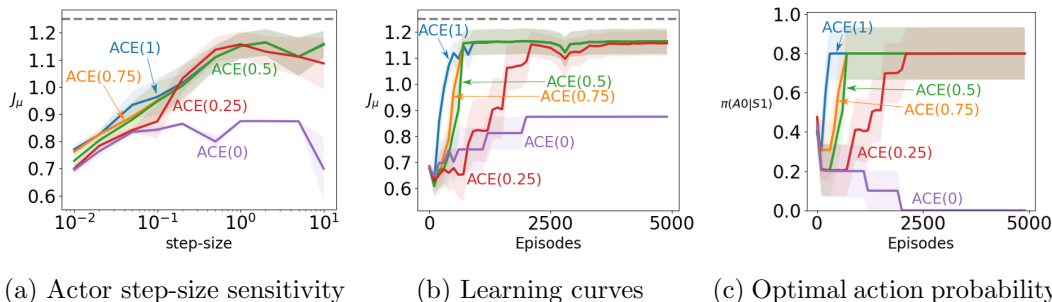
Figure 3: Performance of ACE with a GTD(λ) critic and different values of η in the counterexample.

So far we have only considered ACE with a discrete action policy parameterization. However, an appealing property of actor-critic algorithms is their ability to naturally handle continuous action spaces. To determine if the findings above generalize to continuous action policy parameterizations, we created an environment similar to Figure 1a, but with one

25

continuous unbounded action. Taking action with value a at s_0 will result in a transition to s_1 with probability $1 - \sigma(a)$ and a transition to s_2 with probability $\sigma(a)$, where σ denotes the logistic sigmoid function. For all actions from s_0, the reward is zero. From s_1 and s_2, the agent can only transition to the terminal state, with reward $2\sigma(-a)$ and $\sigma(a)$ respectively. The behaviour policy takes actions drawn from a Gaussian distribution with mean 1.0 and variance 1.0.

Because the environment has continuous actions, we can include both stochastic and deterministic policies, and so can include DPG in the comparison. DPG is built on the semi-gradient, like OffPAC (Silver et al., 2014). We include True-DPG with Emphatic weightings (True-DPGE), which uses the true emphatic weightings rather than estimated ones to avoid the issue of estimating the emphatic weightings for a deterministic target policy, and focus the investigation on whether DPG converges to a suboptimal solution in this setting. Estimation of the emphatic weightings for a deterministic target policy is left for future work. The stochastic actor in ACE has a linear output unit and a softplus output unit to represent the mean and the standard deviation of a Gaussian distribution. All actors are initialized with zero weights.

Figure 4 summarizes the results. The first observation is that DPG demonstrates suboptimal behaviour similar to OffPAC. As training goes on, DPG prefers to take positive actions in all states, because s_2 is updated more often. This problem goes away in True-DPGE. The emphatic weightings emphasize updates in s_1 and, thus, the actor gradually prefers negative actions and surpasses DPG in performance. Similarly, True-ACE learns to take negative actions but, being a stochastic policy, it cannot achieve True-DPGE's performance on this domain. ACE with different η values, however, cannot outperform DPG, and this result suggests that an alternative to importance sampling ratios is needed to effectively extend ACE to continuous actions.

(a) Stepsize sensitivity   (b) Learning curves   (c) Mean action
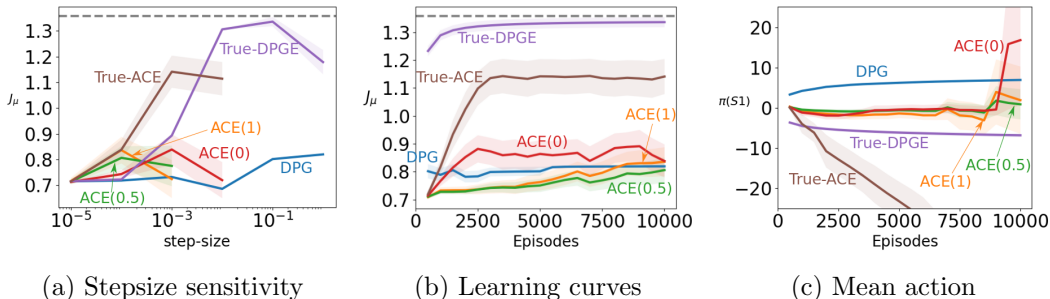
Figure 4: Performance of ACE with different values of η, True-ACE, DPG, and True-DPGE on the continuous action MDP. The results are averaged over 30 runs. For continuous actions, the methods have even more difficulty getting to the optimal solutions, given by True-DPGE and True-ACE, though the action selection graphs suggest that ACE for higher η is staying nearer the optimal action selection than ACE(0) and DPG.

9.2 Challenges in Estimating the Emphatic Weightings

Up to this point we have been using the emphatic trace originally proposed by Sutton et al. (2016) to estimate the emphatic weightings. As discussed in Section 5, there can be multiple sources of inaccuracy from using this Monte Carlo estimate in an actor-critic framework.

However, it is unclear how the issues affecting the emphatic trace manifest in practice, and hence whether introducing a parametrized function to directly estimate the emphatic weightings is worth the added bias—especially if the representation is poor. To study the effects of this choice of estimator on the ACE algorithm, we conducted a series of experiments starting with a modified version of the counterexample in Figure 1a, moving to two classic control environments, and ending with a challenging environment designed to illustrate the issues associated with both estimators.
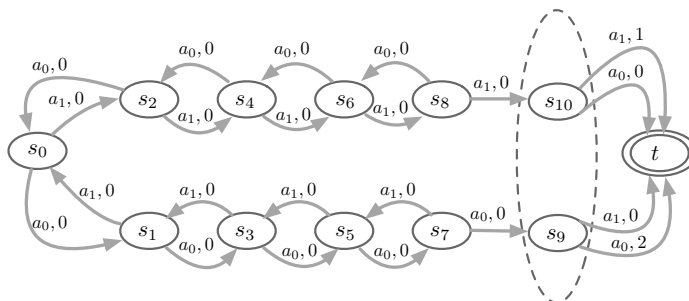
Figure 5: An 11-state MDP that makes estimating the emphatic weightings with the emphatic trace more difficult.

The first environment, shown in Figure 5, is an extended version of the counterexample with two long chains before the aliased states. Like the original counterexample, the behaviour policy takes a_0 with probability 0.25 and a_1 with probability 0.75 in all non-terminal states, and the interest $i(s)$ is set to 1 for all states. Each state is represented by a unique feature vector except states s_9 and s_{10}, which are aliased and appear the same to the actor. The addition of the new states makes trajectories considerably longer, which may exacerbate the issues with the emphatic trace.

We repeated the experiment from Figure 2 on the long counterexample. The following actor step-sizes were tested: $\{5 \cdot 10^{-5}, 10^{-4}, 2 \cdot 10^{-4}, 5 \cdot 10^{-4}, 10^{-3}, 2 \cdot 10^{-3}, 5 \cdot 10^{-3}, 10^{-2}\}$, with the best-performing value (by total area under the learning curve, averaged over 10 runs) for each value of η used in Figures 6b and 6c. The actor was again initialized to take each action with equal probability, and the true state values were again used in the updates in order to isolate the effects of the emphatic trace.

We also trained an actor called True-ACE that uses the true emphatic weightings for the current target policy and behaviour policy, computed at each timestep. The performance of True-ACE is included here for the sake of comparison, as computing the exact emphatic weightings is not generally possible in an unknown environment.

The results in Figure 6 show that, even though performance improves as η is increased, there is a significant gap between ACE with $\eta = 1$ and True-ACE. Unlike Figure 2, the methods have more difficulty reaching the optimal solution, although ACE with larger η does still find a significantly better solution than $\eta = 0$. Additionally, values of η greater

than 0 result in high-variance estimates of the emphatic weightings, which lead to more variable performance, as shown by the larger shaded regions representing standard error. These results overall show the inaccuracies pointed out in Section 5 indeed disturb the updates in long trajectories.
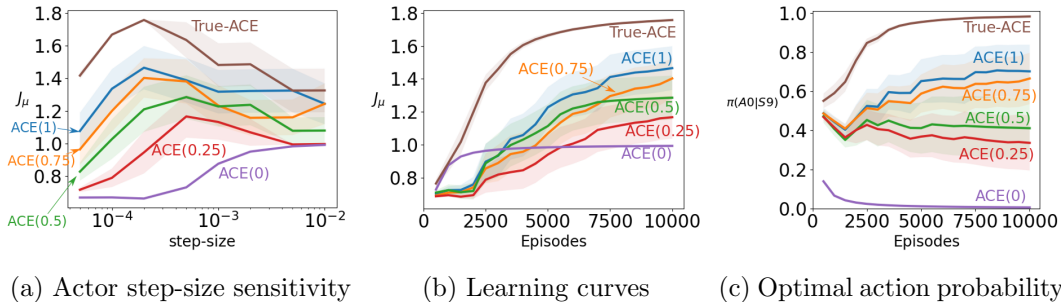

(a) Actor step-size sensitivity   (b) Learning curves   (c) Optimal action probability

Figure 6: Performance of ACE with different values of η on the 11-state MDP.

9.3 Estimating the Emphatic Weightings in Classic Control Environments

The results of the previous experiments suggest that using the emphatic trace to estimate the emphatic weightings can significantly impact the performance of ACE. To better understand how the issues with the emphatic trace affect the learning process beyond our counterexample, we tested several variants of ACE on off-policy versions of two classic control environments: Puddle World (Degris et al., 2012b) and Mountain Car (Moore, 1990).

As an off-policy Monte Carlo estimator of the emphatic weightings, the emphatic trace can yield extremely high-variance estimates which can interfere with learning (Ghiassian et al., 2018). To see how the variance of the emphatic trace affects ACE, we tested a variant called ACE-direct that uses the one-step temporal-difference update from Section 5.2.2 to estimate the emphatic weightings. Temporal-difference methods often have lower variance than Monte Carlo methods—at the cost of introducing bias—which makes using a TD-style update to estimate the emphatic weightings an appealing alternative to the emphatic trace (Sutton and Barto, 2018).

As discussed in Section 5.2.2, the emphatic trace can also be biased when used in an actor-critic algorithm where the target policy is changing. To determine how detrimental this bias is to the performance of ACE, we tested a variant called ACE-ideal where all importance sampling ratios in the emphatic trace are re-computed at each time step using the current target policy, yielding an unbiased Monte Carlo estimate of the emphatic weighting for the current state. ACE-ideal is not a practical algorithm, as the computation and memory required grow with the number of time steps, but it allows us to isolate the effects of the bias introduced by using the emphatic trace with a changing target policy.

Other baselines for comparison included OffPAC, and ACE using the emphatic trace (referred to as ACE-trace). To determine the effects of different choices of critic, we included versions of each algorithm using an ETD critic (Sutton et al., 2016) and a TDRC critic (Ghiassian et al., 2020). We also included versions of each algorithm using the uniform interest function (i.e., $i = 1$ for all time steps), as well as the episodic interest function from Section 7—with the exception of OffPAC. OffPAC scales policy updates using only the

interest for the current time step, which when combined with the episodic interest function leads to a single policy update on the first time step followed by no updates thereafter.

To get a clear picture of the performance of each algorithm, we conducted a grid search on the free parameters of each algorithm. For the step size parameter of the actor, critic, and the direct estimator of emphatic weightings, we tested values of the form $\frac{1}{2}^i$ where i ranged from 0 to 15. For the trace decay rate of the critic, we tested values of the form $1 - \frac{1}{2}^j$ where j ranged from 0 to 6. The discount factor was .95. We sought to establish the performance of each variant for $\eta = 1$, as they all reduce to OffPAC when $\eta = 0$.

Each combination of parameters for each algorithm was run on 5 different trajectories generated by a fixed behaviour policy interacting with the environment for 100,000 time steps. The learned policies were saved every 1,000 time steps and evaluated 50 times using both the episodic and excursions objective functions from Section 3. For the episodic objective function, the policies were evaluated by creating 50 different instances of the environment and executing the target policy from the starting state until termination or 1000 time steps had elapsed. The excursions objective function was evaluated similarly, but with the environment's starting state drawn from the behaviour policy's steady state distribution, chosen by running the behaviour policy for 50,000 time steps and saving every thousandth state. The results were averaged, and the best-performing combinations (by area under the learning curve for the appropriate objective function) were re-run on enough different trajectories to reduce the standard error to an acceptable level (100 runs for Puddle World, 30 runs for Mountain Car). The initial parameter sweep was intended to find good parameter settings for each method at a reasonable computational cost, and not necessarily the absolute best performance.

9.3.1 Puddle World

The Puddle World environment is a 2-dimensional continuous gridworld containing a start location, goal location, and puddles through which it is costly to move. While it first appeared in Boyan and Moore (1995), we use the version from Degris et al. (2012b) that includes an additional puddle near the goal state which makes the task more difficult.[8] The behaviour policy took the North, East, South, and West actions with probabilities .45, .45, .05, and .05 respectively. The observations were tile coded with a fairly low-resolution tile coder (4 tilings of 2×2 tiles plus a bias unit) to generate feature vectors with a large degree of generalization.

Figure 7 presents the results for the Puddle World environment. The left-hand column shows learning curves, while the right-hand column contains sensitivity analyses for the actor's step size. The first four plots (figures 7a, 7b, 7c, and 7d) show results when using ETD as the critic, while the last four plots (figures 7e, 7f, 7g, and 7h) show results for a TDRC critic. The first and third rows show results using the excursions objective function, and the second and fourth rows show results for the episodic objective function.

Comparing the performance of each algorithm under the excursions and episodic objective function, we can see that the algorithms all performed better in the excursions setting than the episodic setting. This is due to the excursions setting being easier than the episodic setting for the classic control problems in this section. In the excursions setting, the agent
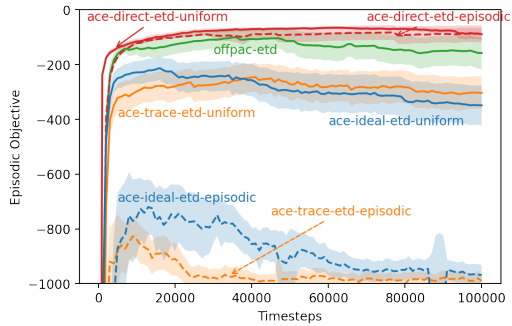
---

8. Please see Degris et al. (2012b) for a picture of the Puddle World environment.
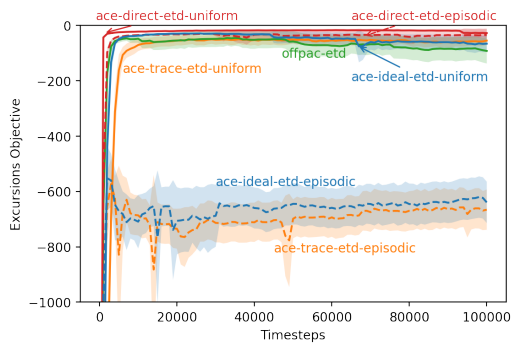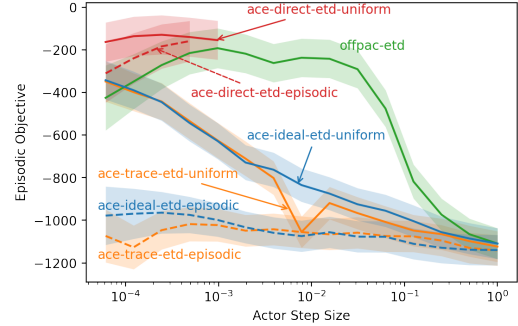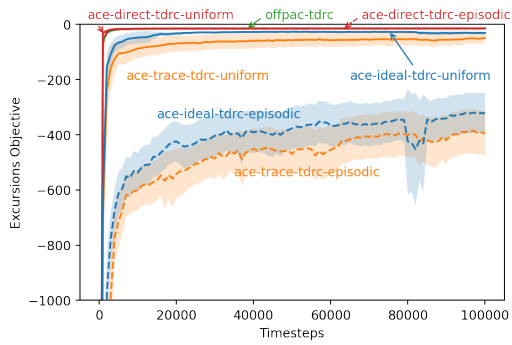
(a) Excursions learning curves (ETD)
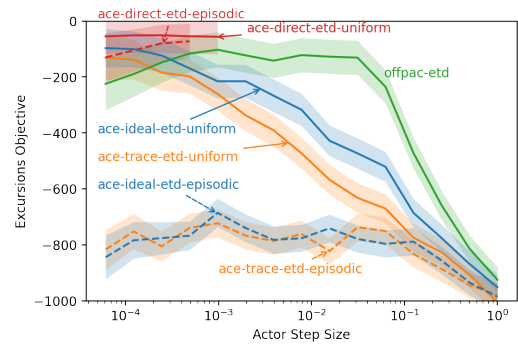
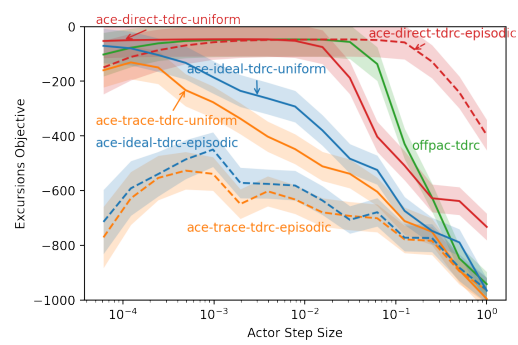(b) Excursions step-size sensitivity (ETD)
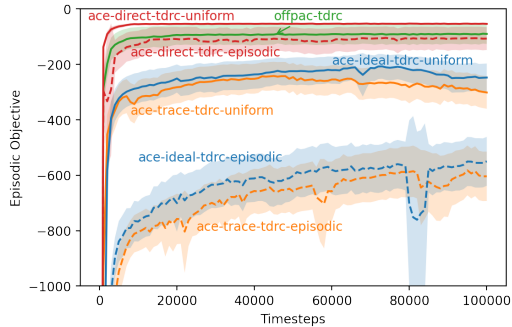
(c) Episodic learning curves (ETD)
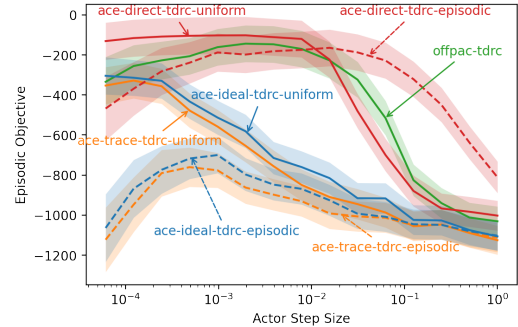
(d) Episodic step-size sensitivity (ETD)

(e) Excursions learning curves (TDRC)

(f) Excursions step-size sensitivity (TDRC)

(g) Episodic learning curves (TDRC)

(h) Episodic step-size sensitivity (TDRC)

Figure 7: Results on Puddle World. Shaded regions are 95% confidence intervals.

starts from the behaviour policy's state distribution, which in the problems studied is closer to the goal state than the start state. This is not necessarily true for all problems, however, and in general there is no strict ordering of objectives in terms of difficulty.

ACE variants that used the episodic interest function (dashed lines) generally performed much worse than their counterparts using the uniform interest function (solid lines), with the exception of ACE-direct. This is consistent with the discussion in Section 7 of Thomas (2014)—although they deal with the on-policy case—which states scaling policy updates by an accumulating product of discount factors hurts the sample efficiency of the algorithm. However, the off-policy case can exacerbate this issue as the accumulating product includes both discount factors and importance sampling ratios. The fact that the direct method of estimating emphatic weightings allowed ACE to perform well when using the episodic interest function is intriguing and merits further study.

The choice of critic affected the performance of the algorithms to differing degrees. Replacing an ETD critic with a TDRC critic improved the performance of each of the algorithms, with episodic performance showing the greatest improvement (compare figures 7c and 7g), perhaps due to the excursions objective being easier for this environment as previously mentioned. ACE-direct with uniform interest performed well regardless of the choice of critic, but improved substantially in the episodic setting when using a TDRC critic. Using a TDRC critic instead of an ETD critic also improved the sensitivity of the algorithms to the actor's step size, allowing a slightly wider range of step sizes to perform well for most methods (compare Figure 7b to 7f, and Figure 7d to 7h).

When comparing ACE-trace (orange) to ACE-ideal (blue)—where the emphatic trace's importance sampling ratios were computed using the current policy, resulting in an unbiased Monte Carlo estimate of the emphatic weightings—we can see that correcting the bias introduced by using the emphatic trace with a changing policy did not improve performance significantly in all but one situation. The one exception was when performance was evaluated with the excursions objective function (figures 7a and 7e). When using an ETD critic (Figure 7a), ACE-ideal outperformed ACE-trace early in learning before deteriorating to a similar level of performance by the end of the experiment. Conversely, ACE-ideal outperformed ACE-trace throughout the experiment when using a TDRC critic (Figure 7e), although the difference was barely significant. In all other situations, the performance of ACE-ideal was not significantly different from the performance of ACE-trace.

Overall, ACE-direct performed better than or equal to ACE-trace, ACE-ideal, and Off-PAC across all combinations of critic, interest function, and objective function. However, it diverged for some actor step sizes when used with an ETD critic (figures 7b and 7d). This could be due to the use of a semi-gradient update rule (analogous to off-policy TD(0) which is not guaranteed to converge when used with function approximation) for the direct method of estimating emphatic weightings. To resolve this issue, one could use the gradient-based update rule in equation 18 of Section 5.2.2. We chose to use the semi-gradient update to keep both the explanation and experiments simple, as there were already a large number of concepts and algorithmic parameters involved.

9.3.2 Off-policy Mountain Car

In the original Mountain Car environment, the agent attempts to drive an underpowered car out of a valley (Moore, 1990).[9] In the off-policy version, the agent learns from experience generated by a fixed behaviour policy, in this case the uniform random policy. The observations were tile coded using 8 tilings of 4×4 tiles plus a bias unit. ACE-ideal was omitted from the Mountain Car experiments due to computational constraints; the uniform random behaviour policy rarely completed an episode, which resulted in extremely long trajectories that prevented ACE-ideal from running in a reasonable amount of time.

Figure 8 contains the results for the Off-policy Mountain Car environment. The left-hand column shows learning curves, while the right-hand column shows sensitivity analyses for the actor's step size. The first four plots (figures 8a, 8b, 8c, and 8d) show results when using ETD as the critic, while the last four plots (figures 8e, 8f, 8g, and 8h) show results for a TDRC critic. The first and third rows show results using the excursions objective function, and the second and fourth rows show results for the episodic objective function.

Across all choices of critic and objective function, algorithms trained using the episodic interest function (dashed lines) performed much worse than their counterparts trained using the uniform interest function (solid lines). This finding is consistent with the results from Puddle World, although much more pronounced. In this specific experiment, the extremely long trajectories generated by the uniform random behaviour policy caused the emphatic trace (and hence the policy updates) of the algorithms that used episodic interest to quickly shrink to the point of irrelevance, whereas in the Puddle World experiment the behaviour policy encounters the goal state more often, limiting trajectory length and shrinking the updates more slowly. Nevertheless, using the direct method to estimate the emphatic weightings allowed ACE to improve the performance of the policy in some cases, whereas using the emphatic trace prevented the policy from meaningfully improving.

Again, the choice of critic played a role in the performance of all the algorithms to varying degrees. Using a TDRC critic instead of an ETD critic improved the performance of each of the algorithms, with OffPAC seeing the greatest improvement, followed by ACE-trace. ACE-direct again performed well regardless of the choice of critic. Using a TDRC critic improved the sensitivity of each of the algorithms to the actor's step size, allowing a wider range of step sizes to perform well (compare figures 8b and 8f, and 8d and 8h).

Comparing ACE-direct (red) to ACE-trace (orange) reveals several advantages of using the direct method to estimate the emphatic weightings. The direct method allowed ACE to learn faster, with less variance, and find a better-performing target policy by the end of the experiment, regardless of the choice of critic or objective function used for evaluation. In addition, the direct method allowed a wider range of actor step sizes to perform well, with performance increasing and decreasing more smoothly and predictably than when using the emphatic trace. Finally, the choice of critic had little effect on the performance of ACE-direct, whereas ACE-trace performed better with a TDRC critic than with an ETD critic. Overall, ACE with the direct method performed better than or equal to OffPAC (green) and ACE-trace for all critics and objective functions, but was less sensitive to the actor step size and choice of critic than OffPAC.

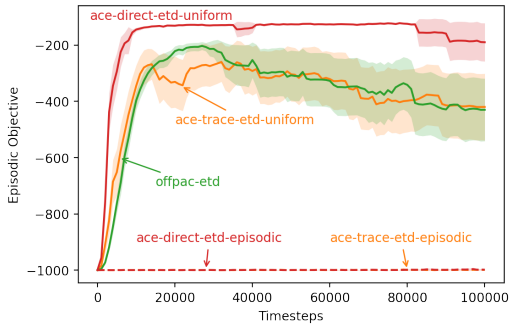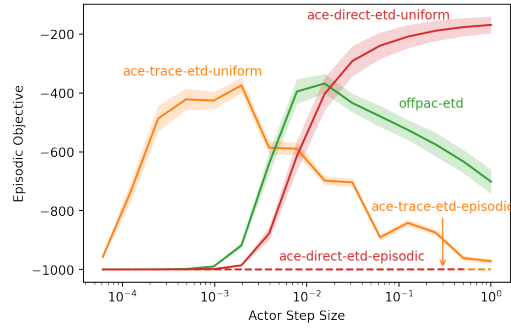9. Please see Degris et al. (2012b) for a picture of the Mountain Car environment.
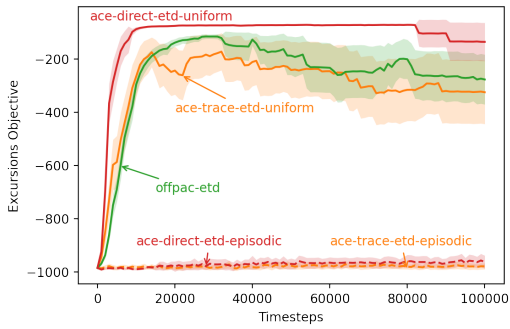
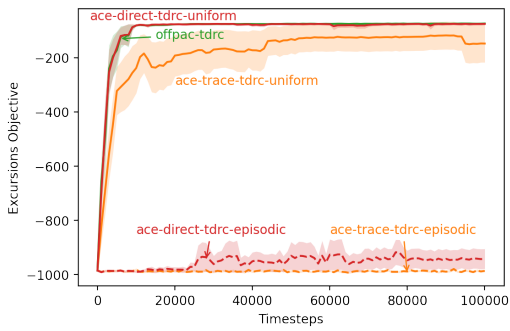(a) Excursions learning curves (ETD)
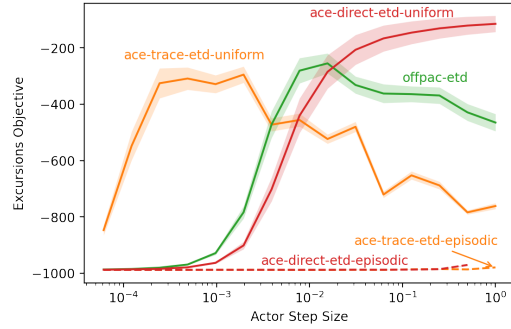
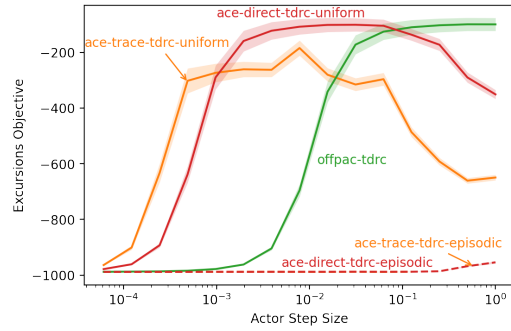(b) Excursions step-size sensitivity (ETD)
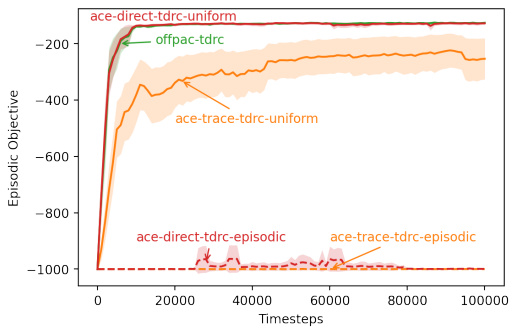
(c) Episodic learning curves (ETD)
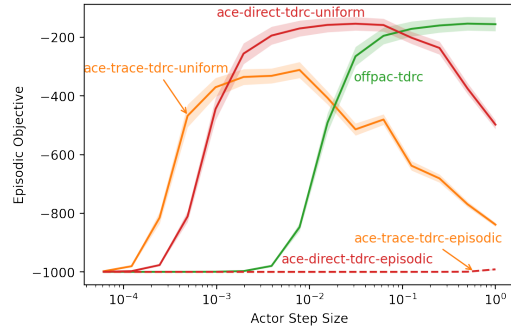
(d) Episodic step-size sensitivity (ETD)

(e) Excursions learning curves (TDRC)

(f) Excursions step-size sensitivity (TDRC)

(g) Episodic learning curves (TDRC)

(h) Episodic step-size sensitivity (TDRC)

Figure 8: Results on Off-policy Mountain Car. Shaded regions are 95% confidence intervals.

9.4 The Virtual Office Environment

The classic control environments considered in the previous section are well-known environments for testing general off-policy algorithms, but were not designed to probe the weaknesses of the specific algorithms being studied. Hence, both semi-gradient updates and the direct method of estimating emphatic weightings perform well, and the tradeoffs made in the design of each algorithm are not obvious. In this section, we introduce an environment designed to highlight issues with both semi-gradient updates and the direct method of estimating emphatic weightings.

The environment is based on Dung et al. (2007)'s Virtual Office, a grid world consisting of a hallway and two cubicles that are identical except for the goal locations. To illustrate the problem with semi-gradient updates, two key properties of the counterexample from Section 8.1 must be incorporated: the optimal actions in the two identical rooms must be different, and the behaviour policy must visit the suboptimal room more often. To accomplish the first goal, we introduced hidden goal states in the north-east and south-east corners of the two rooms, and assigned rewards of 1 and 0 respectively to the goal states of the north-east room and rewards of 0 and .5 respectively to the goal states of the south-east room. To encourage the behaviour policy to visit the suboptimal room (i.e., the south-east room) more often, we fixed the starting state to the left-most, middle-most state and used a behaviour policy that favoured the South and East actions, taking the North, East, South, and West actions with probabilities .2, .4, .3, and .1 respectively.

While Dung et al. (2007)'s Virtual Office contains some partial observability in the form of the two identical rooms, to fully illustrate the tradeoff made by the Markov assumption used in the derivation of the direct method, we limited the agent's observations to a 3×3 grid in front of the agent. The RGB colour values of each square in the agent's view were used as observations, with the agent unable to see through walls or perceive the goal states.

The final environment was implemented using Minigrid (Chevalier-Boisvert et al., 2018), and is depicted in Figure 9. The agent (red triangle) must navigate to one of the terminal states (the north-east and south-east squares of the green rooms) to obtain the associated reward. The agent observes the RGB colour values of the 3×3 grid of squares in front of it (the lighter blue squares), and selects the North, East, South, or West action in response.

For this experiment we followed the methodology detailed in the previous section, with minor changes. First, we restricted the critic trace decay rate to be 0, as eligibility traces can alleviate some of the issues caused by partial observability (Loch and Singh, 1998). Second, we ran each combination of parameters for 30 runs of 200,000 time steps each. The learned policies were saved every 5,000 time steps and evaluated 50 times using both the episodic and excursions objective functions. The best-performing parameter settings were then re-run 100 times and averaged.

Figure 10 contains the results of our experiment on the Virtual Office environment. The left-hand column shows learning curves, while the right-hand column shows sensitivity analyses for the actor's step size. The first four plots (figures 10a, 10b, 10c, and 10d) show results when using ETD as the critic, while the last four plots (figures 10e, 10f, 10g, and 10h) show results for a TDRC critic. The first and third rows show results for the excursions objective function, and the second and fourth rows show the episodic objective function.
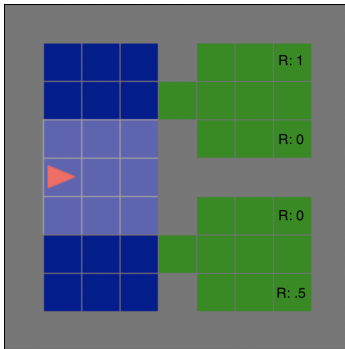
Figure 9: The Virtual Office environment.

Interestingly, algorithms that used the episodic interest function (dashed lines) performed better than or equal to their counterparts that used the uniform interest function (solid lines) for all combinations of objective function and critic, unlike the previous experiments on Mountain Car and Puddle World. In addition, the algorithms that used the episodic interest function worked well for a larger range of step sizes, making it easier to find a value that performs well. This could be due to the partially observable nature of the environment; many of the agent's observations are similar, which could cause the emphatic weightings with a uniform interest function to quickly become very large, necessitating a very small step size to compensate.

As in the previous experiments, the choice of critic played an important role in the performance of all the algorithms, although this time with mixed results. Using a TDRC critic instead of an ETD critic improved the performance of OffPAC and ACE-direct with uniform interest, but had very little impact on ACE-trace with uniform interest and ACE-direct with episodic interest, and actually reduced the performance of ACE-trace with episodic interest (compare figures 10a and 10e, and 10c and 10g). Similarly, using a TDRC critic had mixed effects on the sensitivity of each of the algorithms to the actor's step size. It allowed a wider range of step sizes to perform well for some algorithms, but also led to a narrower range of well-performing step sizes for some algorithms (compare figures 10b and 10f, and 10d and 10h).

ACE-direct with the episodic interest function performed better than or equal to the other methods across most combinations of critic and objective function, with the possible exception of the episodic objective function. With a TDRC critic, OffPAC learned more quickly than the other methods before eventually being surpassed by both ACE-direct variants. However, OffPAC was able to close the gap near the end of the experiment, with the final learned policy of OffPAC performing comparably to the ACE-direct variants. With an ETD critic, ACE-trace with episodic interest performed surprisingly well, with the final policy possibly outperforming ACE-direct with episodic interest. However, the variance of the policy learned by ACE-trace was much larger than the other methods, rendering the difference in performance not statistically significant. Conversely, ACE-direct with episodic interest learned a low-variance policy on each of the settings tested, performing as well as or better than the other algorithms.
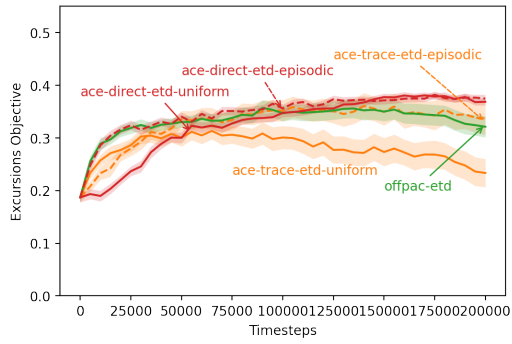
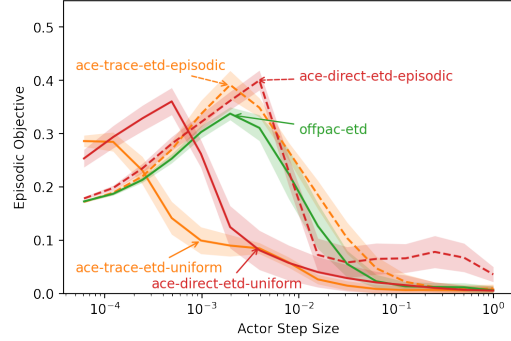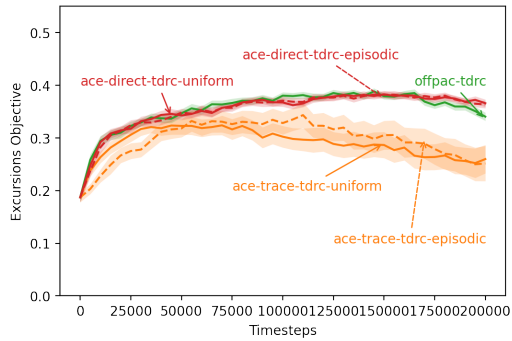(a) Excursions learning curves (ETD)
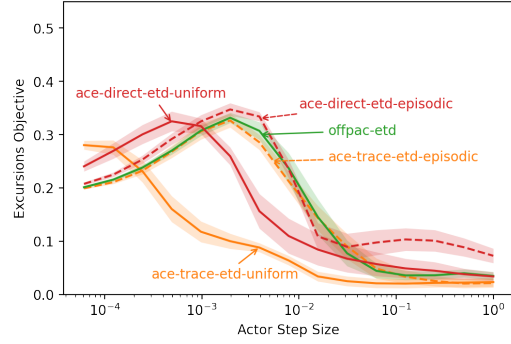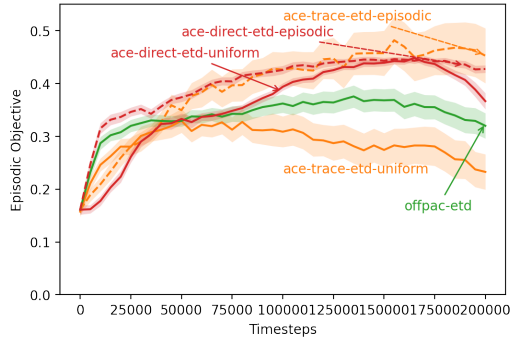
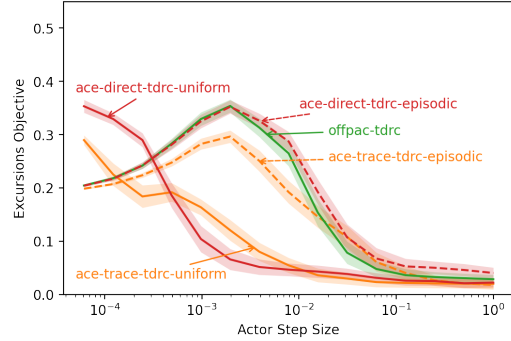(b) Excursions step-size sensitivity (ETD)
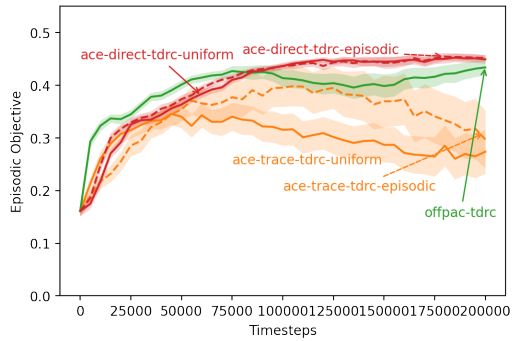
(c) Episodic learning curves (ETD)

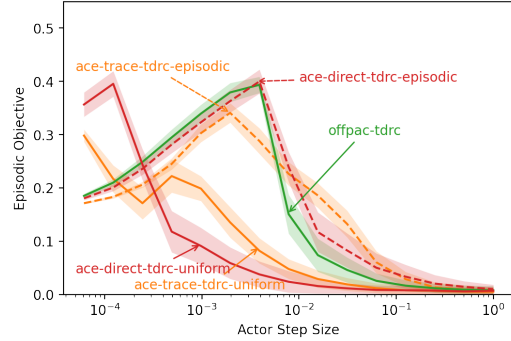(d) Episodic step-size sensitivity (ETD)

(e) Excursions learning curves (TDRC)

(f) Excursions step-size sensitivity (TDRC)

(g) Episodic learning curves (TDRC)

(h) Episodic step-size sensitivity (TDRC)

Figure 10: Results on the Virtual Office. Shaded regions are 95% confidence intervals.

9.5 Summary

In this section we investigated several key properties of the ACE algorithm empirically. First we studied how the parameter η trades off bias and variance in the counterexample from Section 8.1 and a continuous action version, finding that even small values of η (i.e., closer to semi-gradient updates) greatly improved the quality of the final learned policy, both when using the true value function and when using a learned critic.

Next we tested the ability of the emphatic trace to accurately estimate the emphatic weightings on an extended version of the counterexample, showing that while higher values of η improved performance, a significant gap remained between ACE with $\eta = 1$ and ACE with the true emphatic weightings.

Then we moved to two classic control domains and tested several variants of ACE to determine how the choice of estimator for the emphatic weightings (and associated bias and variance properties) and the choice of critic affects the learning process. We found that the emphatic trace's extreme variance greatly reduced the performance of ACE—with its bias playing somewhat less of a role—and replacing the emphatic trace with the direct method from Section 5.2.2 resulted in ACE performing as well as or better than the other variants. The TDRC critic was the better critic overall, but ACE with the direct method of estimating emphatic weightings was insensitive to the choice of critic and performed well regardless of whether ETD or TDRC was used.

Finally, we compared the performance of several variants of ACE on a new environment designed to highlight the weaknesses of each of the variants, finding that the episodic variant of ACE with the direct method of estimating emphatic weightings performed as well as or better than the other variants in almost all the situations tested, suggesting it should be the default method of estimating the emphatic weightings.

10. Conclusion

In this paper we introduced a generalized objective for learning policies and proved the off-policy policy gradient theorem using emphatic weightings. Using this theorem, we derived an off-policy actor-critic algorithm that follows the gradient of the objective function, as opposed to previous methods like OffPAC and DPG that follow an approximate semi-gradient. We designed a simple MDP to highlight that stationary points for semi-gradients can be highly suboptimal, with semi-gradient updates moving away from the optimal solution. We also show, though, that the semi-gradient methods can produce reasonable solutions, either because of sufficiently rich policy parameterizations or because in some cases they can be seen as using the true gradient update with a different state weighting.

We leverage these insights to improve the practicality of the method empirically. We design Actor-Critic with Emphatic Weightings (ACE) to have a tuneable parameter η that sweeps between the full gradient (at $\eta = 1$) and the semi-gradient (at $\eta = 0$), often obtaining the best performance with a relatively small but non-zero η (typically $\eta = 0.1$). We additionally reduce variance by directly estimating emphatic weightings with function approximation rather than using the emphatic trace. Empirically, the direct method of estimating the emphatic weightings performed better than the emphatic trace across all objective functions and environments tested. In addition, ACE with the direct method was less sensitive to the actor step size and choice of critic than OffPAC. Overall, ACE with the

direct method performed better than or equal to OffPAC in all situations, suggesting that incorporating (low-variance) state reweightings might be a reasonable direction for future policy gradient methods.

Acknowledgments

Appendix A. Proof of Deterministic Off-Policy Policy Gradient Theorem

A.1 Assumptions

We make the following assumptions on the MDP:

Assumption 1 $P(s'|s, a), r(s, a, s'), \gamma(s, a, s'), \pi(s; \boldsymbol{\theta})$ *and their derivatives are continuous in all variables* $s, a, s', \boldsymbol{\theta}$.

Assumption 2 $\mathcal{S}$ *is a compact set in* $\mathbb{R}^d$, *and* $\mathcal{A}$ *is a compact set in* $\mathbb{R}$.

Assumption 3 *The policy* π *and discount* γ *are such that the inverse kernel of* $\delta(s, s') - \mathbf{P}_{\pi,\gamma}(s, s')$ *exists, where* $\mathbf{P}_{\pi,\gamma}(s, s') = \int_{\mathcal{A}} \pi(a|s)\gamma(s, a, s')P(s'|s, a)da$.

Under Assumption 1, $v_{\pi_{\boldsymbol{\theta}}}(s)$ and $\frac{\partial v_{\pi_{\boldsymbol{\theta}}}(s)}{\partial \boldsymbol{\theta}}$ are continuous functions of $\boldsymbol{\theta}$ and s. Together, Assumptions 1 and 2 imply that $\left\|\frac{\partial v_{\pi_{\boldsymbol{\theta}}}(s)}{\partial \boldsymbol{\theta}}\right\|$, $\left\|\frac{\partial \pi(s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\right\|$, and $\left\|\frac{\partial q_{\pi_{\boldsymbol{\theta}}}(s,a)}{\partial a}\Big|_{a=\pi(s;\boldsymbol{\theta})}\right\|$ are bounded functions of s, which allows us to switch the order of integration and differentiation, and the order of multiple integrations.

A.2 Proof of Theorem 3

Proof We start by deriving a recursive form for the gradient of the value function with respect to the policy parameters:

$$\frac{\partial v_{\pi}(s)}{\partial \boldsymbol{\theta}} = \frac{\partial}{\partial \boldsymbol{\theta}} q_{\pi}(s, \pi(s; \boldsymbol{\theta}))$$

$$= \frac{\partial}{\partial \boldsymbol{\theta}} \int_{\mathcal{S}} \mathrm{P}(s'|s, \pi(s; \boldsymbol{\theta}))\Big(r(s, \pi(s; \boldsymbol{\theta}), s') + \gamma(s, \pi(s; \boldsymbol{\theta}), s')v_{\pi}(s')\Big) \mathrm{d}s'$$

$$= \int_{\mathcal{S}} \frac{\partial}{\partial \boldsymbol{\theta}} \Big(\mathrm{P}(s'|s, \pi(s; \boldsymbol{\theta}))\Big(r(s, \pi(s; \boldsymbol{\theta}), s') + \gamma(s, \pi(s; \boldsymbol{\theta}), s')v_{\pi}(s')\Big)\Big) \mathrm{d}s', \quad (21)$$

where in Equation 21 we used the Leibniz integral rule to switch the order of integration and differentiation. We proceed with the derivation using the product rule:

$$= \int_{\mathcal{S}} \frac{\partial}{\partial \boldsymbol{\theta}} \mathrm{P}(s'|s, \pi(s; \boldsymbol{\theta}))\Big(r(s, \pi(s; \boldsymbol{\theta}), s') + \gamma(s, \pi(s; \boldsymbol{\theta}), s')v_{\pi}(s')\Big)$$

$$+ \operatorname{P}(s'|s, \pi(s; \boldsymbol{\theta})) \frac{\partial}{\partial \boldsymbol{\theta}} \Big(r(s, \pi(s; \boldsymbol{\theta}), s') + \gamma(s, \pi(s; \boldsymbol{\theta}), s') v_\pi(s') \Big) \operatorname{d}s'$$

$$= \int_{\mathbb{S}} \frac{\partial \pi(s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \left. \frac{\partial \operatorname{P}(s'|s, a)}{\partial a} \right|_{a=\pi(s;\boldsymbol{\theta})} \Big(r(s, \pi(s; \boldsymbol{\theta}), s') + \gamma(s, \pi(s; \boldsymbol{\theta}), s') v_\pi(s') \Big) + \operatorname{P}(s'|s, \pi(s; \boldsymbol{\theta}))$$

$$\left(\frac{\partial}{\partial \boldsymbol{\theta}} r(s, \pi(s; \boldsymbol{\theta}), s') + \frac{\partial}{\partial \boldsymbol{\theta}} \gamma(s, \pi(s; \boldsymbol{\theta}), s') v_\pi(s') + \gamma(s, \pi(s; \boldsymbol{\theta}), s') \frac{\partial}{\partial \boldsymbol{\theta}} v_\pi(s') \right) \operatorname{d}s'$$

$$= \int_{\mathbb{S}} \frac{\partial \pi(s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \left. \frac{\partial \operatorname{P}(s'|s, a)}{\partial a} \right|_{a=\pi(s;\boldsymbol{\theta})} \Big(r(s, \pi(s; \boldsymbol{\theta}), s') + \gamma(s, \pi(s; \boldsymbol{\theta}), s') v_\pi(s') \Big)$$

$$+ \operatorname{P}(s'|s, \pi(s; \boldsymbol{\theta})) \left(\frac{\partial \pi(s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \left. \frac{\partial r(s, a, s')}{\partial a} \right|_{a=\pi(s;\boldsymbol{\theta})} + \frac{\partial \pi(s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \left. \frac{\partial \gamma(s, a, s')}{\partial a} \right|_{a=\pi(s;\boldsymbol{\theta})} v_\pi(s') \right) \operatorname{d}s'$$

$$+ \int_{\mathbb{S}} \operatorname{P}(s'|s, \pi(s; \boldsymbol{\theta})) \gamma(s, \pi(s; \boldsymbol{\theta}), s') \frac{\partial}{\partial \boldsymbol{\theta}} v_\pi(s') \operatorname{d}s'$$

$$= \int_{\mathbb{S}} \frac{\partial \pi(s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \left(\left. \frac{\partial \operatorname{P}(s'|s, a)}{\partial a} \right|_{a=\pi(s;\boldsymbol{\theta})} \Big(r(s, \pi(s; \boldsymbol{\theta}), s') + \gamma(s, \pi(s; \boldsymbol{\theta}), s') v_\pi(s') \Big) \right.$$

$$\left. + \operatorname{P}(s'|s, \pi(s; \boldsymbol{\theta})) \left. \frac{\partial}{\partial a} \Big(r(s, a, s') + \gamma(s, a, s') v_\pi(s') \Big) \right|_{a=\pi(s;\boldsymbol{\theta})} \right) \operatorname{d}s'$$

$$+ \int_{\mathbb{S}} \operatorname{P}(s'|s, \pi(s; \boldsymbol{\theta})) \gamma(s, \pi(s; \boldsymbol{\theta}), s') \frac{\partial}{\partial \boldsymbol{\theta}} v_\pi(s') \operatorname{d}s'$$

$$= \frac{\partial \pi(s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \int_{\mathbb{S}} \left. \frac{\partial}{\partial a} \Big(\operatorname{P}(s'|s, a) \big(r(s, a, s') + \gamma(s, a, s') v_\pi(s') \big) \Big) \right|_{a=\pi(s;\boldsymbol{\theta})} \operatorname{d}s'$$

$$+ \int_{\mathbb{S}} \operatorname{P}(s'|s, \pi(s; \boldsymbol{\theta})) \gamma(s, \pi(s; \boldsymbol{\theta}), s') \frac{\partial v_\pi(s')}{\partial \boldsymbol{\theta}} \operatorname{d}s'$$

$$= \frac{\partial \pi(s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \left. \frac{\partial q_\pi(s, a)}{\partial a} \right|_{a=\pi(s;\boldsymbol{\theta})} + \int_{\mathbb{S}} \operatorname{P}(s'|s, \pi(s; \boldsymbol{\theta})) \gamma(s, \pi(s; \boldsymbol{\theta}), s') \frac{\partial v_\pi(s')}{\partial \boldsymbol{\theta}} \operatorname{d}s'. \tag{22}$$

For simplicity of notation, we will write Equation 22 as

$$\frac{\partial v_\pi(s)}{\partial \boldsymbol{\theta}} = g(s) + \int_{\mathbb{S}} \mathbf{P}_{\pi, \gamma}(s, s') \frac{\partial v_\pi(s')}{\partial \boldsymbol{\theta}} \operatorname{d}s', \tag{23}$$

since $\operatorname{P}(s'|s, \pi(s; \boldsymbol{\theta})) \gamma(s, \pi(s; \boldsymbol{\theta}), s')$ is a function of s and s' for a fixed deterministic policy. Then we can write $\frac{\partial v_\pi(s)}{\partial \boldsymbol{\theta}}$ as an integral transform using the delta function:

$$\frac{\partial v_\pi(s)}{\partial \boldsymbol{\theta}} = \int_{\mathbb{S}} \delta(s, s') \frac{\partial v_\pi(s')}{\partial \boldsymbol{\theta}} \operatorname{d}s'. \tag{24}$$

Plugging Equation 24 into the left-hand side of Equation 23, we obtain:

$$\int_{\mathbb{S}} \delta(s, s') \frac{\partial v_\pi(s')}{\partial \boldsymbol{\theta}} \operatorname{d}s' = g(s) + \int_{\mathbb{S}} \mathbf{P}_{\pi, \gamma}(s, s') \frac{\partial v_\pi(s')}{\partial \boldsymbol{\theta}} \operatorname{d}s'$$

$$\implies \int_{\mathbb{S}} \Big(\delta(s, s') - \mathbf{P}_{\pi, \gamma}(s, s') \Big) \frac{\partial v_\pi(s')}{\partial \boldsymbol{\theta}} \operatorname{d}s' = g(s)$$

$$\implies \frac{\partial v_\pi(s)}{\partial \boldsymbol{\theta}} = \int_{\mathbb{S}} k(s, s') g(s') \operatorname{d}s', \tag{25}$$

where $k(s, s')$ is the inverse kernel of $\delta(s, s') - \mathbf{P}_{\pi,\gamma}(s, s')$. Now, using the continuous version of the weighted excursions objective defined in Equation 6, we have:

$$
\begin{aligned}
\frac{\partial J_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} &= \frac{\partial}{\partial \boldsymbol{\theta}} \int_{\mathbb{S}} d_\mu(s) i(s) v_\pi(s) \, \mathrm{d}s \\
&= \int_{\mathbb{S}} \frac{\partial v_\pi(s)}{\partial \boldsymbol{\theta}} d_\mu(s) i(s) \, \mathrm{d}s \\
&= \int_{\mathbb{S}} \int_{\mathbb{S}} k(s, s') g(s') \, \mathrm{d}s' \, d_\mu(s) i(s) \, \mathrm{d}s \\
&= \int_{\mathbb{S}} \int_{\mathbb{S}} k(s, s') d_\mu(s) i(s) \, \mathrm{d}s \, g(s') \, \mathrm{d}s',
\end{aligned}
\tag{26}
$$

where in Equation 26 we used Fubini's theorem to switch the order of integration.

Now, we convert the recursive definition of emphatic weightings for deterministic policies over continuous state-action spaces into a non-recursive form. Recall the definition:

$$
m(s') = d_\mu(s') i(s') + \int_{\mathbb{S}} \mathbf{P}_{\pi,\gamma}(s, s') m(s) \, \mathrm{d}s.
\tag{27}
$$

Again, we can write $m(s')$ as an integral transform using the delta function:

$$
m(s') = \int_{\mathbb{S}} \delta(s, s') m(s) \, \mathrm{d}s.
\tag{28}
$$

Plugging Equation 28 into the left-hand side of Equation 27, we obtain:

$$
\begin{aligned}
\int_{\mathbb{S}} \delta(s, s') m(s) \, \mathrm{d}s &= d_\mu(s') i(s') + \int_{\mathbb{S}} \mathbf{P}_{\pi,\gamma}(s, s') m(s) \, \mathrm{d}s \\
\implies \int_{\mathbb{S}} (\delta(s, s') - \mathbf{P}_{\pi,\gamma}(s, s')) m(s) \, \mathrm{d}s &= d_\mu(s') i(s') \\
\implies m(s') &= \int_{\mathbb{S}} k(s, s') d_\mu(s) i(s) \, \mathrm{d}s,
\end{aligned}
\tag{29}
$$

where $k(s, s')$ is again the inverse kernel of $\delta(s, s') - \mathbf{P}_{\pi,\gamma}(s, s')$. Plugging Equation 29 into Equation 26 yields

$$
\begin{aligned}
\frac{\partial J_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} &= \int_{\mathbb{S}} m(s') g(s') \, \mathrm{d}s' \\
&= \int_{\mathbb{S}} m(s) \frac{\partial \pi(s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \left. \frac{\partial q_\pi(s, a)}{\partial a} \right|_{a=\pi(s;\boldsymbol{\theta})} \mathrm{d}s.
\end{aligned}
$$

$\blacksquare$

Appendix B. Continuous State Off-Policy Policy Gradient Theorem

Given the Deterministic Off-Policy Policy Gradient Theorem in Appendix A, we can now provide a proof for the continuous-state version of the Off-Policy Policy Gradient Theorem provided in Theorem 2.

Proof The gradient of the objective in the continuous-states case is

$$\frac{\partial J_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \frac{\partial \int_{\mathcal{S}} i(s) v_\pi(s) \, \mathrm{d}s}{\partial \boldsymbol{\theta}} = \int_{\mathcal{S}} i(s) \frac{\partial v_\pi(s)}{\partial \boldsymbol{\theta}} \, \mathrm{d}s. \tag{30}$$

Therefore, again to compute the gradient of J_μ we need to compute the gradient of the value function with respect to the policy parameters. A recursive form of the gradient of the value function can be derived, as we show below. Before starting, for simplicity of notation, we will again use

$$\mathbf{g}(s) = \sum_a \frac{\partial \pi(s, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, a),$$

where $\mathbf{g} : \mathcal{S} \to \mathbb{R}^d$. Now let us compute the gradient of the value function:

$$\begin{aligned}
\frac{\partial v_\pi(s)}{\partial \boldsymbol{\theta}} &= \frac{\partial}{\partial \boldsymbol{\theta}} \sum_a \pi(s, a; \boldsymbol{\theta}) q_\pi(s, a) \\
&= \sum_a \frac{\partial \pi(s, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, a) + \sum_a \pi(s, a; \boldsymbol{\theta}) \frac{\partial q_\pi(s, a)}{\partial \boldsymbol{\theta}} \\
&= \mathbf{g}(s) + \sum_a \pi(s, a; \boldsymbol{\theta}) \frac{\partial \int_{\mathcal{S}} \mathrm{P}(s'|s, a)(r(s, a, s') + \gamma(s, a, s') v_\pi(s')) \, \mathrm{d}s'}{\partial \boldsymbol{\theta}} \\
&= \mathbf{g}(s) + \sum_a \pi(s, a; \boldsymbol{\theta}) \int_{\mathcal{S}} \mathrm{P}(s'|s, a) \gamma(s, a, s') \frac{\partial v_\pi(s')}{\partial \boldsymbol{\theta}} \, \mathrm{d}s' \\
&= \mathbf{g}(s) + \int_{\mathcal{S}} \sum_a \pi(s, a; \boldsymbol{\theta}) \mathrm{P}(s'|s, a) \gamma(s, a, s') \frac{\partial v_\pi(s')}{\partial \boldsymbol{\theta}} \, \mathrm{d}s'. \tag{31}
\end{aligned}$$

For simplicity of notation we will write Equation 31 as

$$\frac{\partial v_\pi(s)}{\partial \boldsymbol{\theta}} = \mathbf{g}(s) + \int_{\mathcal{S}} \mathbf{P}_{\pi, \gamma}(s, s') \frac{\partial v_\pi(s')}{\partial \boldsymbol{\theta}} \, \mathrm{d}s', \tag{32}$$

where $\mathbf{P}_{\pi, \gamma}(s, s') = \sum_a \pi(s, a; \boldsymbol{\theta}) \mathrm{P}(s'|s, a) \gamma(s, a, s')$. Again, as in the deterministic off-policy policy gradient theorem, we can write $\frac{\partial v_\pi(s)}{\partial \boldsymbol{\theta}}$ as an integral transform using the delta function:

$$\frac{\partial v_\pi(s)}{\partial \boldsymbol{\theta}} = \int_{\mathcal{S}} \delta(s, s') \frac{\partial v_\pi(s')}{\partial \boldsymbol{\theta}} \, \mathrm{d}s'. \tag{33}$$

Plugging Equation 33 into the left-hand side of Equation 32, we obtain:

$$\begin{aligned}
\int_{\mathcal{S}} \delta(s, s') \frac{\partial v_\pi(s')}{\partial \boldsymbol{\theta}} \, \mathrm{d}s' &= \mathbf{g}(s) + \int_{\mathcal{S}} \mathbf{P}_{\pi, \gamma}(s, s') \frac{\partial v_\pi(s')}{\partial \boldsymbol{\theta}} \, \mathrm{d}s' \\
\implies \int_{\mathcal{S}} \left(\delta(s, s') - \mathbf{P}_{\pi, \gamma}(s, s') \right) \frac{\partial v_\pi(s')}{\partial \boldsymbol{\theta}} \, \mathrm{d}s' &= \mathbf{g}(s) \\
\implies \frac{\partial v_\pi(s)}{\partial \boldsymbol{\theta}} &= \int_{\mathcal{S}} k(s, s') \mathbf{g}(s') \, \mathrm{d}s'. \tag{34}
\end{aligned}$$

Plugging this gradient from Equation 34 back into Equation 30, we obtain

$$
\begin{aligned}
\frac{\partial J_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} &= \int_\mathbb{S} i(s) \frac{\partial v_\pi(s)}{\partial \boldsymbol{\theta}} \, \mathrm{d}s \\
&= \int_\mathbb{S} i(s) \int_\mathbb{S} k(s, s') \mathbf{g}(s') \, \mathrm{d}s' \, \mathrm{d}s \qquad (35) \\
&= \int_\mathbb{S} \int_\mathbb{S} k(s, s') i(s) \, \mathrm{d}s \, \mathbf{g}(s') \, \mathrm{d}s' \qquad (36) \\
&= \int_\mathbb{S} m(s') \mathbf{g}(s') \, \mathrm{d}s' \\
&= \int_\mathbb{S} m(s) \mathbf{g}(s) \, \mathrm{d}s,
\end{aligned}
$$

where in Equation 35 we used Fubini's theorem to switch the order of integration, and in Equation 36 we use the non-recursive version of emphasis as shown in Equation 29. ■

Appendix C. Standard Approaches for Sampling the Gradient from a State

In this section we overview the standard approach to sample the gradient from a given state, that is used in both ACE and OffPAC. This section reiterates known results, but is included here for completeness.

C.1 Sampling the Gradient for a Given State

For a given state, we can use the same strategies to sample the gradient as OffPAC, and many other actor-critic algorithms. We can use the log-likelihood approach, and incorporate baselines. For continuous actions, we can also use other strategies like the reparameterization trick. For concreteness, we explicitly explain how we sample this gradient here, using a value baseline and the log-likelihood approach, but emphasize that this can be replaced with other choices.

Under finite states, the simplest approach to compute the gradient for a given state is to use what is sometimes called the *all-actions* gradient

$$
\sum_a \frac{\partial \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, a).
$$

If there are many actions or if the action space is continuous, this gradient is not an appropriate choice. Instead, we can obtain a sample estimate of this gradient using the log-likelihood trick:

$$
\sum_a \frac{\partial \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, a) = \sum_a \pi(a|s; \boldsymbol{\theta}) \frac{\partial \log \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, a),
$$

which follows from the fact that the derivative of $\log y$ is $\frac{1}{y}$. Above we called this log likelihood form $g(s, \boldsymbol{\theta})$. Now we can use one action sampled according to π to obtain

an unbiased estimate of the all-actions gradient. In the off-policy setting, however, we do not use π to select actions. So, additionally, we can incorporate importance sampling $\rho(s, a; \boldsymbol{\theta}) \stackrel{\text{def}}{=} \frac{\pi(a|s;\boldsymbol{\theta})}{\mu(a|s)}$ to adjust for the fact that the actions are taken according to behaviour policy μ:

$$\sum_a \frac{\partial \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, a) = \sum_a \mu(a|s) \frac{1}{\mu(a|s)} \pi(a|s;\boldsymbol{\theta}) \frac{\partial \log \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, a)$$

$$= \sum_a \mu(a|s) \rho(s, a; \boldsymbol{\theta}) \frac{\partial \log \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, a).$$

We can obtain an unbiased sample of the gradient by sampling an action $A \sim \mu(s, \cdot)$ and using $g(A) \stackrel{\text{def}}{=} \rho(s, A; \boldsymbol{\theta}) \frac{\partial \log \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, A)$.

The estimated gradient has more variance when we only use a sampled action rather than all actions. The standard strategy to reduce this variance is to subtract a baseline. The baseline is a function of state, such as an estimate of the value function $v(s)$, with modified update

$$\rho(s, a; \boldsymbol{\theta}) \frac{\partial \log \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} [q_\pi(s, a) - v(s)].$$

It is straightforward to show that the expected value of this update is the same, i.e. that

$$\sum_a \mu(a|s) \rho(s, a; \boldsymbol{\theta}) \frac{\partial \log \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} [q_\pi(s, a) - v(s)] = \sum_a \mu(a|s) \rho(s, a; \boldsymbol{\theta}) \frac{\partial \log \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} q_\pi(s, a).$$

Note that this is not the minimal variance baseline (Dick, 2015), but it nonetheless is one of the most commonly chosen in practice due to its efficacy and simplicity.

C.2 Estimating the Advantage

In practice, it is difficult to obtain $q_\pi(s, a)$ to compute this gradient exactly. Instead, we obtain (a likely biased) sample of the advantage, $\delta_t \approx q_\pi(S_t, A_t) - v(s_t)$. An unbiased but high variance estimate uses a sample of the return G_t from (s_t, a_t), and uses $\delta_t = G_t - v(s_t)$. Then $\mathbb{E}[\delta_t | S_t = s, A_t = a] = q_\pi(s, a) - v(s)$. On the other extreme, we can directly approximate the action-values, $\hat{q}(s, a)$. In-between, we can use n-step returns, or λ-returns, to balance between using observed rewards and value estimates.

The simplest choice is to use a 1-step return: $r(s, a) + \gamma(s, a, S_{t+1}) v(S_{t+1})$ is used as an approximation of $q_\pi(s, a)$. The value function v is typically called the critic, and plays both the role of estimating the return as well as that of a baseline. The advantage is set to $\delta_t = R_{t+1} + \gamma_{t+1} v(S_{t+1}) - v(S_t)$, which is simply the TD error.

More generally, however, we can also use multi-step returns. For an n-step return, with data sampled on-policy, with $\gamma_{t+1:t+n} \stackrel{\text{def}}{=} \prod_{j=1}^n \gamma_{t+j}$ as short-hand for the product of discounts, we would use

$$\delta_t = R_{t+1} + \gamma_{t+1} R_{t+2} + \ldots \gamma_{t+1:t+n-1} R_{t+n} + \gamma_{t+1:t+n} v(S_{t+n}) - v(S_t).$$

With off-policy data, we would need to incorporate importance sampling ratios

$$\delta_t = R_{t+1} + \rho_{t+1} \gamma_{t+1} R_{t+2} + \ldots \rho_{t+1:t+n-1} \gamma_{t+1:t+n-1} R_{t+n} + \rho_{t+1:t+n} \gamma_{t+1:t+n} v(S_{t+n}) - v(S_t),$$

where $\rho_{t+1:t+n} \stackrel{\text{def}}{=} \prod_{j=1}^{n} \rho_{t+j}$. For sufficiently large n, we recover the sampled return and so obtain an unbiased—but likely high-variance—estimate of the advantage. An interim n likely provides a reasonable choice between bias and variance, between $n = 0$ with a direct estimation of $q_\pi(s, a)$, and large n that gives an unbiased sample of the return.

We can also average across multiple n, and obtain λ-returns (Sutton and Barto, 2018). For larger λ, more weight is put on the n-step returns with larger n, and for $\lambda = 1$ we once again recover an unbiased sample of the return. Such an approach was used in OffPAC, which requires storing traces of gradients of policy parameters (see Algorithm 1 in Degris et al., 2012b). Note that this λ-return is used for the policy update. The update to the value estimate (critic) itself can use a different λ, as it stores a separate trace for the gradient of the value function. The update with eligibility traces is only sound under linear function approximation for the values and linear function approximation for action-preferences. In this work, therefore, we opt for the simplest choice: $n = 1$ with $\delta_t = R_{t+1} + \gamma_{t+1}v(S_{t+1}) - v(S_t)$.

Finally, we need to estimate the values themselves. Any value approximation algorithm can be used to estimate v, such as TD, gradient TD or even emphatic TD. Given we already compute the emphasis weighting, it is straightforward to use emphatic TD. We investigate both a gradient TD critic and an emphatic TD critic in the experiments.

Appendix D. Proof of Theorem 8

D.1 Entropy-regularized OPPG theorem with discrete states and actions

Proof The proof for this theorem is mostly similar to the one for Theorem 2. First, since the interest function does not depend on the parameters,

$$\frac{\partial \tilde{J}_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \frac{\partial \sum_{s \in \mathcal{S}} i(s)\tilde{v}_\pi(s)}{\partial \boldsymbol{\theta}} = \sum_{s \in \mathcal{S}} i(s)\frac{\partial \tilde{v}_\pi(s)}{\partial \boldsymbol{\theta}}.$$

Recall that $\tilde{v}_\pi(s) = \sum_a \pi(a|s; \boldsymbol{\theta})\tilde{q}_\pi(s, a) + \tau\mathcal{H}(\pi(\cdot|s; \boldsymbol{\theta}))$. We define the following notation which helps with the recursive expression of the gradient:

$$\tilde{\mathbf{g}}(s) = \sum_a \frac{\partial \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\tilde{q}_\pi(s, a) + \tau\frac{\partial}{\partial \boldsymbol{\theta}}\mathcal{H}(\pi(\cdot|s; \boldsymbol{\theta})),$$

where $\tilde{\mathbf{g}} : \mathcal{S} \to \mathbb{R}^d$. The gradient of the entropy regularized value function is then

$$\begin{aligned}
\frac{\partial \tilde{v}_\pi(s)}{\partial \boldsymbol{\theta}} &= \frac{\partial}{\partial \boldsymbol{\theta}}\left[\sum_a \pi(a|s; \boldsymbol{\theta})\tilde{q}_\pi(s, a) + \tau\mathcal{H}(\pi(\cdot|s; \boldsymbol{\theta}))\right] \\
&= \sum_a \frac{\partial \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\tilde{q}_\pi(s, a) + \sum_a \pi(a|s; \boldsymbol{\theta})\frac{\partial \tilde{q}_\pi(s, a)}{\partial \boldsymbol{\theta}} + \tau\frac{\partial}{\partial \boldsymbol{\theta}}\mathcal{H}(\pi(\cdot|s; \boldsymbol{\theta})) \qquad (37) \\
&= \tilde{\mathbf{g}}(s) + \sum_a \pi(a|s; \boldsymbol{\theta})\frac{\partial \sum_{s'} p(s'|s, a)(r(s, a, s') + \gamma(s, a, s')\tilde{v}_\pi(s'))}{\partial \boldsymbol{\theta}} \\
&= \tilde{\mathbf{g}}(s) + \sum_a \pi(a|s; \boldsymbol{\theta})\sum_{s'} p(s'|s, a)\gamma(s, a, s')\frac{\partial \tilde{v}_\pi(s')}{\partial \boldsymbol{\theta}}.
\end{aligned}$$

So the gradient of the regularized value function can be written in vector form similar to the unregularized version. Let $\dot{\tilde{\mathbf{v}}}_\pi \in \mathbb{R}^{|\mathcal{S}| \times d}$ be the matrix of gradients of $\tilde{v}_\pi$ for each state s; and $\tilde{\mathbf{G}} \in \mathbb{R}^{|\mathcal{S}| \times d}$ the matrix where each row corresponds to state s in the vector $\tilde{\mathbf{g}}(s)$. Then

$$\dot{\tilde{\mathbf{v}}}_\pi = \tilde{\mathbf{G}} + \mathbf{P}_{\pi,\gamma}\dot{\tilde{\mathbf{v}}}_\pi \tag{38}$$
$$\implies \dot{\tilde{\mathbf{v}}}_\pi = (\mathbf{I} - \mathbf{P}_{\pi,\gamma})^{-1}\tilde{\mathbf{G}}.$$

Therefore, we obtain

$$\sum_{s \in \mathcal{S}} i(s)\frac{\partial \tilde{v}_\pi(s)}{\partial \boldsymbol{\theta}} = \mathbf{i}^\top \dot{\tilde{\mathbf{v}}}_\pi = \mathbf{i}^\top(\mathbf{I} - \mathbf{P}_{\pi,\gamma})^{-1}\tilde{\mathbf{G}} = \mathbf{m}^\top\tilde{\mathbf{G}}$$

$$= \sum_{s \in \mathcal{S}} m(s)\left[\sum_a \frac{\partial \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\tilde{q}_\pi(s,a) + \tau\frac{\partial}{\partial \boldsymbol{\theta}}\mathcal{H}(\pi(\cdot|s;\boldsymbol{\theta}))\right].$$

We can further simplify this gradient by explicitly computing the gradient of the negative of the entropy:

$$\frac{\partial\text{-}\mathcal{H}(\pi(\cdot|s;\boldsymbol{\theta}))}{\partial \boldsymbol{\theta}} = \sum_a \frac{\partial \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\log\pi(a|s;\boldsymbol{\theta}) + \sum_a \pi(a|s;\boldsymbol{\theta})\frac{\partial\log\pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$$
$$= \sum_a \frac{\partial \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\log\pi(a|s;\boldsymbol{\theta}) + \sum_a \frac{\partial \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$$
$$= \sum_a \frac{\partial \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\log\pi(a|s;\boldsymbol{\theta}),$$

where the last line follows from the fact that $\sum_a \frac{\partial \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \frac{\partial}{\partial \boldsymbol{\theta}}\sum_a \pi(a|s;\boldsymbol{\theta}) = \frac{\partial}{\partial \boldsymbol{\theta}}1 = \mathbf{0}.$ $\blacksquare$

This form looks a bit different than the (on-policy) policy gradient update given by (Geist et al., 2019, Theorem 6). We can actually get a similar form by rewriting the inner term above. For a specific s, using chain rule on the entropy we obtain

$$\sum_a \frac{\partial \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\tilde{q}_\pi(s,a) + \tau\frac{\partial}{\partial \boldsymbol{\theta}}\mathcal{H}(\pi(\cdot|s;\boldsymbol{\theta})) = \sum_a \frac{\partial \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\tilde{q}_\pi(s,a) + \tau\frac{\partial \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\frac{\partial\mathcal{H}(\pi(\cdot|s;\boldsymbol{\theta}))}{\partial \pi(a|s;\boldsymbol{\theta})}$$
$$= \sum_a \frac{\partial \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\left[\tilde{q}_\pi(s,a) + \tau\frac{\partial\mathcal{H}(\pi(\cdot|s;\boldsymbol{\theta}))}{\partial \pi(a|s;\boldsymbol{\theta})}\right].$$

D.2 Regularized OPPG theorem with continuous states and actions

Theorem 11

$$\frac{\partial \tilde{J}_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \int_\mathcal{S} m(s)\left[\int_\mathcal{A} \frac{\partial \pi(a|s;\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\tilde{q}_\pi(s,a)\,\mathrm{d}a + \tau\frac{\partial}{\partial \boldsymbol{\theta}}\Omega(\pi(\cdot|s;\boldsymbol{\theta}))\right]\mathrm{d}s$$

Proof We extend the result from the previous section to the continuous state and action setting using a similar strategy to the one we used in Appendix B to extend Theorem 2

to the continuous state setting. In addition, we opt for a generic regularizer $\Omega(\pi(\cdot|s;\boldsymbol{\theta}))$: $\Delta(\mathcal{A}) \to \mathbb{R}$, which slightly modifies the definition of the regularized value functions to be

$$\tilde{v}_\pi(s) = \int_\mathcal{A} \pi(a|s;\boldsymbol{\theta})\tilde{q}_\pi(s,a)\,\mathrm{d}a + \tau\Omega(\pi(\cdot|s;\boldsymbol{\theta})) \qquad \forall s \in \mathcal{S}$$

$$\tilde{q}_\pi(s,a) = \int_\mathcal{S} P(s'|s,a)\big[r(s,a,s') + \gamma(s,a,s')\tilde{v}_\pi(s')\big]\,\mathrm{d}s' \qquad \forall s \in \mathcal{S}, \forall a \in \mathcal{A}.$$

We use the following notation to simplify the derivation:

$$\tilde{\mathbf{g}}(s) = \int_\mathcal{A} \frac{\partial\pi(a|s;\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}\tilde{q}_\pi(s,a)\,\mathrm{d}a + \tau\frac{\partial}{\partial\boldsymbol{\theta}}\Omega(\pi(\cdot|s;\boldsymbol{\theta})).$$

Starting with the continuous objective function from equation (6) and using the Leibniz integral rule to differentiate under the integral (which applies because we assumed the state space is a compact set in Assumption 2 from Appendix A.1):

$$\frac{\partial J_\mu(\boldsymbol{\theta})}{\partial\boldsymbol{\theta}} = \frac{\partial}{\partial\boldsymbol{\theta}}\int_\mathcal{S} i(s)\tilde{v}_\pi(s)\,\mathrm{d}s = \int_\mathcal{S} i(s)\frac{\partial\tilde{v}_\pi(s)}{\partial\boldsymbol{\theta}}\,\mathrm{d}s. \tag{39}$$

Next, we find a recursive expression for the gradient of the regularized value function:

$$\begin{aligned}
\frac{\partial\tilde{v}_\pi(s)}{\partial\boldsymbol{\theta}} &= \frac{\partial}{\partial\boldsymbol{\theta}}\int_\mathcal{A} \pi(a|s;\boldsymbol{\theta})\tilde{q}_\pi(s,a)\,\mathrm{d}a + \tau\Omega(\pi(\cdot|s;\boldsymbol{\theta})) \\
&= \int_\mathcal{A} \frac{\partial\pi(a|s;\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}\tilde{q}_\pi(s,a)\,\mathrm{d}a + \int_\mathcal{A} \pi(a|s;\boldsymbol{\theta})\frac{\partial\tilde{q}_\pi(s,a)}{\partial\boldsymbol{\theta}}\,\mathrm{d}a + \tau\frac{\partial}{\partial\boldsymbol{\theta}}\Omega(\pi(\cdot|s;\boldsymbol{\theta})) \\
&= \tilde{\mathbf{g}}(s) + \int_\mathcal{A} \pi(a|s;\boldsymbol{\theta})\frac{\partial\tilde{q}_\pi(s,a)}{\partial\boldsymbol{\theta}}\,\mathrm{d}a \\
&= \tilde{\mathbf{g}}(s) + \int_\mathcal{A} \pi(a|s;\boldsymbol{\theta})\frac{\partial}{\partial\boldsymbol{\theta}}\int_\mathcal{S} P(s'|s,a)\big[r(s,a,s') + \gamma(s,a,s')\tilde{v}_\pi(s')\big]\,\mathrm{d}s'\,\mathrm{d}a \\
&= \tilde{\mathbf{g}}(s) + \int_\mathcal{A} \pi(a|s;\boldsymbol{\theta})\int_\mathcal{S} P(s'|s,a)\gamma(s,a,s')\frac{\partial\tilde{v}_\pi(s')}{\partial\boldsymbol{\theta}}\,\mathrm{d}s'\,\mathrm{d}a \\
&= \tilde{\mathbf{g}}(s) + \int_\mathcal{S}\int_\mathcal{A} \pi(a|s;\boldsymbol{\theta})P(s'|s,a)\gamma(s,a,s')\,\mathrm{d}a\frac{\partial\tilde{v}_\pi(s')}{\partial\boldsymbol{\theta}}\,\mathrm{d}s',
\end{aligned}$$

where in the last step we used Fubini's Theorem to switch the order of integrals (which we can do because the absolute value of the integral is finite due to Assumptions 1 and 2). Again we will use the shorthand $\mathbf{P}_{\pi,\gamma}(s,s') = \int_\mathcal{A} \pi(a|s;\boldsymbol{\theta})P(s'|s,a)\gamma(s,a,s')\,\mathrm{d}a$ to simplify the derivation. Next we write $\frac{\partial\tilde{v}_\pi(s)}{\partial\boldsymbol{\theta}}$ as the integral transform

$$\frac{\partial\tilde{v}_\pi(s)}{\partial\boldsymbol{\theta}} = \int_\mathcal{S} \delta(s,s')\frac{\partial\tilde{v}_\pi(s')}{\partial\boldsymbol{\theta}}\,\mathrm{d}s',$$

plug it into the previous equation, and solve to get

$$\int_\mathcal{S} \delta(s,s')\frac{\partial\tilde{v}_\pi(s')}{\partial\boldsymbol{\theta}}\,\mathrm{d}s' = \tilde{\mathbf{g}}(s) + \int_\mathcal{S} \mathbf{P}_{\pi,\gamma}(s,s')\frac{\partial\tilde{v}_\pi(s')}{\partial\boldsymbol{\theta}}\,\mathrm{d}s'$$

$$\int_{\mathbb{S}} \left(\delta(s, s') - \mathbf{P}_{\pi,\gamma}(s, s') \right) \frac{\partial \tilde{v}_\pi(s')}{\partial \boldsymbol{\theta}} \, \mathrm{d}s' = \tilde{\mathbf{g}}(s)$$

$$\implies \frac{\partial \tilde{v}_\pi(s)}{\partial \boldsymbol{\theta}} = \int_{\mathbb{S}} k(s, s') \tilde{\mathbf{g}}(s') \, \mathrm{d}s',$$

where $k(s, s')$ is the inverse kernel of $\delta(s, s') - \mathbf{P}_{\pi,\gamma}(s, s')$. Then plugging the above into Equation (39), interchanging integrals using Fubini's theorem, and simplifying gives

$$\begin{aligned}
\frac{\partial J_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} &= \int_{\mathbb{S}} i(s) \int_{\mathbb{S}} k(s, s') \tilde{\mathbf{g}}(s') \, \mathrm{d}s' \, \mathrm{d}s \\
&= \int_{\mathbb{S}} \int_{\mathbb{S}} i(s) k(s, s') \, \mathrm{d}s \tilde{\mathbf{g}}(s') \, \mathrm{d}s' \\
&= \int_{\mathbb{S}} m(s') \tilde{\mathbf{g}}(s') \, \mathrm{d}s' \\
&= \int_{\mathbb{S}} m(s) \tilde{\mathbf{g}}(s) \, \mathrm{d}s \\
&= \int_{\mathbb{S}} m(s) \left[\int_{\mathcal{A}} \frac{\partial \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \tilde{q}_\pi(s, a) \, \mathrm{d}a + \tau \frac{\partial}{\partial \boldsymbol{\theta}} \Omega(\pi(\cdot|s; \boldsymbol{\theta})) \right] \mathrm{d}s.
\end{aligned}$$

■

Appendix E. Proof of Proposition 10

This section proves Proposition 10. First, a result by Mei et al. (2020) for tabular domains is extended to state aggregation and transition dependent discounting (which includes the three-state counterexample) to show that entropy-regularized policy gradient converges to a point where the gradient is zero. Then, it is shown that such point is not a stationary point for semi-gradient updates.

E.1 Entropy-regularized PG under State Aggregation

We assume the policy is a softmax policy and additionally specifically characterize the gradient under state aggregation. This specific characterization facilitates showing that the solution to the objective lies on the interior of the simplex, and so that a stationary point exists.

We define $\mathrm{alias}(s)$ as the set of state that share their representation with s including s itself. We additionally defined $\mathbb{S}_{\mathrm{rep}} \subset \mathbb{S}$ the set of representative states, one for each bin in the aggregation. For example, in the three-state counterexample $\mathrm{alias}(s_0) = \{s_0\}$ and $\mathrm{alias}(s_1) = \mathrm{alias}(s_2) = \{s_1, s_2\}$. We simply need to choose one state in each aliased set, giving $\mathbb{S}_{\mathrm{rep}} = \{s_0, s_1\}$.

For a parameter set $\boldsymbol{\theta}$, the policy is a softmax transform defined as follows. For a state s' with representative state s, i.e., $s' \in \mathrm{alias}(s)$, we have

$$\pi(s', a; \boldsymbol{\theta}) = \frac{\exp\big(\boldsymbol{\theta}(s, a)\big)}{\sum_{a'} \exp\big(\boldsymbol{\theta}(s, a')\big)}. \tag{40}$$

Using again the three-state counterexample, $\boldsymbol{\theta}$ has four components: $\boldsymbol{\theta}(s_0, a_0)$ and $\boldsymbol{\theta}(s_0, a_1)$ specify the policy for s_0, and $\boldsymbol{\theta}(s_1, a_0)$ and $\boldsymbol{\theta}(s_1, a_1)$ specify the policy for both s_1 and s_2 since these two states are aliased.

The softmax policy has a simple well-known gradient, which we can explicitly write for state-aggregation in the following lemma for easy reference in later proofs.

Lemma 12 *Assume the policy uses a softmax distribution, as in Equation 40. For any state $s \in \mathcal{S}_{rep}$, for any $s' \in \mathrm{alias}(s)$,*

$$\frac{\partial \pi(s', a'; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s, a)} = \begin{cases} \pi(s', a; \boldsymbol{\theta})[1 - \pi(s', a; \boldsymbol{\theta})] & \text{if } a' = a \\ -\pi(s', a; \boldsymbol{\theta})\pi(s', a'; \boldsymbol{\theta}) & \text{if } a' \neq a \end{cases} \tag{41}$$

and, across all actions, this can be compactly written as

$$\frac{\partial \pi(s', a'; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s, \cdot)} = \mathrm{diag}(\pi(s, .; \boldsymbol{\theta})) - \pi(s, .; \boldsymbol{\theta})\pi(s, .; \boldsymbol{\theta})^\top.$$

Proof For explicit step to obtain this derivation, notice first that

$$\log \pi(s', a; \boldsymbol{\theta}) = \boldsymbol{\theta}(s, a) - \log \sum_{a'} \exp(\boldsymbol{\theta}(s, a'))$$

and so

$$\begin{aligned}
\frac{\partial \pi(s', a'; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s, a)} &= \pi(s', a'; \boldsymbol{\theta}) \frac{\partial \log \pi(s', a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s, a)} \\
&= \pi(s', a'; \boldsymbol{\theta}) \frac{\partial[\boldsymbol{\theta}(s, a') - \log \sum_{a''} \exp(\boldsymbol{\theta}(s, a''))]}{\partial \boldsymbol{\theta}(s, a)} \\
&= \pi(s', a'; \boldsymbol{\theta}) \left[\frac{\partial \boldsymbol{\theta}(s, a')}{\partial \boldsymbol{\theta}(s, a)} - \frac{\partial \log \sum_{a''} \exp(\boldsymbol{\theta}(s, a''))}{\partial \boldsymbol{\theta}(s, a)} \right].
\end{aligned}$$

If $a' = a$, then

$$\begin{aligned}
\frac{\partial \boldsymbol{\theta}(s, a')}{\partial \boldsymbol{\theta}(s, a)} - \frac{\partial \log \sum_{a''} \exp(\boldsymbol{\theta}(s, a''))}{\partial \boldsymbol{\theta}(s, a)} &= 1 - \frac{1}{\sum_{a''} \exp(\boldsymbol{\theta}(s, a''))} \frac{\partial \sum_{a''} \exp(\boldsymbol{\theta}(s, a''))}{\partial \boldsymbol{\theta}(s, a)} \\
&= 1 - \frac{1}{\sum_{a''} \exp(\boldsymbol{\theta}(s, a''))} \exp(\boldsymbol{\theta}(s, a)) \\
&= 1 - \pi(a|s; \boldsymbol{\theta}) \\
&= 1 - \pi(s', a; \boldsymbol{\theta}),
\end{aligned}$$

where the last step follows from the fact that $\pi(s', a; \boldsymbol{\theta}) = \pi(a|s; \boldsymbol{\theta})$ under state aggregation. If $a' \neq a$, then

$$\frac{\partial \boldsymbol{\theta}(s, a')}{\partial \boldsymbol{\theta}(s, a)} - \frac{\partial \log \sum_{a''} \exp(\boldsymbol{\theta}(s, a''))}{\partial \boldsymbol{\theta}(s, a')} = 0 - \pi(a|s; \boldsymbol{\theta}) = -\pi(s', a; \boldsymbol{\theta}).$$

$\blacksquare$

The gradient ascent update using the entropy-regularized objective is

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \eta \frac{\partial \tilde{J}_\mu(\boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}} \tag{42}$$

for stepsize $\eta > 0$. For an appropriately small stepsize η, this will converge to a stationary point if one exists, and otherwise move to a point on the simplex, where $\boldsymbol{\theta}_t(s,a) \to \infty$ for an action a in state s. The standard condition on η is that $\eta < 1/L$ for Lipschitz constant L of $\tilde{J}_\mu$, which is a common assumption for policy gradient methods. In the next section, we show that this update converges to a stationary on the interior of the policy simplex.

We provide one more result, that is useful for this characterization. Lemma 13 breaks down the gradient into its components. Mei et al. (2020), in their Lemma 10, proved this result for entropy regularized policy gradients, assuming tabular policies in the on-policy setting. We extend it to state aggregation, with our off-policy gradient. Their presentation has an extra term $-\tau \log \pi(a|s; \boldsymbol{\theta})$. This discrepancy is just a difference in notation. In our formulation, entropy regularized action values are defined to contain the entropy term so that the connection between the regularized and unregularized policy gradient is clearer.

Lemma 13 *For each state $s \in \mathcal{S}_{rep}$ and action a*

$$\frac{\partial \tilde{J}_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s,a)} = \sum_{s' \in \text{alias}(s)} m(s')\pi(s',a; \boldsymbol{\theta}) \left[\tilde{q}_\pi(s',a) - \tilde{v}_\pi(s') \right].$$

Proof According to Theorem 8

$$\frac{\partial \tilde{J}_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \sum_{s \in \mathcal{S}} m(s) \sum_a \frac{\partial \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \tilde{q}_\pi(s,a).$$

Notice that only for states $s' \in \text{alias}(s)$, and any a', we have $\frac{\partial \pi(s',a'; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s,a)} \neq 0$. We can write partial derivatives w.r.t. the parameters of each state and action, using Equation 41:

$$\frac{\partial \tilde{J}_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s,a)} = \sum_{s' \in \text{alias}(s)} m(s') \sum_{a'} \frac{\partial \pi(s',a'; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s,a)} \tilde{q}_\pi(s',a')$$

$$= \sum_{s' \in \text{alias}(s)} m(s') \left[\pi(s',a; \boldsymbol{\theta})(1 - \pi(s',a; \boldsymbol{\theta})) \tilde{q}_\pi(s',a) \right.$$

$$\left. - \sum_{a' \neq a} \pi(s',a; \boldsymbol{\theta})\pi(s',a'; \boldsymbol{\theta}) \tilde{q}_\pi(s',a') \right]$$

$$= \sum_{s' \in \text{alias}(s)} m(s')\pi(s',a; \boldsymbol{\theta}) \left[\tilde{q}_\pi(s',a) - \sum_{a'} \pi(s',a'; \boldsymbol{\theta})\tilde{q}_\pi(s',a') \right]$$

$$= \sum_{s' \in \text{alias}(s)} m(s')\pi(s',a; \boldsymbol{\theta}) \left[\tilde{q}_\pi(s',a) - \tilde{v}_\pi(s') \right].$$

■

49

E.2 Existence of stationary points for the entropy-regularized PG objective

We are now ready to prove that a stationary point exists for $\tau > 0$. Mei et al. (2020) (Lemma 17) proved that, under tabular representation and softmax parameterization, entropy regularized policy gradient updates converge to a point where the gradient is zero. We extend their proof to state aggregation and transition-dependent discounting. We additionally avoid assuming that the rewards are non-negative.

Assumption 4 (Lipschitz continuity) $\tilde{J}_\mu$ is Lipschitz continuous with Lipschitz constant L.

Assumption 5 (Bounded reward) $|r(s, a)| \leq R_{max}$.

Assumption 6 (Bounded expected sum of future discounting) There exists $C \in \mathbb{R}$ such that for any $s \in \mathcal{S}$, $\mathbb{E}_\pi[\gamma_{t+1} + \gamma_{t+1}\gamma_{t+2} + \ldots | S_t = s] \leq C$. This assumption is satisfied if π is proper, or if $\gamma_{t+i} < 1$ after a bounded number of steps $i \in \mathbb{N}$.

We first prove that the state values are bounded and show that the action-values are lower bounded by the scaled log of the probabilities. We use this to show the main result in Proposition 15.

Lemma 14 Under Assumptions 5 and 6, for any given policy parameters $\boldsymbol{\theta}$,

$$\tilde{v}_\pi(s) \leq CR_{max} + C\tau \log n_a$$
$$\tilde{q}_\pi(s, a) - \tilde{v}_\pi(s) \geq -\tau \log \pi(a|s; \boldsymbol{\theta}) - (2C + 1)R_{max} - C\tau \log n_a.$$

Proof First notice that for any distribution p over actions

$$\text{Entropy}(p) \overset{\text{def}}{=} -\sum_{a \in \mathcal{A}} p(a) \log p(a) \leq -\sum_{a \in \mathcal{A}} \frac{1}{n_a} \log\left(\frac{1}{n_a}\right) = \log n_a,$$

where n_a is the number of actions. The inequality follows from the fact that the uniform distribution has the highest entropy. We use this inequality to bound $\mathbb{E}_\pi[-\log \pi(A_t|S_t; \boldsymbol{\theta})|S_t = s] = -\sum_a \pi(a|s; \boldsymbol{\theta}) \log \pi(a|s; \boldsymbol{\theta})$, which is the entropy of $\pi(s, \cdot; \boldsymbol{\theta})$. Using this bound on entropy, and the facts that the entropy is nonnegative and $\gamma_{t+i} \leq 1$ for all $i \in \mathbb{N}$, we obtain

$$\tilde{v}_\pi(s) = \mathbb{E}_\pi[G_t|S_t = s] - \tau\mathbb{E}_\pi[\log \pi(A_t|S_t; \boldsymbol{\theta}) + \gamma_{t+1} \log \pi(S_{t+1}, A_{t+1}; \boldsymbol{\theta}) + \ldots | S_t = s]$$
$$\leq CR_{max} + \tau\mathbb{E}_\pi[-\log \pi(A_t|S_t; \boldsymbol{\theta})|S_t = s] + \tau\mathbb{E}_\pi[\gamma_{t+1}(-\log \pi(S_{t+1}, A_{t+1}; \boldsymbol{\theta}))|S_t = s] + \ldots$$
$$\leq CR_{max} + \tau \log n_a + \tau \log n_a \mathbb{E}_\pi[\gamma_{t+1}|S_t = s] + \tau \log n_a \mathbb{E}_\pi[\gamma_{t+2}|S_t = s] + \ldots$$
$$\leq CR_{max} + C\tau \log n_a.$$

Finally, we can lower bound the advantage function, using $\tilde{v}_\pi(s) \geq \mathbb{E}_\pi[G_t|S_t = s] \geq -CR_{max}$ and

$$\tilde{q}_\pi(s, a) = \mathbb{E}_\pi[R_{t+1} + \gamma_{t+1}\tilde{v}_\pi(S_{t+1})|S_t = s, A_t = a] - \tau \log \pi(a|s; \boldsymbol{\theta})$$
$$\geq -R_{max} - CR_{max} - \tau \log \pi(a|s; \boldsymbol{\theta}),$$

giving

$$\tilde{q}_\pi(s, a) - \tilde{v}_\pi(s) \geq -R_{max} - CR_{max} - \tau \log \pi(a|s; \boldsymbol{\theta}) - \tilde{v}_\pi(s)$$
$$\geq -R_{max} - CR_{max} - \tau \log \pi(a|s; \boldsymbol{\theta}) - (CR_{max} + C\tau \log n_a)$$
$$= -\tau \log \pi(a|s; \boldsymbol{\theta}) - (2C + 1)R_{max} - C\tau \log n_a.$$

∎

Proposition 15 *Under Assumptions 4, 5 and 6, the entropy-regularized policy update in Equation 42 with $\tau > 0$ and $\eta < 1/L$ converges to a point with zero gradient.*

Proof Let $\{\boldsymbol{\theta}_t\}_{t=1}^\infty$ be the trajectory of parameters under the gradient ascent update, where $\boldsymbol{\theta}_t \to \boldsymbol{\theta}_\infty$. This trajectory either converges to finite $\boldsymbol{\theta}_\infty$ that provides a policy on the interior of the policy simplex, or it pushes the weights for a subset of actions to infinity to converge to a point on the simplex, where certain actions have zero probability. The gradient is only zero for the solution on the interior, and so to prove the result we simply need to show that this process converges to the interior. We show this is true for every parameter $\boldsymbol{\theta}_t(s, a)$ in the vector $\boldsymbol{\theta}_t$, where $s \in \mathcal{S}_{\text{rep}}$ and $a \in \mathcal{A}$.

Define $\mathcal{A}_0(s)$ and $\mathcal{A}_+(s)$ as the sets of actions with zero and nonzero probability under $\pi(s, .; \boldsymbol{\theta}_\infty)$. We use a proof by contradiction to show that $\mathcal{A}_0(s) = \emptyset$ for any s as long as $\tau > 0$. Suppose there exist s and a_0 such that $a_0 \in \mathcal{A}_0(s)$. Note that $\pi(s', a; \boldsymbol{\theta}) = \pi(s, a; \boldsymbol{\theta})$ for all $s' \in \text{alias}(s)$. We know that $\pi(s, a_0; \boldsymbol{\theta}_t) \to 0$ and $-\log \pi(s, a_0; \boldsymbol{\theta}_t) \to \infty$ as $t \to \infty$. Therefore there exists $t_0 > 0$ such that for all $s' \in \text{alias}(s)$ and $t \geq t_0$

$$-\log \pi(s', a_0; \boldsymbol{\theta}_t) \geq \frac{(2C + 1)R_{max} + C\tau \log n_a}{\tau}.$$

According to Lemma 13, for all $t \geq t_0$

$$\frac{\partial \tilde{J}_\mu(\boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}_t(s, a_0)} = \sum_{s' \in \text{alias}(s)} m(s')\pi(s', a_0; \boldsymbol{\theta}_t)\left[\tilde{q}_\pi(s', a_0) - \tilde{v}_\pi(s')\right] \qquad \triangleright \text{And applying Lemma 14}$$

$$= \sum_{s' \in \text{alias}(s)} m(s')\pi(s', a_0; \boldsymbol{\theta}_t)\left[-\tau \log \pi(a|s; \boldsymbol{\theta}) - (2C + 1)R_{max} - C\tau \log n_a\right]$$

$$\geq \sum_{s' \in \text{alias}(s)} m(s')\pi(s', a_0; \boldsymbol{\theta}_t)\left[\tau \frac{(2C + 1)R_{max} + C\tau \log n_a}{\tau} - ((2C + 1)R_{max} + C\tau \log n_a)\right]$$

$$= 0. \tag{43}$$

So $\boldsymbol{\theta}_t(s, a_0)$ is non-decreasing after t_0, because we only add this non-negative gradient to $\boldsymbol{\theta}_t(s, a_0)$. This means that $\boldsymbol{\theta}_\infty(s, a_0)$ is lower bounded by a constant c:

$$\exp(\boldsymbol{\theta}_\infty(s, a_0)) > e^c > 0.$$

Next, we bound $|\sum_a \exp(\boldsymbol{\theta}_\infty(s, a))|$. Notice that the sum of gradient components over all actions at a state is zero:

$$\sum_a \frac{\partial \tilde{J}_\mu(\boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}_t(s, a)} = \sum_{s' \in \text{alias}(s)} m(s') \sum_a \pi(s', a; \boldsymbol{\theta}_t)[\tilde{q}_\pi(s', a) - \tilde{v}_\pi(s')]$$

$$= \sum_{s' \in \mathrm{alias}(s)} m(s')\big(\tilde{v}_\pi(s') - \tilde{v}_\pi(s')\big) = 0. \tag{44}$$

Because

$$\sum_a \frac{\partial \tilde{J}_\mu(\boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}_t(s,a)} = \sum_{a_0 \in \mathcal{A}_0} \frac{\partial \tilde{J}_\mu(\boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}_t(s,a_0)} + \sum_{a_+ \in \mathcal{A}_+} \frac{\partial \tilde{J}_\mu(\boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}_t(s,a_+)}$$

we get that for all $t > t_0$,

$$\sum_{a_+ \in \mathcal{A}_+} \frac{\partial \tilde{J}_\mu(\boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}_t(s,a_+)} = \sum_a \frac{\partial \tilde{J}_\mu(\boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}_t(s,a)} - \sum_{a_0 \in \mathcal{A}_0} \frac{\partial \tilde{J}_\mu(\boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}_t(s,a_0)}$$

$$= 0 - \sum_{a_0 \in \mathcal{A}_0} \frac{\partial \tilde{J}_\mu(\boldsymbol{\theta}_t)}{\partial \boldsymbol{\theta}_t(s,a_0)} \le 0,$$

where the last line follows from Equation 43. This means that $\sum_{a_+ \in \mathcal{A}_+(s)} \boldsymbol{\theta}_t(s,a_+)$ is non-increasing after t_0. For $a_+ \in \mathcal{A}_+(s)$, we know that there exists some constant that lower bounds $\boldsymbol{\theta}_t(s,a_+)$ for all t. Otherwise, $a_+ \in \mathcal{A}_+(s)$ would not have non-zero probability under $\pi(s,.;\boldsymbol{\theta}_\infty)$. Therefore, non-increasing $\sum_{a_+ \in \mathcal{A}_+(s)} \boldsymbol{\theta}_t(s,a_+)$ cannot be due to some $\boldsymbol{\theta}_t(s,a_+)$ approaching $-\infty$ while others approach ∞. Instead, each $\boldsymbol{\theta}_t(s,a_+)$ must also be bounded above. This means there exists some $b_+ > 0$ such that $\exp(-b_+) < \exp(\boldsymbol{\theta}_t(s,a_+)) \exp(b_+)$ for every $a_+ \in \mathcal{A}_+(s)$.

For all $a_0 \in \mathcal{A}_0$, we also know that there exists b_0 such that $\exp(\boldsymbol{\theta}_t(s,a_0)) < b_0$ for all $t > t_0$. Otherwise, if $\exp(\boldsymbol{\theta}_t(s,a_0))$ approaches infinity, then a_0 would not be in $\mathcal{A}_0$. Putting this together with the above, and knowing there is at least one $a_+ \in \mathcal{A}_+(s)$,

$$\sum_a \exp\big(\boldsymbol{\theta}_t(s,a)\big) \le n_a \exp(b_+) + b_0$$

$$\sum_a \exp\big(\boldsymbol{\theta}_t(s,a)\big) \ge \exp(b_+) > 0.$$

Finally, this gives

$$\pi(s,a_0;\boldsymbol{\theta}_t) = \frac{\exp\big(\boldsymbol{\theta}_t(s,a_0)\big)}{\sum_a \exp\big(\boldsymbol{\theta}_t(s,a)\big)}$$

$$\ge \frac{c}{n_a \exp(b_+) + b_0}.$$

Taking the limit as $t \to \infty$, this lower bound remains positive, implying that $\pi(s,a_0;\boldsymbol{\theta}_\infty) > 0$, which is a contradiction.

Therefore, no such a_0 can exist and so $\mathcal{A}_0(s) = \emptyset$ for all $s \in \mathcal{S}$. Since the policy converges to the interior of the probability simplex and the objective function is Lipschitz continuous, the gradient has to be zero at the point of convergence (Mei et al., 2020) (Lemma 17). ∎

E.3 Proof for Three-State Counterexample

In the previous section, we showed that a stationary point for the entropy-regularized PG objective exists. We now show that in the three-state counterexample that such a point is not a stationary point under semi-gradient updates.

Lemma 16 *A point with zero gradient is not a stationary point under semi-gradient updates in the three-state counterexample.*

Proof Suppose $\boldsymbol{\theta}$ is a point with zero gradient, i.e.:

$$\frac{\partial \tilde{J}_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \mathbf{0}.$$

Every component of the gradient vector is zero, thus

$$\frac{\partial \tilde{J}_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1, .)} = \sum_{s \in \text{alias}(s_1)} m(s) \sum_a \frac{\partial \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s, .)} \tilde{q}_\pi(s, a) = \mathbf{0}.$$

Since taking any action from s_1 or s_2 results in termination, the following holds for $s \in \{s_1, s_2\}$ and any action a:

$$\tilde{q}_\pi(s, a) = \sum_{s'} \mathrm{P}(s'|s, a) \big[r(s, a, s') - \tau \log \pi(a|s; \boldsymbol{\theta}) + \gamma(s, a, s') \tilde{v}_\pi(s') \big]$$

$$= \sum_{s'} \mathrm{P}(s'|s, a) \big[r(s, a, s') - \tau \log \pi(a|s; \boldsymbol{\theta}) + \gamma(s, a, s') \tilde{v}_\pi(s') \big]$$

$$= r(s, a) - \tau \log \pi(a|s; \boldsymbol{\theta}) \qquad \triangleright \text{ Recall that } r(s, a) \stackrel{\text{def}}{=} \sum_{s'} \mathrm{P}(s'|s, a) r(s, a, s').$$

States s_1 and s_2 have the same representation, meaning that $\frac{\partial \pi(s_1, .; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1, .)} = \frac{\partial \pi(s_2, .; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_2, .)}$ and $\pi(s_1, .; \boldsymbol{\theta}) = \pi(s_2, .; \boldsymbol{\theta})$. So

$$\frac{\partial \tilde{J}_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1, .)} = \sum_{s \in \text{alias}(s_1)} m(s) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1, .)} [r(s, a) - \tau \log \pi(s_1, a; \boldsymbol{\theta})]$$

$$= \sum_{s \in \text{alias}(s_1)} m(s) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1, .)} r(s, a)$$

$$- \left(\sum_{s \in \text{alias}(s_1)} m(s) \right) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1, .)} \tau \log \pi(s_1, a; \boldsymbol{\theta}).$$

For simplicity of notation, define

$$M(s_1) \stackrel{\text{def}}{=} \sum_{s \in \text{alias}(s_1)} m(s)$$

$$D(s_1) \stackrel{\text{def}}{=} \sum_{s \in \text{alias}(s_1)} d_\mu(s),$$

letting us write the above as

$$\frac{\partial \tilde{J}_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} = \sum_{s \in \mathrm{alias}(s_1)} m(s) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} r(s, a) - M(s_1) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} \tau \log \pi(s_1, a; \boldsymbol{\theta}).$$

Similarly, the semi-gradient update to $\boldsymbol{\theta}(s_1,.)$ becomes

$$\sum_{s \in \mathrm{alias}(s_1)} d_\mu(s) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} r(s, a) - D(s_1) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} \tau \log \pi(s_1, a; \boldsymbol{\theta}).$$

We show that, even though $\boldsymbol{\theta}$ is a stationary point under the true gradient update, this semi-gradient under $\boldsymbol{\theta}$ is not zero. Notice first that

$$D(s_1) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} \tau \log \pi(s_1, a; \boldsymbol{\theta})$$

$$= D(s_1) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} \tau \log \pi(s_1, a; \boldsymbol{\theta}) + \frac{D(s_1)}{M(s_1)} \frac{\partial \tilde{J}_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} \qquad \triangleright \text{ because } \frac{\partial \tilde{J}_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} = 0$$

$$= D(s_1) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} \tau \log \pi(s_1, a; \boldsymbol{\theta})$$

$$+ \frac{D(s_1)}{M(s_1)} \left[\sum_{s \in \mathrm{alias}(s_1)} m(s) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} r(s, a) - M(s_1) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} \tau \log \pi(s_1, a; \boldsymbol{\theta}) \right]$$

$$= D(s_1) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} \tau \log \pi(s_1, a; \boldsymbol{\theta}) + \frac{D(s_1)}{M(s_1)} \left[\sum_{s \in \mathrm{alias}(s_1)} m(s) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} r(s, a) \right]$$

$$- D(s_1) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} \tau \log \pi(s_1, a; \boldsymbol{\theta})$$

$$= \frac{D(s_1)}{M(s_1)} \left[\sum_{s \in \mathrm{alias}(s_1)} m(s) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} r(s, a) \right].$$

Therefore,

$$\sum_{s \in \mathrm{alias}(s_1)} d_\mu(s) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} r(s, a) - D(s_1) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} \tau \log \pi(s_1, a; \boldsymbol{\theta}) \qquad (45)$$

$$= \sum_{s \in \mathrm{alias}(s_1)} d_\mu(s) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} r(s, a) - \frac{D(s_1)}{M(s_1)} \left[\sum_{s \in \mathrm{alias}(s_1)} m(s) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} r(s, a) \right]$$

$$= \sum_{s \in \mathrm{alias}(s_1)} \delta(s) \sum_a \frac{\partial \pi(s_1, a; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} r(s, a) \qquad \triangleright \text{ where } \delta(s) \overset{\text{def}}{=} d_\mu(s) - \frac{\sum_{s' \in \mathrm{alias}(s_1)} d_\mu(s')}{\sum_{s' \in \mathrm{alias}(s_1)} m(s')} m(s)$$

$$= \delta(s_1) \frac{\partial \pi(s_1, a_0; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} \cdot 2 + \delta(s_2) \frac{\partial \pi(s_1, a_1; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1,.)} \cdot 1$$

$$\hspace{4cm} \triangleright \text{ because } r(s_1, a_0) = 2, r(s_2, a_1) = 1, r(s_1, a_1) = r(s_2, a_0) = 0$$

$$= 2\big(\delta(s_1) - \delta(s_2)\big) \frac{\partial \pi(s_1, a_0; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1, .)}. \qquad \triangleright \text{ as } \frac{\partial \pi(s_1, a_1; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1, .)} = -\frac{\partial \pi(s_1, a_0; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1, .)} \text{ by Lemma 12}$$

The second factor $\frac{\partial \pi(s_1, a_0; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1, .)}$ is nonzero due to softmax parametrization. It therefore suffices to show that the first term $2\big(\delta(s_1) - \delta(s_2)\big)$ is not zero.

In each episode in the counterexample, regardless of the policy, the agent spends one step in s_0 and one step in either s_1 or s_2. Therefore

$$d_\mu(s_0) = 0.5, \ d_\mu(s_1) + d_\mu(s_2) = 0.5.$$

Unfolding the emphatic weightings gives

$$
\begin{aligned}
m(s_0) &= d_\mu(s_0) = 0.5 \\
m(s_1) &= d_\mu(s_1) + d_\mu(s_0)\pi(s_0, a_0; \boldsymbol{\theta}) = d_\mu(s_1) + 0.5\pi(s_0, a_0; \boldsymbol{\theta}) \\
m(s_2) &= d_\mu(s_2) + d_\mu(s_0)\pi(s_0, a_1; \boldsymbol{\theta}) = d_\mu(s_2) + 0.5\pi(s_0, a_1; \boldsymbol{\theta}) \\
&= 0.5 - d_\mu(s_1) + 0.5(1 - \pi(s_0, a_0; \boldsymbol{\theta})) = 1 - d_\mu(s_1) - 0.5\pi(s_0, a_0; \boldsymbol{\theta}).
\end{aligned}
$$

Plugging these values into $\delta(s)$ results in

$$
\begin{aligned}
\delta(s) &\overset{\text{def}}{=} d_\mu(s) - \frac{\sum_{s' \in \text{alias}(s_1)} d_\mu(s')}{\sum_{s' \in \text{alias}(s_1)} m(s')} m(s) \\
&= d_\mu(s) - \frac{d_\mu(s_1) + d_\mu(s_2)}{m(s_1) + m(s_2)} m(s) \\
&= d_\mu(s) - \frac{0.5}{1} m(s) = d_\mu(s) - 0.5m(s).
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
\delta(s_1) &= d_\mu(s_1) - 0.5m(s_1) = 0.5d_\mu(s_1) - 0.25\pi(s_0, a_0; \boldsymbol{\theta}) \\
\delta(s_2) &= d_\mu(s_2) - 0.5m(s_2) = 0.5 - d_\mu(s_1) - 0.5 + 0.5d_\mu(s_1) + 0.25\pi(s_0, a_0; \boldsymbol{\theta}) \\
&= -0.5d_\mu(s_1) + 0.25\pi(s_0, a_0; \boldsymbol{\theta}),
\end{aligned}
$$

and we get that

$$
\begin{aligned}
2\delta(s_1) - \delta(s_2) &= d_\mu(s_1) - 0.5\pi(s_0, a_0; \boldsymbol{\theta}) + 0.5d_\mu(s_1) - 0.25\pi(s_0, a_0; \boldsymbol{\theta}) \\
&= 1.5d_\mu(s_1) - 0.75\pi(s_0, a_0; \boldsymbol{\theta}) \\
&= 0.75\big(\mu(s_0, a_0) - \pi(s_0, a_0; \boldsymbol{\theta})\big). \qquad \triangleright \text{ because } d_\mu(s_1) = 0.5\mu(s_0, a_0)
\end{aligned}
$$

As long as $\pi(s_0, a_0; \boldsymbol{\theta}) \neq \mu(s_0, a_0)$, the semi-gradient is not zero. For example, as in Section 8.1, we can choose $\mu(s_0, a_0) = 0.25$. Now we show that $\pi(s_0, a_0; \boldsymbol{\theta}) \neq 0.25$ for a stationary point under the true gradient.

Let us first write the partial derivative w.r.t. the first parameter given that $\pi(s_0, a_0; \boldsymbol{\theta}) = 0.25$:

$$\frac{\partial \tilde{J}_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_0, a_0)} = \sum_{s \in \text{alias}(s_0)} m(s) \sum_a \frac{\partial \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_0, a_0)} \tilde{q}_\pi(s, a)$$

$$= m(s_0)\left[\frac{\partial \pi(s_0, a_0; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_0, a_0)}\tilde{q}_\pi(s_0, a_0) + \frac{\partial \pi(s_0, a_1; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_0, a_0)}\tilde{q}_\pi(s_0, a_1)\right]$$

$$= 0.5\left[\pi(s_0, a_0; \boldsymbol{\theta})(1 - \pi(s_0, a_0; \boldsymbol{\theta}))\tilde{q}_\pi(s_0, a_0)\right.$$

$$\left. - \pi(s_0, a_1; \boldsymbol{\theta})\pi(s_0, a_0; \boldsymbol{\theta})\tilde{q}_\pi(s_0, a_1)\right]$$

$$= 0.5 * 0.25 * 0.75\left[\tilde{q}_\pi(s_0, a_0) - \tilde{q}_\pi(s_0, a_1)\right].$$

For this derivative to be zero, $\tilde{q}_\pi(s_0, a_0) - \tilde{q}_\pi(s_0, a_1)$ must be zero.

$$\tilde{q}_\pi(s_0, a_0) - \tilde{q}_\pi(s_0, a_1)$$
$$= -\tau \log \pi(s_0, a_0; \boldsymbol{\theta}) + \pi(s_1, a_0, \boldsymbol{\theta})\tilde{q}_\pi(s_1, a_0) + \pi(s_1, a_1, \boldsymbol{\theta})\tilde{q}_\pi(s_1, a_1)$$
$$+ \tau \log \pi(s_0, a_1; \boldsymbol{\theta}) - \pi(s_2, a_0, \boldsymbol{\theta})\tilde{q}_\pi(s_2, a_0) - \pi(s_2, a_1, \boldsymbol{\theta})\tilde{q}_\pi(s_2, a_1)$$
$$= -\tau \log 0.25 + \pi(s_1, a_0, \boldsymbol{\theta})(2 - \tau \log \pi(s_1, a_0, \boldsymbol{\theta})) + \pi(s_1, a_1, \boldsymbol{\theta})(-\tau \log \pi(s_1, a_1, \boldsymbol{\theta}))$$
$$+ \tau \log 0.75 - \pi(s_2, a_0, \boldsymbol{\theta})(-\tau \log \pi(s_2, a_0, \boldsymbol{\theta})) - \pi(s_2, a_1, \boldsymbol{\theta})(1 - \tau \log \pi(s_2, a_1, \boldsymbol{\theta}))$$

Recall that $\pi(s_2, \cdot, \boldsymbol{\theta}) = \pi(s_1, \cdot, \boldsymbol{\theta})$ and $\pi(s_1, a_1, \boldsymbol{\theta}) = 1 - \pi(s_1, a_0, \boldsymbol{\theta})$. Then

$$\tilde{q}_\pi(s_0, a_0) - \tilde{q}_\pi(s_0, a_1)$$
$$= -\tau \log 0.25 + \pi(s_1, a_0, \boldsymbol{\theta})(2 - \tau \log \pi(s_1, a_0, \boldsymbol{\theta})) + \pi(s_1, a_1, \boldsymbol{\theta})(-\tau \log \pi(s_1, a_1, \boldsymbol{\theta}))$$
$$+ \tau \log 0.75 - \pi(s_1, a_0, \boldsymbol{\theta})(-\tau \log \pi(s_1, a_0, \boldsymbol{\theta})) - \pi(s_1, a_1, \boldsymbol{\theta})(1 - \tau \log \pi(s_1, a_1, \boldsymbol{\theta}))$$
$$= \tau(\log 0.75 - \log 0.25) + 2\pi(s_1, a_0, \boldsymbol{\theta}) - \pi(s_1, a_1, \boldsymbol{\theta})$$
$$= \tau(\log 3) + 3\pi(s_1, a_0, \boldsymbol{\theta}) - 1, \tag{46}$$

because $\log 0.75 - \log 0.25 = \log(0.75/0.25) = \log 3$. For small values of τ, $\pi(s_1, a_0, \boldsymbol{\theta})$ should be close to $1/3$ to make $\boldsymbol{\theta}$ a stationary point.

Now let us write the partial derivative w.r.t. $\boldsymbol{\theta}(s_1, a_0)$, again assuming $\pi(s_0, a_0; \boldsymbol{\theta}) = 0.25$ and by noting that s_1 and s_2 are aliased:

$$\frac{\partial \tilde{J}_\mu(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1, a_0)} = \sum_{s \in \text{alias}(s_1)} m(s) \sum_a \frac{\partial \pi(a|s; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1, a_0)}\tilde{q}_\pi(s, a)$$

$$= m(s_1)\left[\frac{\partial \pi(s_1, a_0; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1, a_0)}\tilde{q}_\pi(s_1, a_0) + \frac{\partial \pi(s_1, a_1; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1, a_0)}\tilde{q}_\pi(s_1, a_1)\right]$$

$$+ m(s_2)\left[\frac{\partial \pi(s_2, a_0; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1, a_0)}\tilde{q}_\pi(s_2, a_0) + \frac{\partial \pi(s_2, a_1; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}(s_1, a_0)}\tilde{q}_\pi(s_2, a_1)\right]$$

$$= m(s_1)\left[\pi(s_1, a_0; \boldsymbol{\theta})(1 - \pi(s_1, a_0; \boldsymbol{\theta}))\tilde{q}_\pi(s_1, a_0)\right.$$

$$\left. - \pi(s_1, a_0; \boldsymbol{\theta})(1 - \pi(s_1, a_0; \boldsymbol{\theta}))\tilde{q}_\pi(s_1, a_1)\right]$$

$$+ m(s_2)\left[\pi(s_1, a_0; \boldsymbol{\theta})(1 - \pi(s_1, a_0; \boldsymbol{\theta}))\tilde{q}_\pi(s_2, a_0)\right.$$

$$\left. - \pi(s_1, a_0; \boldsymbol{\theta})(1 - \pi(s_1, a_0; \boldsymbol{\theta}))\tilde{q}_\pi(s_2, a_1)\right]$$

$$= m(s_1)\Big[\pi(s_1, a_0; \boldsymbol{\theta})(1 - \pi(s_1, a_0; \boldsymbol{\theta}))\tilde{q}_\pi(s_1, a_0)$$

$$- \pi(s_1, a_0; \boldsymbol{\theta})(1 - \pi(s_1, a_0; \boldsymbol{\theta}))\tilde{q}_\pi(s_1, a_1)\Big]$$

$$+ m(s_2)\Big[\pi(s_1, a_0; \boldsymbol{\theta})(1 - \pi(s_1, a_0; \boldsymbol{\theta}))\tilde{q}_\pi(s_2, a_0)$$

$$- \pi(s_1, a_0; \boldsymbol{\theta})(1 - \pi(s_1, a_0; \boldsymbol{\theta}))\tilde{q}_\pi(s_2, a_1)\Big]$$

$$= \pi(s_1, a_0; \boldsymbol{\theta})(1 - \pi(s_1, a_0; \boldsymbol{\theta}))$$

$$\Big[m(s_1)\big(\tilde{q}_\pi(s_1, a_0) - \tilde{q}_\pi(s_1, a_1)\big) + m(s_2)\big(\tilde{q}_\pi(s_2, a_0) - \tilde{q}_\pi(s_2, a_1)\big)\Big].$$

Making the partial derivative zero requires either $\pi(s_1, a_0; \boldsymbol{\theta})$, $1 - \pi(s_1, a_0; \boldsymbol{\theta})$, or the contents of the brackets to be zero. The first two are incompatible with the requirement for making Equation 46 zero so we will continue with the third one. Note that

$$m(s_1) = d_\mu(s_1) + d_\mu(s_0)\pi(s_0, a_0; \boldsymbol{\theta}) = 0.125 + 0.5 * 0.25 = 0.25,$$
$$m(s_2) = d_\mu(s_2) + d_\mu(s_0)\pi(s_0, a_1; \boldsymbol{\theta}) = 0.375 + 0.5 * 0.75 = 0.75,$$

and the factor in the brackets becomes

$$m(s_1)\big(\tilde{q}_\pi(s_1, a_0) - \tilde{q}_\pi(s_1, a_1)\big) + m(s_2)\big(\tilde{q}_\pi(s_2, a_0) - \tilde{q}_\pi(s_2, a_1)\big)$$
$$= 0.25\big(2 - \tau \log \pi(s_1, a_0, \boldsymbol{\theta}) + \tau \log \pi(s_1, a_1, \boldsymbol{\theta})\big)$$
$$+ 0.75\big(-\tau \log \pi(s_2, a_0, \boldsymbol{\theta}) - 1 + \tau \log \pi(s_2, a_1, \boldsymbol{\theta})\big)$$
$$= -0.25 + \tau\big(-\log \pi(s_1, a_0, \boldsymbol{\theta}) + \log \pi(s_1, a_1, \boldsymbol{\theta})\big)$$
$$= -0.25 + \tau\big(\log(1 - \pi(s_1, a_0, \boldsymbol{\theta})) - \log \pi(s_1, a_0, \boldsymbol{\theta})\big)$$
$$= -0.25 + \tau \log\Big(\frac{1 - \pi(s_1, a_0, \boldsymbol{\theta})}{\pi(s_1, a_0, \boldsymbol{\theta})}\Big).$$

Making this zero is also at odds with the requirement for Equation 46. To see why, let's consider both conditions, where we use $p = \pi(s_1, a_0, \boldsymbol{\theta})$ to simplify notation.

$$\tau \log 3 + 3p - 1 = 0$$
$$-0.25 + \tau \log \frac{1 - p}{p} = 0$$

The first equation requires $p = (1 - \tau \log 3)/3$. The second equation requires $p = (\exp(0.25/\tau) + 1)^{-1}$. These two equations intersect when $\tau_i \approx 0.2779$, but otherwise do not equal each other, meaning we cannot satisfy both of these criteria. Therefore, for any $\tau \neq \tau_i$, a stationary point $\boldsymbol{\theta}$ under the true gradient cannot have $\pi(s_0, a_0, \boldsymbol{\theta}) = \mu(s_0, a_0) = 0.25$ and thus cannot be a stationary point under semi-gradient. ∎

This counterexample shows one environment where the sets of stationary points under the true gradient is disjoint from the set of stationary points under the semi-gradient.

ACE(η)	Actor-Critic with Emphatic weightings. η interpolates between semi-gradient updates ($\eta = 0$) and gradient updates ($\eta = 1$).
OffPAC	Off-Policy Actor-Critic (Degris et al., 2012b). Equivalent to ACE(0).
GTD(λ)	Gradient Temporal-Difference learning (Sutton et al., 2009). λ is the decay rate of the eligibility trace vector.
ETD(λ)	Emphatic Temporal-Difference learning (Sutton et al., 2016). λ is the decay rate of the eligibility trace vector.
TDRC(λ)	Temporal-Difference learning with Regularized Corrections (Ghiassian et al., 2020; Patterson et al., 2022). λ is the decay rate of the eligibility trace vector.
DPG	Deterministic Policy Gradient (Silver et al., 2014). Uses a deterministic policy parameterization and semi-gradient updates.
True-DPGE	A version of DPG using true gradient updates (i.e., scaling the update by the true emphatic weightings).
True-ACE	A version of ACE using the true emphatic weightings.
ACE-direct	A version of ACE that uses the direct method from Section 5.2.2 to estimate the emphatic weightings.
ACE-trace	A version of ACE that uses the emphatic trace from Sutton et al. (2016) to estimate the emphatic weightings.
ACE-ideal	A version of ACE that re-computes the emphatic trace on each time step using the current policy to remove the influence of old versions of the policy.

Table 1: Descriptions of the algorithms used in the experiments.

Appendix F. Descriptions of Algorithms in Section 9

References

Zafarali Ahmed, Nicolas Le Roux, Mohammad Norouzi, and Dale Schuurmans. Understanding the impact of entropy on policy optimization. In *International Conference on Machine Learning*, 2019.

Andrew G. Barto, Richard S. Sutton, and Charles W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, (5), 1983.

Shalabh Bhatnagar, Richard S. Sutton, Mohammad Ghavamzadeh, and Mark Lee. Incremental natural actor–critic algorithms. In *Conference on Neural Information Processing Systems*, 2007.

Shalabh Bhatnagar, Richard S. Sutton, Mohammad Ghavamzadeh, and Mark Lee. Natural actor–critic algorithms. *Automatica*, 45(11), 2009.

Justin Boyan and Andrew W. Moore. Generalization in reinforcement learning: Safely approximating the value function. *Conference on Neural Information Processing Systems*, pages 369–376, 1995.

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

Maxime Chevalier-Boisvert, Lucas Willems, and Suman Pal. Minimalistic gridworld environment for OpenAI Gym. `https://github.com/maximecb/gym-minigrid`, 2018.

Thomas Degris, Patrick M. Pilarski, and Richard S. Sutton. Model–free reinforcement learning with continuous action in practice. In *American Control Conference*, 2012a.

Thomas Degris, Martha White, and Richard S. Sutton. Off–policy actor–critic. In *International Conference on Machine Learning*, 2012b.

Travis B. Dick. Policy gradient reinforcement learning without regret. Master's thesis, University of Alberta, 2015.

Le Tien Dung, Takashi Komeda, and Motoki Takagi. Reinforcement learning in non-Markovian environments using automatic discovery of subgoals. In *Society of Instrument and Control Engineers Annual Conference*, pages 2601–2605. IEEE, 2007.

Charles R. Harris et al. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL `https://doi.org/10.1038/s41586-020-2649-2`.

Matthieu Geist, Bruno Scherrer, and Olivier Pietquin. A theory of regularized markov decision processes. In *International Conference on Machine Learning*, pages 2160–2169, 2019.

Carles Gelada and Marc G. Bellemare. Off-policy deep reinforcement learning by bootstrapping the covariate shift. In *AAAI Conference on Artificial Intelligence*, volume 33, 2019.

Sina Ghiassian, Andrew Patterson, Martha White, Richard S. Sutton, and Adam White. Online off-policy prediction. *arXiv:1811.02597*, 2018.

Sina Ghiassian, Andrew Patterson, Shivam Garg, Dhawal Gupta, Adam White, and Martha White. Gradient temporal-difference learning with regularized corrections. In *International Conference on Machine Learning*, pages 3524–3534. PMLR, 2020.

Dibya Ghosh, Marlos C. Machado, and Nicolas Le Roux. An operator view of policy gradient methods. In *Conference on Neural Information Processing Systems*, 2020.

Evan Greensmith, Peter L. Bartlett, and Jonathan Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. *The Journal of Machine Learning Research*, 5, 2004.

Ivo Grondman, Lucian Busoniu, Gabriel A.D. Lopes, and Robert Babuska. A survey of actor–critic reinforcement learning: Standard and natural policy gradients. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6), 2012.

Shixiang Gu, Tim Lillicrap, Richard E. Turner, Zoubin Ghahramani, Bernhard Schölkopf, and Sergey Levine. Interpolated policy gradient: Merging on–policy and off–policy gradient estimation for deep reinforcement learning. In *Conference on Neural Information Processing Systems*, 2017a.

Shixiang Gu, Timothy Lillicrap, Zoubin Ghahramani, Richard E. Turner, and Sergey Levine. Q–prop: Sample–efficient policy gradient with an off–policy critic. In *International Conference on Learning Representations*, 2017b.

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

Assaf Hallak and Shie Mannor. Consistent on-line off-policy evaluation. In *International Conference on Machine Learning*, volume 70, 2017.

Bojun Huang. Steady state analysis of episodic reinforcement learning. In *Conference on Neural Information Processing Systems*, 2020.

Ehsan Imani, Eric Graves, and Martha White. An off–policy policy gradient theorem using emphatic weightings. In *Conference on Neural Information Processing Systems*, 2018.

Nathan Kallus and Masatoshi Uehara. Statistically efficient off–policy policy gradients. In *International Conference on Machine Learning*, 2020.

Andreas Kirsch. Mdp environments for the OpenAI Gym. Technical report, 2017. URL https://arxiv.org/abs/1709.09069.

Martin Klissarov and Doina Precup. Flexible option learning. *Conference on Neural Information Processing Systems*, pages 4632–4646, 2021.

Vijay R. Konda and John N. Tsitsiklis. Actor–critic algorithms. In *Conference on Neural Information Processing Systems*, 2000.

Romain Laroche and Remi Tachet des Combes. Dr jekyll & mr hyde: the strange case of off-policy policy updates. *Conference on Neural Information Processing Systems*, 2021.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations*, 2015.

Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine learning*, 8(3-4), 1992.

Long-Ji Lin. Reinforcement learning for robots using neural networks. Technical report, Carnegie-Mellon University, 1993.

Qiang Liu, Lihong Li, Ziyang Tang, and Dengyong Zhou. Breaking the curse of horizon: Infinite-horizon off-policy estimation. In *Conference on Neural Information Processing Systems*, 2018.

Yao Liu, Adith Swaminathan, Alekh Agarwal, and Emma Brunskill. Off-policy policy gradient with stationary distribution correction. In *Conference on Uncertainty in Artificial Intelligence*, 2019.

Yao Liu, Pierre-Luc Bacon, and Emma Brunskill. Understanding the curse of horizon in off-policy evaluation via conditional importance sampling. In *International Conference on Machine Learning*, pages 6184–6193. PMLR, 2020a.

Yao Liu, Adith Swaminathan, Alekh Agarwal, and Emma Brunskill. Off–policy policy gradient with stationary distribution correction. In *Conference on Uncertainty in Artificial Intelligence*, 2020b.

John Loch and Satinder P. Singh. Using eligibility traces to find the best memoryless policy in partially observable Markov decision processes. In *International Conference on Machine Learning*, pages 323–331, 1998.

Hamid Maei. *Gradient Temporal–Difference Learning Algorithms*. PhD thesis, University of Alberta, 2011.

Hamid Reza Maei. Convergent actor–critic algorithms under off-policy training and function approximation. *arXiv:1802.07842*, 2018.

A. Rupam Mahmood, Huizhen Yu, Martha White, and Richard S. Sutton. Emphatic temporal-difference learning. *arXiv:1507.01569*, 2015.

Peter Marbach and John N. Tsitsiklis. Simulation–based optimization of Markov reward processes. *IEEE Transactions on Automatic Control*, 46(2), 2001.

Jincheng Mei, Chenjun Xiao, Csaba Szepesvari, and Dale Schuurmans. On the global convergence rates of softmax policy gradient methods. In *International Conference on Machine Learning*, 2020.

Nicolas Meuleau, Leonid Peshkin, Leslie P. Kaelbling, and Kee-Eung Kim. Off–policy policy search. Technical report, MIT Artificial Intelligence Laboratory, 2000.

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), 2015.

Andrew William Moore. *Efficient memory-based learning for robot control*. PhD thesis, University of Cambridge, 1990.

Chris Nota and Philip S. Thomas. Is the policy gradient a gradient? In *International Conference on Autonomous Agents and Multiagent Systems*, 2020.

Andrew Patterson, Adam White, and Martha White. A generalized projected bellman error for off-policy value estimation in reinforcement learning. *The Journal of Machine Learning Research*, 2022.

Jan Peters, Sethu Vijayakumar, and Stefan Schaal. Natural actor-critic. In *European Conference on Machine Learning*, 2005.

Doina Precup. *Temporal Abstraction In Reinforcement Learning*. PhD thesis, University of Massachusetts Amherst, 2000.

Doina Precup, Richard S. Sutton, and Satinder P. Singh. Eligibility Traces for Off–Policy Policy Evaluation. In *International Conference on Machine Learning*, 2000.

Doina Precup, Richard S. Sutton, and Sanjoy Dasgupta. Off–policy temporal–difference learning with function approximation. *International Conference on Machine Learning*, 2001.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *International Conference on Learning Representations*, 2016.

Mark Schmidt and Nicolas Le Roux. Fast convergence of stochastic gradient descent under a strong growth condition. *arXiv:1308.6370*, 2013.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International Conference on Machine Learning*, 2014.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. 2nd edition, 2018.

Richard S. Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Conference on Neural Information Processing Systems*, 1999a.

Richard S. Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112 (1-2), 1999b.

Richard S. Sutton, Hamid Reza Maei, Doina Precup, Shalabh Bhatnagar, David Silver, Csaba Szepesvári, and Eric Wiewiora. Fast gradient-descent methods for temporal–difference learning with linear function approximation. In *International Conference on Machine Learning*, 2009.

Richard S. Sutton, Joseph Modayil, Michael Delp, Thomas Degris, Patrick M. Pilarski, Adam White, and Doina Precup. Horde: A scalable real–time architecture for learning knowledge from unsupervised sensorimotor interaction. In *International Conference on Autonomous Agents and MultiAgent Systems*, 2011.

Richard S. Sutton, A. Rupam Mahmood, and Martha White. An emphatic approach to the problem of off–policy temporal–difference learning. *The Journal of Machine Learning Research*, 17, 2016.

Philip Thomas. Bias in natural actor–critic algorithms. In *International Conference on Machine Learning*, 2014.

Samuele Tosatto, Joao Carvalho, Hany Abdulsamad, and Jan Peters. A nonparametric off–policy policy gradient. In *International Conference on Artificial Intelligence and Statistics*, 2020.

Ziyu Wang, Victor Bapst, Nicolas Heess, Volodymyr Mnih, Rémi Munos, Koray Kavukcuoglu, and Nando de Freitas. Sample efficient actor–critic with experience replay. In *International Conference on Learning Representations*, 2016.

Christopher J.C.H. Watkins and Peter Dayan. Q–learning. *Machine Learning*, 8(3-4), 1992.

Lex Weaver and Nigel Tao. The optimal reward baseline for gradient–based reinforcement learning. In *Conference on Uncertainty in Artificial Intelligence*, 2001.

Adam White. *Developing A Predictive Approach To Knowledge*. PhD thesis, University of Alberta, 2015.

Martha White. Unifying task specification in reinforcement learning. In *International Conference on Machine Learning*, 2017.

Ronald J. Williams. Simple statistical gradient–following algorithms for connectionist reinforcement learning. *Machine Learning*, 8, 1992.

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

Ian H. Witten. An adaptive optimal controller for discrete–time Markov environments. *Information and Control*, 34(4), 1977.

Huizhen Yu. On convergence of emphatic temporal–difference learning. In *Conference on Learning Theory*, 2015.

Shangtong Zhang, Wendelin Boehmer, and Shimon Whiteson. Generalized off-policy actor-critic. In *Conference on Neural Information Processing Systems*, 2019.

Shangtong Zhang, Bo Liu, Hengshuai Yao, and Shimon Whiteson. Provably convergent two–timescale off-policy actor–critic with function approximation. In *International Conference on Machine Learning*, 2020.