Convergence of a Human-in-the-Loop Policy-Gradient Algorithm With Eligibility Trace Under Reward, Policy, and Advantage Feedback

Ishaan Shah [*1] **David Halpern** [*1] **Kavosh Asadi** [1] **Michael L. Littman** [1]

Abstract

Abstract—Fluid human–agent communication is essential for the future of human-in-the-loop reinforcement learning. An agent must respond appropriately to feedback from its human trainer even before they have significant experience working together. Therefore, it is important that learning agents respond well to various feedback schemes human trainers are likely to provide. This work analyzes the COnvergent Actor–Critic by Humans (COACH) algorithm under three different types of feedback—policy feedback, reward feedback, and advantage feedback. For these three feedback types, we find that COACH can behave sub-optimally. We propose a variant of COACH, episodic COACH (E-COACH), which we prove converges for all three types. We compare our COACH variant with two other reinforcement-learning algorithms: Q-learning and TAMER.

1. Introduction

We study the algorithm COACH (MacGlashan et al., 2017a), designed to learn from evaluative feedback. We would like for the algorithm to find an optimal policy under different feedback schemes, since a human trainer is apt to select from several possible approaches and we do not know which will be chosen *a priori*.

We present a proof of convergence for three natural feedback schemes. 1) Feedback can take the form of an economic incentive in which the learner gets an immediate reward for moving into a state based on the state's desirability—one-step **reward**. 2) Feedback can be a binary signal that tells the learner whether the action it took was correct (1) or not (0) with respect to the trainer's intended **policy**. And, 3) feedback can reveal how good an action was relative to the agent's recent behavior—the action's **advantage**. It is desirable for a learning algorithm to perform appropriately in all three of these settings.

E-COACH (Algorithm 1) is such a learning algorithm. It takes input policy π_θ, discount factor γ, and a learning rate α.

Algorithm 1 E-COACH $\langle \pi_\theta, \gamma, \alpha \rangle$

$\theta_0 \leftarrow 0$
for episode $= 0, 1, 2, \ldots$ **do**
  $e_0 \leftarrow 0$
  for $t = 0, 1, 2, \ldots$ **do**
    $a_t \sim \pi_\theta(s_t, \cdot)$
    observe state s_{t+1} and human feedback f_{t+1}
    $e_{t+1} \leftarrow e_t + \frac{1}{\pi_\theta(s_t, a_t)} \nabla_\theta \pi_\theta(s_t, a_t)$
    $\theta_{t+1} \leftarrow \theta_t + \alpha \gamma^t e_{t+1} f_{t+1}$
  end for
end for

E-COACH (Episodic COACH) is a close variant of the original COACH with a few differences. 1) It keeps track explicitly of the number of steps t elapsed in the current episode. 2) E-COACH's most notable difference from COACH is E-COACH's use of a γ^t decay factor. This element emphasizes information from temporally closer decisions over more distant ones. 3) In addition, E-COACH does not use a λ parameter to decay the eligibility trace e_t. This makes E-COACH's treatment of eligibility traces like setting $\lambda = 1$ in the original COACH algorithm.

We propose E-COACH instead of COACH because COACH does not take advantage of the discount factor, γ^t. This causes it to incorrectly estimate the expected reward, causing it to perform poorly on the given environment. We provide an example of such a scenario in 6.1. In contrast to COACH, we show that E-COACH can find converge under all three feedback schemes described above.

2. Background

A Markov Decision Process (MDP) is a five-tuple: $\langle S, A, T, R, \gamma \rangle$. Here, S is a set of reachable states, A is the

[*]Equal contribution [1]Department of Computer Science, Brown University, Providence, Rhode Island, USA. Correspondence to: Ishaan Shah <ishah@cra.com>, David Halpern <david_halpern@alumni.brown.edu>.

Presented at ICML 2021 workshops Human-AI Collaboration in Sequential Decision-Making and Human in the Loop Learning.

set of actions an agent might use, $T(s' \mid s, a)$ is a probability that the agent would move to state s' from the given state s having taken action a, $R(s, a)$ is the reward obtained for taking action a from state s, and $\gamma \in [0, 1)$ is a discount factor indicating the importance of immediate rewards as opposed to rewards received in distant future.

A stochastic policy $\pi_\theta : S \times A \rightarrow [0, 1]$, where $\sum_{a \in A} \pi_\theta(s, a) = 1, \forall\, s \in S$, defines an agent's behavior via $\pi_\theta(s, a) = \mathbb{P}\{a_t = a \mid s_t = s, \theta\}, \forall\, s \in S, a \in A$. Note that θ is a vector parameter of the policy, and we assume that π is differentiable with respect to this parameter. For brevity, we will denote $\pi_\theta(s, a)$ as $\pi(s, a)$ when the parameter vector is clear from context.

The value functions Q^π and V^π measure the performance of policy π:

$$Q^\pi(s, a) = \mathbb{E}\big[\sum_{k=1,2,\ldots} \gamma^{k-1} r_{t+k} \mid s_t = s, a_t = a, \pi\big]$$

and

$$V^\pi(s) = \mathbb{E}_{a \sim \pi(s,\cdot)}[Q^\pi(s, a) \mid s, \pi].$$

When an agent's policy $\pi^* = \text{argmax}_\pi V^\pi, \forall s \in S$, then we call that policy *optimal*. We will denote optimal policies as π^* and use the shorthand $V^*(s) = V^{\pi^*}(s)$. Also, for sake of brevity, we will write $\mathbb{E}[\cdot \mid \pi]$ simply as $\mathbb{E}[\cdot]$ from now on. Note that all expectations we consider are conditioned on the policy. If not specified otherwise , $\mathbb{E}[\cdot]$ is an expectation over $s_1, a_1, s_2, a_2 \ldots$ where $s_{t+1} \sim \mathrm{T}(\cdot \mid s_t, a_t)$ and $a_{t+1} \sim \pi(s_{t+1}, \cdot)$.

3. E-COACH Under Reward Feedback

A simple form of feedback a trainer may choose to give a learner is the one-step reward obtained from the MDP for the action the agent just took. Such *reward feedback* is convenient since it is myopic and does not require the trainer to consider future rewards. It assumes a direct analogy between the rewards that define the task and the feedback provided by the trainer—it is the simplest extension of standard reinforcement learning to the interactive setting. For the following theoretical results to hold, we assume the human-trainer gives consistent reward, as per the definition of our feedback, f. f represents our feedback, which we will redefine in Sections 3, 4.2, and 5.

We look at an MDP $M = \langle S, A, R, T, \gamma \rangle$. Under reward feedback, when an agent takes an action a in state s, the trainer gives feedback

$$f(s, a) = R(s, a).$$

Theorem 1: E-COACH converges under reward feedback $f(s, a) = R(s, a), \forall\, s \times a \in S \times A$.

Proof: Consider the sequence of updates on e_t and θ_t at each time step t:

$$e_{t+1} \leftarrow e_t + \frac{1}{\pi_\theta(s_t, a_t)} \nabla_\theta \pi_\theta(s_t, a_t)$$
$$\theta_{t+1} \leftarrow \theta_t + \gamma^t e_{t+1} r_{t+1}$$

where, for brevity, we define $r_{t+1} = R(s_t, a_t)$.

To better understand what the updates mean, consider some terminal time L. The value L may refer to the time at which an agent reaches the goal or a pre-decided time at which the trainer stops the agent. We use L only for the purpose of elucidation and the analysis below also extends to the infinite horizon case when L is unbounded. We ignore the α in the θ update above for sake of clarity. By linearity of the updates, it is trivial to incorporate α into the calculations below.

$$\begin{aligned}\theta_{L+1} &= \Sigma_{\tau=0}^{L} \gamma^\tau e_{\tau+1} r_{\tau+1} \\ &= \Sigma_{\tau=0}^{L} \gamma^\tau r_{\tau+1} (\Sigma_{t=0}^{\tau} \frac{\nabla_\theta \pi_\theta(s_t, a_t)}{\pi_\theta(s_t, a_t)}) \\ &= \Sigma_{\tau=0}^{L} \Sigma_{t=0}^{\tau} \gamma^\tau r_{\tau+1} \frac{\nabla_\theta \pi_\theta(s_t, a_t)}{\pi_\theta(s_t, a_t)}\end{aligned}$$

Rearranging the order of summation

$$\begin{aligned}\theta_{L+1} &= \Sigma_{t=0}^{L} \Sigma_{\tau=t}^{L} \frac{\nabla_\theta \pi_\theta(s_t, a_t)}{\pi_\theta(s_t, a_t)} \gamma^\tau r_{\tau+1} \\ &= \Sigma_{t=0}^{L} \gamma^t \frac{\nabla_\theta \pi_\theta(s_t, a_t)}{\pi_\theta(s_t, a_t)} (\Sigma_{\tau=0}^{L-t} \gamma^\tau r_{\tau+t+1})\end{aligned}$$

Taking expectation

$$\mathbb{E}[\theta_{L+1}] = \Sigma_{t=0}^{L} \gamma^t \mathbb{E}[\frac{\nabla_\theta \pi_\theta(s_t, a_t)}{\pi_\theta(s_t, a_t)} Q^\pi(s_t, a_t)]$$

3.1. E-COACH Objective Function

In this section we show that the gradient of the objective function $\Sigma_{t=0}^{\infty} \gamma^t \mathbb{E}[\frac{\nabla_\theta \pi_\theta(s_t, a_t)}{\pi_\theta(s_t, a_t)} Q^\pi(s_t, a_t)]$ is what REINFORCE performs gradient ascent on. Note that this quantity is what E-COACH is estimating (via θ_L parameter) and performing gradient ascent on.

Consider the REINFORCE algorithm (Algorithm 2). Here, G_t is a Monte Carlo estimate of $Q^\pi(s_t, a_t)$. Hence, at any terminal time L, it is clear that the expected value of θ_L obtained by REINFORCE is equal to that of E-COACH.

Consider the unnormalised state visitation distribution, such that $\forall s \in S, d^\pi(s) = \mathbb{P}_0^\pi(s) + \gamma \mathbb{P}_1^\pi(s) + \ldots + \gamma^i \mathbb{P}_i^\pi(s) + \ldots$ where $\mathbb{P}_t^\pi(s)$ denotes the probability of arriving in state s at time t following policy π. The objective to maximize, as described in the Policy Gradient Theorem (Sutton et al., 2000), is

$$\rho^\pi = \Sigma_s d^\pi(s) \Sigma_a \pi(s, a) R(s, a),$$

Algorithm 2 REINFORCE$< \pi_\theta, \gamma, \alpha >$

Generate an episode $s_0, a_0, r_1, \ldots s_L, a_L, r_{L+1}$
for $t = 0, 1, 2, \ldots$ **do**
  G_t = return from step t
  $\theta_{t+1} \leftarrow \theta_t + \gamma^t G_t \nabla_\theta \log(\pi_\theta(s_t, a_t))$
end for

and its gradient is

$$\nabla_\theta \rho^\pi = \Sigma_s d^\pi(s) \Sigma_a \nabla_\theta \pi(s,a) Q^\pi(s,a).$$

The gradient can be rewritten as $\nabla_\theta \rho^\pi = \mathbb{E}_{s\sim d^\pi, a\sim\pi(s,\cdot)}[\frac{\nabla_\theta \pi_\theta(s_t,a_t)}{\pi_\theta(s_t,a_t)} Q^\pi(s,a)]$.

Expanding with respect to $d^\pi(s)$ yields

$$\mathbb{E}_{s\sim d^\pi, a\sim\pi(s,\cdot)}[\frac{\nabla_\theta \pi_\theta(s_t,a_t)}{\pi_\theta(s_t,a_t)} Q^\pi(s,a)]$$
$$= \Sigma_{t=0}^\infty \gamma^t \mathbb{E}_{s_t\sim\mathbb{P}_t^\pi, a_t\sim\pi_\theta(s_t,\cdot)}[\frac{\nabla_\theta \pi_\theta(s_t,a_t)}{\pi_\theta(s_t,a_t)} Q^\pi(s_t,a_t)]$$

which is exactly what E-COACH and REINFORCE are estimating. Since E-COACH is performing gradient ascent on the policy gradient objective, we can use results from (Agarwal et al., 2020) to say that E-COACH converges to local optima or saddle points. Although, recent work has shown that policy gradient methods can escape saddle points under mild assumptions on the rewards and minor modifications to existing algorithms (Zhang et al., 2020) (Jin et al., 2017).

□

Something to note is that behavioral evidence indicates that one-step reward is not a typical choice of human trainers (Ho et al., 2019).

4. E-COACH Under Policy Feedback

To argue that E-COACH converges under policy feedback, we first consider a more general form of feedback and then show policy feedback is a special case.

4.1. E-COACH with a More General Type of Feedback

Let us start by considering two similar MDPs $M_1 = \langle S, A, R, T, \gamma \rangle$ and $M_2 = \langle S, A, f, T, \gamma \rangle$. Note the differing reward functions R and f in the two MDPs.

We will denote the value functions for M_1 and M_2 as V_1 and V_2, respectively. We will say that the starting state for both of our MDPs is s_0. Define $V_1^{\min} = \min_{\pi\in\Pi} V_1^\pi(s_0)$, $V_1^* = \max_{\pi\in\Pi} V_1^\pi(s_0)$.

The following theorem will have the following assumption:

1. E-COACH (see algorithm 1) will give us a policy $\pi_2(s,a)$ such that $\mathbb{E}_{s\sim d^{\pi_2^*}}\big[\sum_a \big|\pi_2^*(s,a) - \pi_2(s,a)\big|\big] \leq \delta$ for some optimal policy π_2^* on the domain M_2. The proof in section 3 strengthens this assumption by showing that E-COACH optimizes the policy gradient objective. Note π_2^* may not be the only optimal policy; instead, it is a single optimal policy.

2. We also assume that $\gamma \neq 1$ for the case where the MDP has an infinite horizon, which will we will justify later on.

Theorem 2 requires the condition that all optimal policies for M_2 are also optimal for M_1. We will later show that this condition holds true for the case of policy feedback in theorem 3, allowing us to leverage these results.

Theorem 2: If all optimal policies for M_2 are also optimal for M_1 (optimal policies of M_2 are a subset of those for M_1), then running E-COACH on M_2 will result in a policy that is close to an optimal policy on M_2, which will also be close to an optimal policy for M_1. Let's define $W = \max(|V_1^*|, |V_1^{\min}|)$. Then we find that,

$$0 \leq V_1^* - V_1^{\pi_2} \leq W\delta$$

Proof: We have to show that running E-COACH in M_2 will yield a policy that is not too far off from an optimal policy for M_1. We would like to run E-COACH on M_2, using the alternate form of feedback as the reward function, and for any good policy (as per assumption 1) we get from E-COACH on M_2, we would like for that policy to also be good on M_1, the original MDP we are trying to solve.

The lower-bound in the theorem statement is immediate.

For the upper-bound, let's let $\pi^{(n)}$ denote a policy that follows/simulates π_2^* for the first $n-1$ time-steps and π_2 for the rest. Hence, on the n^{th} time-step, $\pi^{(n)}$ will follow/simulate π_2 and not π_2^*. Let $V^{(n)}$ denote the value of policy $\pi^{(n)}$. Therefore, we can say that $V_1^{\pi_2} = V^{(0)}$ and $V_1^* = V^{(\infty)}$. Remember that π_2^* is optimal on M_1 *and* M_2 by the condition above, and thus has value V^*.

We'll start by considering $V^{(t)} - V^{(t-1)}$. Both $\pi^{(t)}$ and $\pi^{(t-1)}$ accumulate the same expected reward for the first $t-2$ steps and so these rewards cancel out. Note that the $\mathbb{P}$ we use below is the same as that defined in section 3.1. We find the following:

$$\begin{aligned}
V^{(t)} - V^{(t-1)} &= \gamma^{t-1} \sum_s \mathbb{P}_{t-1}^{\pi_2^*}(s) \sum_a \pi_2^*(s,a) Q^{\pi_2}(s,a) \\
&\quad - \gamma^{t-1} \sum_s \mathbb{P}_{t-1}^{\pi_2^*}(s) \sum_a \pi_2(s,a) Q^{\pi_2}(s,a) \\
&= \gamma^{t-1} \sum_s \mathbb{P}_{t-1}^{\pi_2^*}(s) \sum_a (\pi_2^*(s,a) - \pi_2(s,a)) Q^{\pi_2}(s,a) \\
&\leq \gamma^{t-1} \sum_s \mathbb{P}_{t-1}^{\pi_2^*}(s) \sum_a |\pi_2^*(s,a) - \pi_2(s,a)| W
\end{aligned}$$

Now we'll use the above fact when considering $V_1^* - V_1^{\pi_2}$.

$$\begin{aligned}
V_1^* - V_1^{\pi_2} =& (V^1 - V^0) + (V^2 - V^1) + (V^3 - V^2) + \cdots \\
\leq& \sum_i^\infty \gamma^i \sum_s \mathbb{P}_i^{\pi_2^*}(s) \sum_a |\pi_2^*(s,a) - \pi_2(s,a)| W \\
=& \sum_s \sum_i^\infty \gamma^i \mathbb{P}_i^{\pi_2^*}(s) \sum_a |\pi_2^*(s,a) - \pi_2(s,a)| W \\
=& \sum_s d^{\pi_2^*}(s) \sum_a |\pi_2^*(s,a) - \pi_2(s,a)| W \\
=& W \mathbb{E}_{s \sim d^{\pi_2^*}} \big[\sum_a |\pi_2^*(s,a) - \pi_2(s,a)| \big] \\
\leq& W\delta
\end{aligned}$$

As $\delta \to 0$, we have that $V_1^* - V_1^{\pi_2} \to 0$.

□

Note that our theorem says something different than the Simulation Lemma (Kearns & Singh, 1998) as we make no assumptions about how close the reward functions of M_1 and M_2 are. Instead our theorem requires optimal policies in M_2 be optimal in M_1 and bounds the return of a policy learnt by E-COACH in M_2.

4.2. E-COACH Under Policy Feedback

Let $M_1 = \langle S, A, R, T, \gamma \rangle$ be an MDP without any specific reward function. Under *policy feedback*, a trainer has a target stationary deterministic policy π_1^* in mind and delivers feedback based on whether the trainer's decision agrees with π_1^*. When an agent takes an action a in state s, the trainer will give feedback

$$f(s,a) = I(s,a),$$

with $I(s,a)$ defined as,

$$I(s,a) = \begin{cases} 1, & \text{if } \pi_1^*(s) = a, \\ 0, & \text{otherwise.} \end{cases}$$

Theorem 3: E-COACH converges under feedback $f(s,a) = I(s,a), \forall\, s \times a \in S \times A$.

Proof: Consider the case of replacing the reward function $R(s,a)$ with $I(s,a)$ in MDP M_1, constructing a new MDP $M_2 = \langle S, A, f, T, \gamma \rangle$. We would like to show that, in this setting, the E-COACH algorithm converges to the optimal solution. M_1 and M_2 satisfy the prerequisites for theorem 2.

Consider the optimal policy for M_2. The best policy will select the best action in every state. We have that $V_2^*(s_0) = \sum_{i=0}^\infty 1 \cdot \gamma^i$. The optimal policy for M_2 will achieve this value function because, if not, then we have a policy such that $V_2'(s_0) = \sum_{i=0}^\infty t(i) \cdot 1 \cdot \gamma^i$, where $t(i) \in \{0,1\} \forall\, i$ and $t(j) = 0$ for some j. Take the smallest value $k \in \mathbb{N}$ such that $t(k) = 0$, then $V_2^*(s_0) - V_2'(s_0) \geq \gamma^k$ so then the policy achieving V_2' is sub-optimal. We can conclude that $V_2^*(s_0)$ is the value function for the optimal policy. Therefore, the policy that always chooses the action that gives a value of one is optimal. Also, note that always choosing the action that results in a feedback of one corresponds exactly to the decision of π_1^* by construction of $f(s,a)$. So, we obtain that $\pi_1^*(s,a) = \pi_2^*(s,a), \forall\, (s,a) \in S \times A$. In other words, an optimal policy in the new domain is equivalent to the target policy from the original one.

We can leverage Theorem 2 to show that the algorithm converges under policy feedback.

□

5. E-COACH Under Advantage Feedback

COACH (MacGlashan et al., 2017a) was originally motivated by the observation that human feedback is observed to be policy dependent—if a decision *improves* over the agent's recent decisions, trainers provide positive feedback. If it is worse, the trainer is more likely to provide negative feedback. As such, feedback is well modeled by the advantage function of the agent's current policy.

In *advantage feedback*, when an agent takes action a in state s, the trainer will give feedback

$$f(s,a) = A^\pi(s,a),$$

with $A^\pi(s,a)$ defined as,

$$A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s).$$

Theorem 4: E-COACH converges under feedback $f(s,a) = A^\pi(s,a), \forall\, s \times a \in S \times A$.

Proof: Since $V^\pi(s) = \mathbb{E}_{a \sim \pi(\cdot|s)}[Q^\pi(s,a)\,|s]$, we have that

$$\begin{aligned}
&\mathbb{E}_{a \sim \pi(s,\cdot)}[A^\pi(s,a) \,|\, s] \\
&\quad = \mathbb{E}_{a \sim \pi(s,\cdot)}[Q^\pi(s,a) \,|\, s] - V^\pi(s) \\
&\quad = 0
\end{aligned}$$

By the equation above we can say the following

$$\mathbb{E}_{s_{t+1}, a_{t+1}, s_{t+2} \ldots}[A^\pi(s_{t+\tau}, a_{t+\tau}) | s_t, a_t] = 0 \;\forall \tau > 0 \quad (1)$$

We also know that

$$\mathbb{E}_{s_{t+1}, a_{t+1}, s_{t+2} \ldots}[A^\pi(s_t, a_t) | s_t, a_t] = A^\pi(s_t, a_t) \quad (2)$$

We will use equations 1 and 2 later on in this proof.

Using the same approach as in theorem 1, we look at the sequence of updates made to the policy parameter θ until some terminal time L.

$$\theta_{L+1} = \Sigma_{\tau=0}^{L} \gamma^{\tau} e_{\tau+1} A^{\pi}(s_\tau, a_\tau)$$
$$= \Sigma_{\tau=0}^{L} \gamma^{\tau} A^{\pi}(s_\tau, a_\tau)(\Sigma_{t=0}^{\tau} \frac{\nabla_\theta \pi_\theta(s_t, a_t)}{\pi_\theta(s_t, a_t)})$$
$$= \Sigma_{\tau=0}^{L} \Sigma_{t=0}^{\tau} \gamma^{\tau} A^{\pi}(s_\tau, a_\tau) \frac{\nabla_\theta \pi_\theta(s_t, a_t)}{\pi_\theta(s_t, a_t)}$$

Rearranging the order of summation

$$\theta_{L+1} = \Sigma_{t=0}^{L} \Sigma_{\tau=t}^{L} \frac{\nabla_\theta \pi_\theta(s_t, a_t)}{\pi_\theta(s_t, a_t)} \gamma^{\tau} A^{\pi}(s_\tau, a_\tau)$$
$$= \Sigma_{t=0}^{L} \gamma^{t} \frac{\nabla_\theta \pi_\theta(s_t, a_t)}{\pi_\theta(s_t, a_t)} (\Sigma_{\tau=0}^{L-t} \gamma^{\tau} A^{\pi}(s_{\tau+t}, a_{\tau+t}))$$

Therefore, taking the expectation

$$\mathbb{E}[\theta_{L+1}] = \Sigma_{t=0}^{L} \gamma^{t} \mathbb{E}[\frac{\nabla_\theta \pi_\theta(s_t, a_t)}{\pi_\theta(s_t, a_t)} (\Sigma_{\tau=0}^{L-t} \gamma^{\tau} A^{\pi}(s_{\tau+t}, a_{\tau+t}))]$$
$$= \Sigma_{t=0}^{L} \gamma^{t} \mathbb{E}_{s_t, a_t}[\frac{\nabla_\theta \pi_\theta(s_t, a_t)}{\pi_\theta(s_t, a_t)} \times$$
$$\Sigma_{\tau=0}^{L-t} \gamma^{\tau} \mathbb{E}_{s_{t+1}, a_{t+1}, s_{t+2} \ldots}[A^{\pi}(s_{\tau+t}, a_{\tau+t}) | s_t, a_t]]$$

Using equations 1 and 2, we can say that

$$\mathbb{E}[\theta_{L+1}] = \Sigma_{t=0}^{L} \gamma^{t} \mathbb{E}[\frac{\nabla_\theta \pi_\theta(s_t, a_t)}{\pi_\theta(s_t, a_t)} A^{\pi}(s_t, a_t)]$$

Using the fact that $\mathbb{E}[\frac{\nabla_\theta \pi_\theta(s_t, a_t)}{\pi_\theta(s_t, a_t)} V^{\pi}(s_t)] = 0$ (Thomas & Brunskill, 2017)

$$\mathbb{E}[\theta_{L+1}] = \Sigma_{t=0}^{L} \gamma^{t} \mathbb{E}[\frac{\nabla_\theta \pi_\theta(s_t, a_t)}{\pi_\theta(s_t, a_t)} Q^{\pi}(s_t, a_t)]$$

Thus, using the argument in 3.1, we can state that E-COACH converges under advantage feedback.

□

6. Original COACH

This section assesses the convergence of the original COACH algorithm (MacGlashan et al., 2017a) under the three different types of feedback defined in this paper. Recall the main differences between E-COACH and COACH:

1. COACH makes use of an eligibility decay factor λ.
2. COACH does not discount the feedback by γ^t as part of the algorithm.

Note that λ and γ are not replaceable as the λ can only be used to discount stored gradients and thereby discount future rewards. On the other hand, γ is used to both discount future rewards as well as estimate the unnormalised state visitation distribution $d^{\pi}(s)$ described in 3.1.

As a result, the original COACH is incapable of estimating the state visitation distribution. Hence, the updates made at $t = 0$ and $t = 10$ would be weighted equally by COACH.

This property goes against what policy-gradient algorithms would do. By not using γ, COACH is basically drawing from a state visitation distribution different from the state visitation distribution $d^{\pi}(s)$ that is part of the objective function ρ that we described in 3.1. As a result, the updates made by COACH are not estimating ρ. Although COACH may learn to do reasonably well, we cannot say that it will behave optimally.

6.1. COACH Under One-Step Reward

The algorithm will converge, but the policy it converges to will be suboptimal for $\gamma \neq 1$ because COACH does not estimate the gradient of the policy gradient objective, $\nabla_\theta \rho$. Because COACH does not incorporate the discount factor, it behaves as if the domain has $\gamma = 1$, even if it isn't necessarily the case. If the domain has a $\gamma \in [0, 1)$, then the policy will have a long-term view because it will ignore this discount factor. Ignoring discounts can lead to suboptimal behavior.

Consider the five-state domain in Figure 2. It shows how the optimal decision in a state can change with the discount factor.

6.2. Policy Feedback

COACH will converge, but to a poor performing policy for the same reason given in Section 6.1.

6.3. Advantage Feedback

COACH should converge under this feedback type as per the argument given by MacGlashan et al..

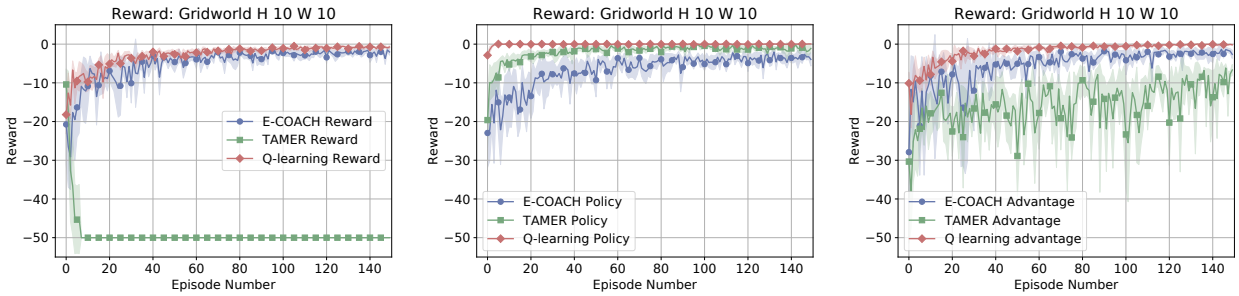
7. Comparison With Other Algorithms

We now know that E-COACH converges under several types of feedback. The three highlighted in this paper are Policy, Advantage, and Reward feedback. In this section, we compare E-COACH to TAMER (Knox & Stone, 2008) and Q-learning under these three types of feedback.

7.1. TAMER

TAMER expects the human trainer to take each action's long-term implications into account when providing feedback. TAMER learns the trainer's feedback function, then returns the policy that maximizes one-step feedback in each state.

The pseudocode as described in algorithm 3 is the TAMER algorithm. See (Knox & Stone, 2008) for more details.



(a) (b) (c)

Figure 1. Performance of E-COACH, TAMER and Q-learning under each feedback scheme. The domain is a 10 by 10 GridWorld coded using simple_rl (Abel, 2019). Each agent was run for 150 episodes and was cut short after 1000 steps. We ran 10 instances of each agent and plotted the reward averaged over these instances. E-COACH and Q-learning maximize rewards in all three settings, while TAMER falters under reward feedback and has difficulty with advantage. Although, other experimental results show TAMER doing well with advantage feedback; see MacGlashan et al. (2017b). These experimental results are meant to support our proofs of convergence for E-COACH. In addition, they support arguments made in Section 7.

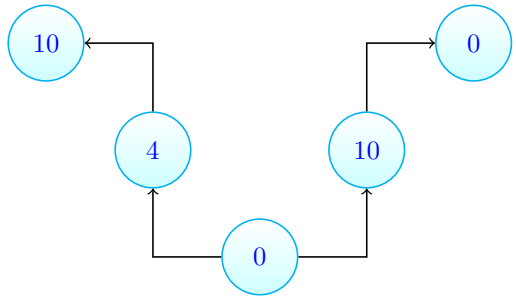


Figure 2. Example of the impact of discount factor on optimal policies. The number in the circle represents reward. The state in the middle is our starting state. For $\gamma \approx 1$, it is clear that the optimal policy is to go left to obtain a value of ≈ 14. For $\gamma \approx 0$, the optimal actions is to go right, instead, to obtain a value of ≈ 10 instead of ≈ 4. In general, the left action is preferred for $\gamma > 0.6$ and the right action is preferred for $\gamma < 0.6$. The choice of γ impacts the optimal policy.

The t represents the time, the weights $\overrightarrow{w}$ is used for the reward model, and the feature vectors $\overrightarrow{f_{t-2}}$ and $\overrightarrow{f_{t-1}}$ are state feature vectors. It takes input α, a learning rate.

Note that there are several different versions of TAMER. The one we are analyzing is the original by Knox & Stone (2008).

7.1.1. Reward Feedback

Because TAMER will maximize over the learned function, it will result in a bad policy for this form of feedback. TAMER does not take future rewards into account and instead will greedily maximize for immediate reward. TAMER assumes the trainer has taken future rewards into account already. See figure 1(a).

Algorithm 3 TAMER $\langle\alpha\rangle$

$t \leftarrow 0$
$\overrightarrow{w} \leftarrow \overrightarrow{0}$
$\overrightarrow{f_{t-2}} \leftarrow \overrightarrow{0}$
$\overrightarrow{f_{t-1}} \leftarrow \overrightarrow{0}$
$a \leftarrow ChooseAction(s_t, \overrightarrow{w})$
takeAction(a)
while true **do**
  $t \leftarrow t + 1$
  if $t \geq 2$ **then**
    $r_{t-2} \leftarrow getHumanFeedback()$
    if $r_{t-2} \neq 0$ **then**
      $\overrightarrow{w} \leftarrow UpdateRewModel(r_{t-2}, \overrightarrow{f_{t-2}}, \overrightarrow{f_{t-2}}, \overrightarrow{w}, \alpha)$
    end if
  end if
  $a \leftarrow ChooseAction(s_t, \overrightarrow{w})$
  $takeAction(a)$
  $s_t \leftarrow getState()$
  $f_{t-2} \leftarrow f_{t-1}$
  $f_{t-1} \leftarrow getFeatureVec(s_t)$
end while

7.1.2. Policy Feedback

TAMER expects policy feedback and chooses correct actions assuming sufficient exploration. See figure 1(b).

7.1.3. Advantage Feedback

It is not known precisely how TAMER responds to advantage feedback. Knox & Stone (2008) claim that TAMER should work under moving feedback. That is, TAMER should behave properly even when feedback changes over time because the algorithm expects the human trainer to be

Algorithm 4 UpdateRewModel $\langle r_{t-2}, \overrightarrow{f_{t-2}}, \overrightarrow{f_{t-1}}, \overrightarrow{w}, \alpha \rangle$

Set α as a parameter.
$\overrightarrow{\Delta f_{t-1,t-2}} \leftarrow \overrightarrow{f_{t-1}} - \overrightarrow{f_{t-2}}$
$projectedRew_{t-2} \leftarrow \sum_i (w_i \times \overrightarrow{\Delta f_{t-1,t-2}})$
$error \leftarrow r_{t-2} - projectedRew_{t-2}$
for i in $range(0, length(\overrightarrow{w}))$ **do**
  $w_i \leftarrow w_i + \alpha \times error \times \overrightarrow{\Delta f_{t-1,t-2}}$
end for
return $\overrightarrow{w}$

Algorithm 5 ChooseAction $\langle s_t, \overrightarrow{w} \rangle$

$\overrightarrow{f_t} \leftarrow getFeatureVec(s_t)$
for each $a \in getAction(s_t)$ **do**
  $s_{t+1,a} \leftarrow T(s_t, a)$
  $\overrightarrow{f_{t+1,a}} \leftarrow getFeatureVec(s_{t+1}, a)$
  $\overrightarrow{\Delta f_{t+1,t}} \leftarrow \overrightarrow{f_{t+1,a}} - \overrightarrow{f_t}$
  $projectedRew_a \leftarrow \sum_i (w_i \times \delta f_{t+1,t,i}$
end for
return $argmax_a(projectedRew_a)$

inconsistent and continues to update its choices even in the face of changes. The advantage function assigns different values to actions as the policy is updated, so at different times it gets different values. Assuming TAMER is able to learn this moving function, then a greedy one-step policy should be optimal because the maximal value of the advantage function is always the optimal action for the given state. See figure 1(c).

7.2. Q-learning

Q-learning (Watkins, 1989) is an algorithm that expects feedback in the form of immediate reward and calculates long-term value from these signals. Specifically, $Q_k(s, a)$ is its estimate of long-term value and, when it is informed of a transition from s_t to s_{t+1} via action a_t and feedback f_t, it makes the update:

$$Q_{k+1}(s_t, a_t) \leftarrow (1-\alpha) Q_k(s_t, a_t) + \alpha(f_t + \gamma V(s_{t+1})).$$

7.2.1. Reward Feedback

Q-learning is typically defined to expect the feedback to be the expected one-step reward $f_t = R(s_t, a_t)$ or a value whose expectation is $R(s_t, a_t)$. It has been proven to converge to optimal behavior under this type of feedback (Watkins & Dayan, 1992; Littman & Szepesvári, 1996; Singh et al., 2000; Melo, 2001). See figure 1(a).

7.2.2. Policy Feedback

Given policy feedback, Q-learning will optimize the expected sum of future "rewards", which, in this case, is an indicator of whether the agent's selected action is the trainer's target policy or not.

Policy feedback depends on only the previous state and action, and, as such, Q-learning can treat this feedback as a reward function and converge on the behavior that optimizes the sum of these feedbacks.

Interestingly, the policy that optimizes the sum of policy feedbacks is exactly the target policy. This observation follows from the fact that matching the trainer's target policy results in a value of $1 + \gamma^1 + \gamma^2 + \gamma^3 + \cdots$. On the other hand, selecting even a single action that does not match the trainer's target policy results in the removal of one of these terms and therefore lower value. Under policy feedback, Q-learning thus converges to the policy that matches the trainer's target policy. See figure 1(b).

7.2.3. Advantage Feedback

Q-learning is not designed to work with advantage feedback because the advantage function is policy dependent and can cause its reward signals to change as it updates its value. Nevertheless, advantage feedback does provide a signal for how values should *change* and, empirically, we often see Q-learning handling advantage feedback well. The analytical challenge is that the changes in the policy influence the reward and the changes in the reward influence the policy, so these two functions need to converge *together* for Q-learning to handle advantage feedback successfully.

We conjecture that careful annealing of Q-learning's learning rate could provide a mechanism for stabilizing these two different adaptive processes. Resolving this question is a topic for future work. We believe the work done by Konda & Borkar (1999) could provide greater insight. See figure 1(c), where Q-learning appears to converge for a simple GridWorld domain.

8. Conclusion

In this paper, we analyzed the convergence of COnvergent Actor-Critic by Humans (MacGlashan et al., 2017a) under three types of feedback—one-step reward, policy, and advantage feedback. These are all examples of feedback a human trainer might give.

We defined a COACH variant called E-COACH and demonstrated its convergence under these types of feedback. Original COACH, unfortunately, does not necessarily converge to an optimal policy under the feedback types defined in this paper. In addition, we compared the new E-COACH with two algorithms: Q-learning and TAMER. TAMER does

poorly under one-step-reward feedback. And Q-learning appears to converge to optimal behavior under one-step-reward and policy feedback, but future work is required to determine its performance under advantage feedback.

References

Abel, D. simple_rl: Reproducible reinforcement learning in python. In *ICLR Workshop on Reproducibility in Machine Learning*, 2019.

Agarwal, A., Kakade, S. M., Lee, J. D., and Mahajan, G. On the theory of policy gradient methods: Optimality, approximation, and distribution shift. 2020. arXiv:1908.00251v5.

Ho, M. K., Cushman, F., Littman, M. L., and Austerweil, J. L. People teach with rewards and punishments as communication, not reinforcements. *Journal of Experimental Psychology: General*, pp. 520–549, 2019.

Jin, C., Ge, R., Netrapalli, P., Kakade, S. M., and Jordan, M. I. How to escape saddle points efficiently. 2017. arXiv:1703.00887v1.

Kearns, M. and Singh, S. Near-optimal reinforcement learning in polynomial time. pp. 12–16, 1998.

Knox, W. B. and Stone, P. TAMER: Training an agent manually via evaluative reinforcement. In *2008 7th IEEE International Conference on Development and Learning*, pp. 292–297. IEEE, 2008.

Konda, V. R. and Borkar, V. S. Actor-critic–type learning algorithms for markov decision processes. *SIAM J. CONTROL OPTIM*, 1999.

Littman, M. L. and Szepesvári, C. A generalized reinforcement-learning model: Convergence and applications. In Saitta, L. (ed.), *Proceedings of the Thirteenth International Conference on Machine Learning*, pp. 310–318, 1996.

MacGlashan, J., Ho, M. K., Loftin, R., Peng, B., Wang, G., Roberts, D. L., Taylor, M. E., and Littman, M. L. Interactive learning from policy-dependent human feedback. In *Proceedings of the Thirty-Fourth International Conference on Machine Learning*, 2017a.

MacGlashan, J., Ho, M. K., Loftin, R., Peng, B., Wang, G., Roberts, D. L., Taylor, M. E., and Littman, M. L. Interactive learning from policy-dependent human feedback. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 2285–2294. JMLR. org, 2017b.

Melo, F. S. Convergence of Q-learning: A simple proof. *Institute Of Systems and Robotics, Tech. Rep*, pp. 1–4, 2001.

Singh, S., Jaakkola, T., Littman, M. L., and Szepesvári, C. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine Learning*, 39:287–308, 2000.

Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In *Neural Information Processing Systems*, pp. 1057–1063, 2000.

Thomas, P. S. and Brunskill, E. Policy gradient methods for reinforcement learning with function approximation and action-dependent baselines. arXiv:1706.06643v1, 2017.

Watkins, C. J. C. H. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, UK, 1989.

Watkins, C. J. C. H. and Dayan, P. Q-learning. *Machine Learning*, 8(3):279–292, 1992.

Zhang, K., Koppel, A., Zhu, H., and Basar, T. Global convergence of policy gradient methods to (almost) locally optimal policies. 2020. arXiv:1906.08383v3.