

SAC, Eligibility Traces

Milan Straka

 November 22, 2021



EUROPEAN UNION
European Structural and Investment Fund
Operational Programme Research,
Development and Education

Charles University in Prague
Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics



unless otherwise stated

The paper *Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor* by Tuomas Haarnoja et al. from Jan 2018 introduces a different off-policy algorithm for continuous action space.

It was followed by a continuation paper *Soft Actor-Critic Algorithms and Applications* in Dec 2018.

The general idea is to introduce entropy directly in the value function we want to maximize, instead of just ad-hoc adding the entropy penalty.

Until now, our goal was to optimize

$$\mathbb{E}_{\pi} [G_0].$$

Assume the rewards are deterministic and that μ_{π} is on-policy distribution of a policy π .

In the soft actor-critic, the authors instead propose to optimize the maximum entropy objective

$$\pi_{*} = \arg \max_{\pi} \mathbb{E}_{s,a \sim \mu_{\pi}} [r(s, a) + \alpha H(\pi(\cdot|s))].$$

Note that the value of α is dependent on the magnitude of returns and that for a fixed policy, the entropy penalty can be “hidden” in the reward.

Soft Actor Critic Objective

The authors in fact utilize slightly different objective, notably a one with the following augmented reward

$$r_{\pi}(s, a) \stackrel{\text{def}}{=} r(s, a) + \mathbb{E}_{s' \sim p(s, a)} [\alpha H(\pi(\cdot | s'))].$$

From now on, we consider **soft action-value** function corresponding to this augmented reward.

Our goal is now to derive **soft policy iteration**, an analogue of policy iteration algorithm.

We start by considering soft policy evaluation. Let a modified Bellman backup operator \mathcal{T}_π be defined as

$$\mathcal{T}_\pi q(s, a) \stackrel{\text{def}}{=} r(s, a) + \gamma \mathbb{E}_{s' \sim p(s, a)} [v(s')],$$

where the **soft (state-)value** function $v(s)$ is defined as

$$v(s) = \mathbb{E}_{a \sim \pi} [q(s, a) - \alpha \log \pi(a|s)].$$

This modified Bellman backup operator corresponds to the usual one for the augmented rewards $r_\pi(s, a)$, and therefore the repeated application $\mathcal{T}_\pi^k q$ converges to q_π according to the original proof.

Soft Policy Improvement

While the soft policy evaluation was a straightforward modification of the original policy evaluation, the soft policy improvement is quite different.

Assume we have a policy π , its action-value function q_π from the soft policy evaluation, and we want to improve the policy. Furthermore, we should select the improved policy from a family of parametrized distributions Π .

We define the improved policy π' as

$$\pi'(\cdot|s) \stackrel{\text{def}}{=} \arg \min_{\bar{\pi} \in \Pi} J_\pi(\bar{\pi}) \stackrel{\text{def}}{=} \arg \min_{\bar{\pi} \in \Pi} D_{\text{KL}} \left(\bar{\pi}(\cdot|s) \left\| \frac{\exp\left(\frac{1}{\alpha} q_\pi(s, \cdot)\right)}{z_\pi(s)} \right. \right),$$

where $z_\pi(s)$ is the partition function (i.e., normalization factor such that the right-hand side is a distribution), which does not depend on the new policy and thus can be ignored.

Soft Policy Improvement

We now prove that $q_{\pi'}(s, a) \geq q_{\pi}(s, a)$ for any state s and action a .

We start by noting that $J_{\pi}(\pi') \leq J_{\pi}(\pi)$, because we can always choose π as the improved policy. Therefore,

$$\mathbb{E}_{a \sim \pi'} [\alpha \log \pi'(a|s) - q_{\pi}(s, a) + \alpha \log z_{\pi}(s)] \leq \mathbb{E}_{a \sim \pi} [\alpha \log \pi(a|s) - q_{\pi}(s, a) + \alpha \log z_{\pi}(s)],$$

which results in

$$\mathbb{E}_{a \sim \pi'} [q_{\pi}(s, a) - \alpha \log \pi'(a|s)] \geq v_{\pi}(s).$$

We now finish the proof analogously to the original one:

$$\begin{aligned} q_{\pi}(s, a) &= r(s, a) + \gamma \mathbb{E}_{s'} [v_{\pi}(s')] \\ &\leq r(s, a) + \gamma \mathbb{E}_{s'} [\mathbb{E}_{a' \sim \pi'} [q_{\pi}(s', a') - \alpha \log \pi'(a'|s')]] \\ &\dots \\ &\leq q_{\pi'}(s, a). \end{aligned}$$

The soft policy iteration algorithm alternates between the soft policy evaluation and soft policy improvement steps.

The repeated application of these two steps produce better and better policies. In other words, we get a monotonically increasing sequence of soft action-value functions.

If the soft action-value function is bounded (the paper assumes a bounded reward and a finite number of actions to bound the entropy), the repeated application converges to some q_* , from which we get a π_* using the soft policy improvement step.

It remains to show that the π_* is indeed the optimal policy fulfilling $q_{\pi_*}(s, a) \geq q_{\pi}(s, a)$.

However, this follows from the fact that at convergence, $J_{\pi_*}(\pi_*) \leq J_{\pi_*}(\pi)$, and following the same reasoning as in the proof of the soft policy improvement, we obtain the required $q_{\pi_*}(s, a) \geq q_{\pi}(s, a)$.

Our soft actor critic will be an off-policy algorithm with continuous action space. The model consist of two critics q_{θ_1} and q_{θ_2} , two target critics $q_{\bar{\theta}_1}$ and $q_{\bar{\theta}_2}$ and finally a single actor π_{φ} .

The authors state that

- with a single critic, all the described experiments still converge;
- they adopted the two critics from the TD3 paper;
- using two critics “significantly speed up training”.

To train the critic, we use the modified Bellman backup operator, resulting in the loss

$$J_q(\boldsymbol{\theta}_i) = \mathbb{E}_{s,a \sim \mu} \left[\left(q_{\boldsymbol{\theta}_i}(s, a) - \left(r(s, a) + \gamma \mathbb{E}_{s' \sim p(s,a)} [v_{\min}(s')] \right) \right)^2 \right],$$

where

$$v_{\min}(s) = \mathbb{E}_{a \sim \pi_{\varphi}} \left[\min_i (q_{\bar{\boldsymbol{\theta}}_i}(s, a)) - \alpha \log \pi_{\varphi}(a|s) \right].$$

The target critics are updated using exponential moving averages with momentum τ .

The actor is updated by directly minimizing the KL divergence, resulting in the loss

$$J_{\pi}(\boldsymbol{\varphi}) = \mathbb{E}_{s \sim \mu} \mathbb{E}_{a \sim \pi_{\boldsymbol{\varphi}}} \left[\alpha \log (\pi_{\boldsymbol{\varphi}}(a, s)) - \min_i (q_{\boldsymbol{\theta}_i}(s, a)) \right].$$

Given that our critics are differentiable, we now reparametrize the policy as

$$a = f_{\boldsymbol{\varphi}}(s, \varepsilon).$$

Specifically, we sample $\varepsilon \sim \mathcal{N}(0, 1)$ and let $f_{\boldsymbol{\varphi}}$ produce an unbounded Gaussian distribution (a diagonal one if the actions are vectors).

Together, we obtain

$$J_{\pi}(\boldsymbol{\varphi}) = \mathbb{E}_{s \sim \mu} \mathbb{E}_{\varepsilon \sim \mathcal{N}(0,1)} \left[\alpha \log (\pi_{\boldsymbol{\varphi}}(f_{\boldsymbol{\varphi}}(s, \varepsilon), s)) - \min_i (q_{\boldsymbol{\theta}_i}(s, f_{\boldsymbol{\varphi}}(s, \varepsilon))) \right].$$

In practice, the actions need to be bounded.

The authors propose to apply an invertible squashing function \tanh on the unbounded Gaussian distribution.

Consider that our policy produces an unbounded action $\pi(u|s)$. To define a distribution $\bar{\pi}(a|s)$ with $a = \tanh(u)$, we need to employ the change of variables, resulting in

$$\bar{\pi}(a|s) = \pi(u|s) \left(\frac{\partial a}{\partial u} \right)^{-1} = \pi(u|s) \left(\frac{\partial \tanh(u)}{\partial u} \right)^{-1}.$$

Therefore, the log-likelihood has quite a simple form

$$\log \bar{\pi}(a|s) = \log \pi(u|s) - \log (1 - \tanh^2(u)).$$

One of the most important hyperparameters is the entropy penalty α .

In the second paper, the authors presented an algorithm for automatic adjustment of its value.

Instead of setting the entropy penalty α , they propose to specify target entropy value \mathcal{H} and then solve a constrained optimization problem

$$\pi_* = \arg \max_{\pi} \mathbb{E}_{s,a \sim \mu_{\pi}} [r(s, a)] \quad \text{such that} \quad \mathbb{E}_{s,a \sim \mu_{\pi}} [-\log \pi(a|s)] \geq \mathcal{H}.$$

We can then form a Lagrangian with a multiplier α

$$\mathbb{E}_{s,a \sim \mu_{\pi}} \left[r(s, a) + \alpha (-\log \pi(a|s) - \mathcal{H}) \right],$$

which should be maximized with respect to π and minimized with respect to $\alpha \geq 0$.

To optimize the Lagrangian, we perform the *dual gradient descent*, where we alternate between maximization with respect to π and minimization with respect to α .

While such a procedure is guaranteed to converge only under the convexity assumptions, the authors report that the dual gradient descent works in practice also with non-linear function approximation.

To conclude, the automatic entropy adjustment is performed by introducing a final loss

$$J(\alpha) = \mathbb{E}_{a \sim \pi} \left[-\alpha \log \pi(a|s) - \alpha \mathcal{H} \right].$$

Algorithm 1 Soft Actor-Critic

Input: θ_1, θ_2, ϕ

$\bar{\theta}_1 \leftarrow \theta_1, \bar{\theta}_2 \leftarrow \theta_2$

$\mathcal{D} \leftarrow \emptyset$

for each iteration **do**

for each environment step **do**

$\mathbf{a}_t \sim \pi_\phi(\mathbf{a}_t | \mathbf{s}_t)$

$\mathbf{s}_{t+1} \sim p(\mathbf{s}_{t+1} | \mathbf{s}_t, \mathbf{a}_t)$

$\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{s}_t, \mathbf{a}_t, r(\mathbf{s}_t, \mathbf{a}_t), \mathbf{s}_{t+1})\}$

end for

for each gradient step **do**

$\theta_i \leftarrow \theta_i - \lambda_Q \hat{\nabla}_{\theta_i} J_Q(\theta_i)$ for $i \in \{1, 2\}$

$\phi \leftarrow \phi - \lambda_\pi \hat{\nabla}_\phi J_\pi(\phi)$

$\alpha \leftarrow \alpha - \lambda \hat{\nabla}_\alpha J(\alpha)$

$\bar{\theta}_i \leftarrow \tau \theta_i + (1 - \tau) \bar{\theta}_i$ for $i \in \{1, 2\}$

end for

end for

Output: θ_1, θ_2, ϕ

▷ Initial parameters

▷ Initialize target network weights

▷ Initialize an empty replay pool

▷ Sample action from the policy

▷ Sample transition from the environment

▷ Store the transition in the replay pool

▷ Update the Q-function parameters

▷ Update policy weights

▷ Adjust temperature

▷ Update target network weights

▷ Optimized parameters

Algorithm 1 of the paper "Soft Actor-Critic Algorithms and Applications" by Tuomas Haarnoja et al.

Table 1: SAC Hyperparameters

Parameter	Value
optimizer	Adam (Kingma & Ba, 2015)
learning rate	$3 \cdot 10^{-4}$
discount (γ)	0.99
replay buffer size	10^6
number of hidden layers (all networks)	2
number of hidden units per layer	256
number of samples per minibatch	256
entropy target	$-\dim(\mathcal{A})$ (e.g. , -6 for HalfCheetah-v1)
nonlinearity	ReLU
target smoothing coefficient (τ)	0.005
target update interval	1
gradient steps	1

Table 1 of the paper "Soft Actor-Critic Algorithms and Applications" by Tuomas Haarnoja et al.

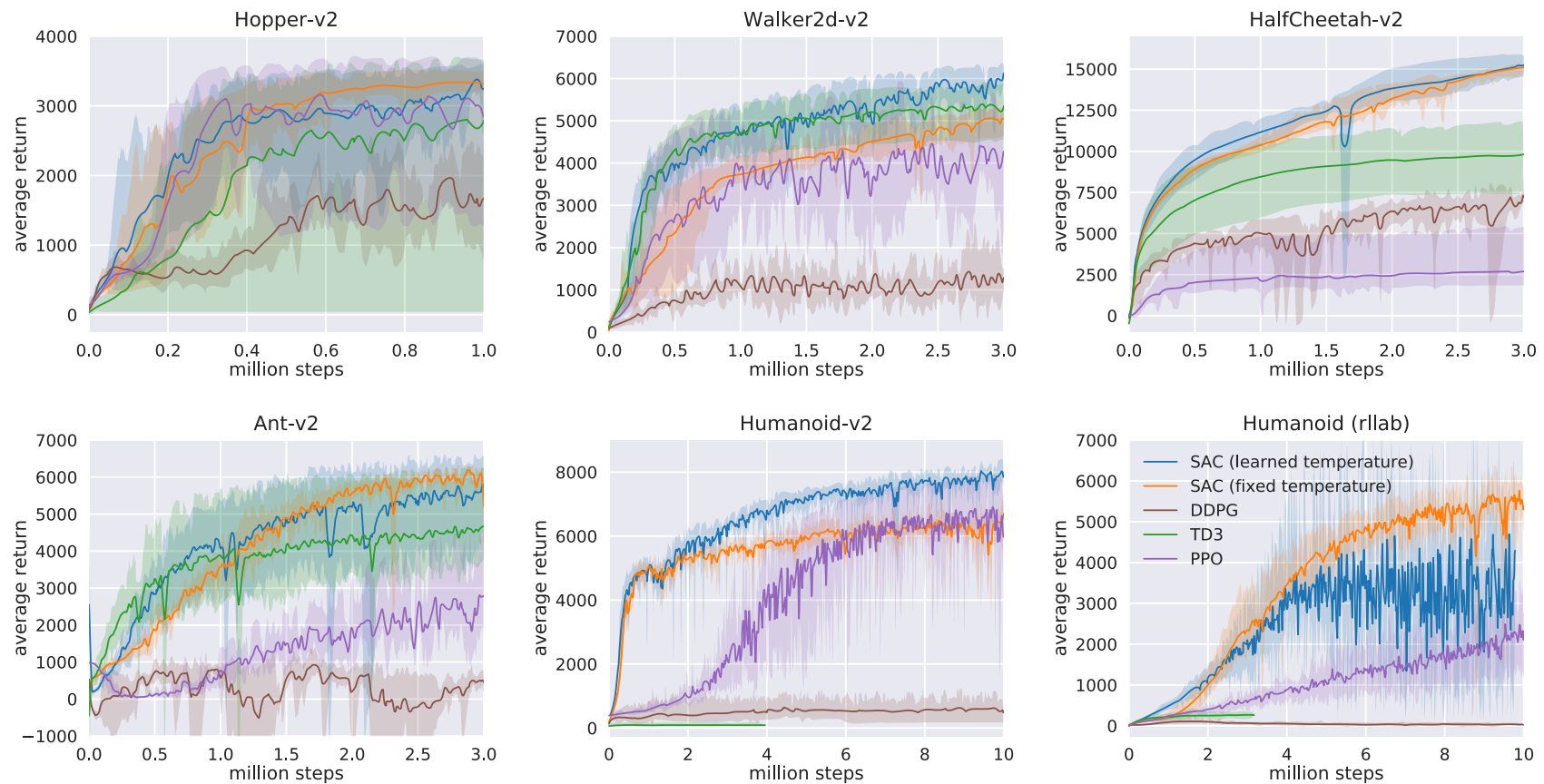


Figure 1: Training curves on continuous control benchmarks. Soft actor-critic (blue and yellow) performs consistently across all tasks and outperforming both on-policy and off-policy methods in the most challenging tasks.

Figure 1 of the paper "Soft Actor-Critic Algorithms and Applications" by Tuomas Haarnoja et al.

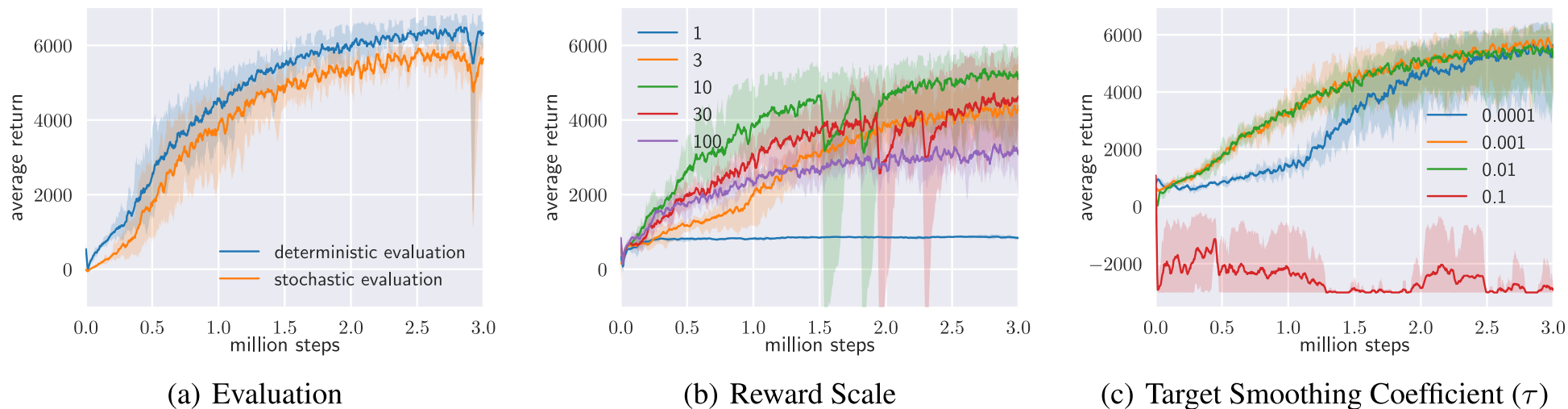


Figure 3. Sensitivity of soft actor-critic to selected hyperparameters on Ant-v1 task. (a) Evaluating the policy using the mean action generally results in a higher return. Note that the policy is trained to maximize also the entropy, and the mean action does not, in general, correspond the optimal action for the maximum return objective. (b) Soft actor-critic is sensitive to reward scaling since it is related to the temperature of the optimal policy. The optimal reward scale varies between environments, and should be tuned for each task separately. (c) Target value smoothing coefficient τ is used to stabilize training. Fast moving target (large τ) can result in instabilities (red), whereas slow moving target (small τ) makes training slower (blue).

Figure 3 of the paper "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor" by Tuomas Haarnoja et al.

Off-policy Correction Using Control Variates

Let $G_{t:t+n}$ be the estimated n -step return

$$G_{t:t+n} \stackrel{\text{def}}{=} \left(\sum_{k=t}^{t+n-1} \gamma^{k-t} R_{k+1} \right) + \left[\text{episode still running in } t+n \right] \gamma^n V(S_{t+n}),$$

which can be written recursively as

$$G_{t:t+n} \begin{cases} 0 & \text{if episode ended before } t, \\ V(S_t) & \text{if } n = 0, \\ R_{t+1} + \gamma G_{t+1:t+n} & \text{otherwise.} \end{cases}$$

Note that we can write

$$\begin{aligned} G_{t:t+n} - V(S_t) &= R_{t+1} + \gamma G_{t+1:t+n} - V(S_t) \\ &= R_{t+1} + \gamma(G_{t+1:t+n} - V(S_{t+1})) + \gamma V(S_{t+1}) - V(S_t), \end{aligned}$$

which yields

$$G_{t:t+n} - V(S_t) = R_{t+1} + \gamma V(S_{t+1}) - V(S_t) + \gamma(G_{t+1:t+n} - V(S_{t+1})).$$

Denoting the TD error as $\delta_t \stackrel{\text{def}}{=} R_{t+1} + \gamma V(S_{t+1}) - V(S_t)$, we can therefore write the n -step estimated return as a sum of TD errors:

$$G_{t:t+n} = V(S_t) + \sum_{i=0}^{n-1} \gamma^i \delta_{t+i}.$$

Off-policy Correction Using Control Variates

Now consider applying the IS off-policy correction to $G_{t:t+n}$ using the importance sampling ratio

$$\rho_t \stackrel{\text{def}}{=} \frac{\pi(A_t|S_t)}{b(A_t|S_t)}, \quad \rho_{t:t+n} \stackrel{\text{def}}{=} \prod_{i=0}^n \rho_{t+i}.$$

First note that

$$\mathbb{E}_{A_t \sim b} [\rho_t] = \sum_{A_t} b(A_t|S_t) \frac{\pi(A_t|S_t)}{b(A_t|S_t)} = 1,$$

which can be extended to

$$\mathbb{E}_b [\rho_{t:t+n}] = 1.$$

Until now, we used

$$G_{t:t+n}^{\text{IS}} \stackrel{\text{def}}{=} \rho_{t:t+n-1} G_{t:t+n}.$$

However, such correction has unnecessary variance. Notably, when expanding $G_{t:t+n}$

$$G_{t:t+n}^{\text{IS}} = \rho_{t:t+n-1} (R_{t+1} + \gamma G_{t+1:t+n}),$$

the R_{t+1} depends only on ρ_t , not on $\rho_{t+1:t+n-1}$, and given that the expectation of the importance sampling ratio is 1, we can simplify to

$$G_{t:t+n}^{\text{IS}} = \rho_t R_{t+1} + \rho_{t:t+n-1} \gamma G_{t+1:t+n}.$$

Such an estimate can be written recursively as

$$G_{t:t+n}^{\text{IS}} = \rho_t (R_{t+1} + \gamma G_{t+1:t+n}^{\text{IS}}).$$

Off-policy Correction Using Control Variates

We can reduce the variance even further – when $\rho_t = 0$, we might consider returning the value of $V(S_t)$ instead of 0.

Therefore, we might add another term, the so-called **control variate**, to the estimate

$$G_{t:t+n}^{\text{CV}} \stackrel{\text{def}}{=} \rho_t (R_{t+1} + \gamma G_{t+1:t+n}^{\text{CV}}) + (1 - \rho_t)V(S_t),$$

which is valid, since the expected value of $1 - \rho_t$ is zero and ρ_t and S_t are independent.

Similarly as before, rewriting to

$$\begin{aligned} G_{t:t+n}^{\text{CV}} - V(S_t) &= \rho_t (R_{t+1} + \gamma G_{t+1:t+n}^{\text{CV}}) - \rho_t V(S_t) \\ &= \rho_t (R_{t+1} + \gamma V(S_{t+1}) - V(S_t) + \gamma (G_{t+1:t+n}^{\text{CV}} - V(S_{t+1}))) \end{aligned}$$

results in

$$G_{t:t+n}^{\text{CV}} = V(S_t) + \sum_{i=0}^{n-1} \gamma^i \rho_{t:t+i} \delta_{t+i}.$$

Eligibility Traces

Eligibility traces are a mechanism of combining multiple n -step return estimates for various values of n .

First note instead of an n -step return, we can use any average of n -step returns for different values of n , for example $\frac{2}{3}G_{t:t+2} + \frac{1}{3}G_{t:t+4}$.

For a given $\lambda \in [0, 1]$, we define λ -return as

$$G_t^\lambda \stackrel{\text{def}}{=} (1 - \lambda) \sum_{i=1}^{\infty} \lambda^{i-1} G_{t:t+i}.$$

Alternatively, the λ -return can be written recursively as

$$G_t^\lambda = (1 - \lambda)G_{t:t+1} + \lambda(R_{t+1} + \gamma G_{t+1}^\lambda).$$

Weighting

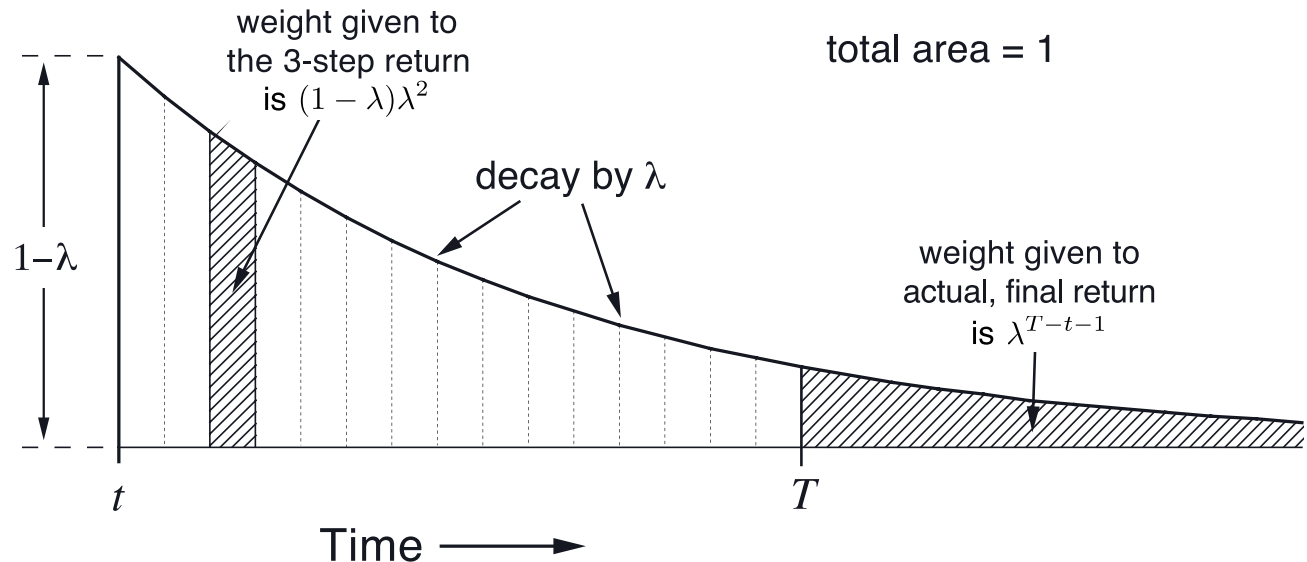


Figure 12.2: Weighting given in the λ -return to each of the n -step returns.

Figure 12.2 of "Reinforcement Learning: An Introduction, Second Edition".

In an episodic task with time of termination T , we can rewrite the λ -return to

$$G_t^\lambda = (1 - \lambda) \sum_{i=1}^{T-t-1} \lambda^{i-1} G_{t:t+i} + \lambda^{T-t-1} G_t.$$

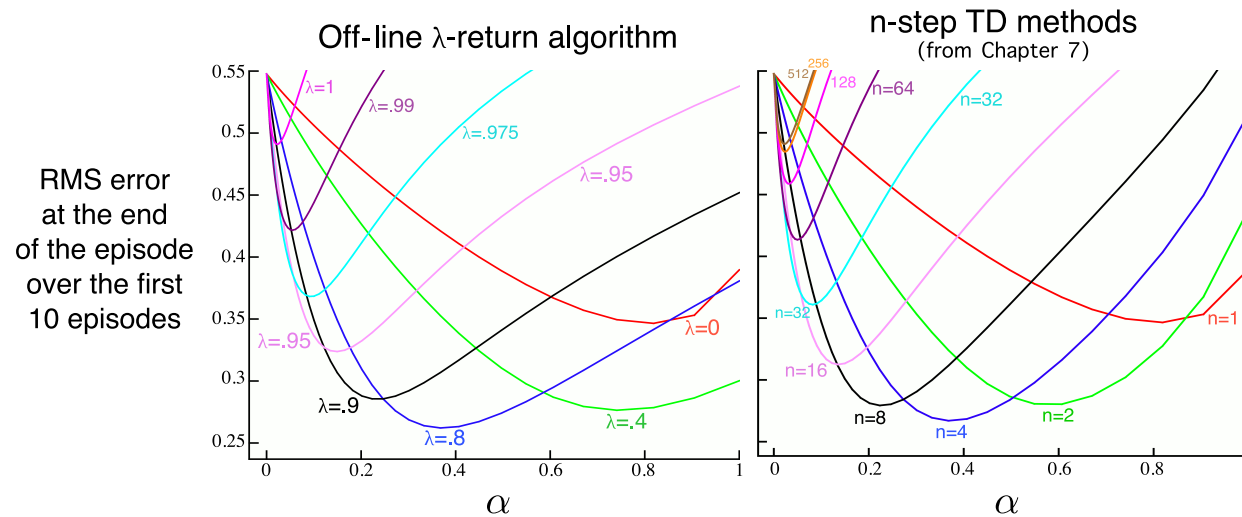


Figure 12.3: 19-state Random walk results (Example 7.1): Performance of the off-line λ -return algorithm alongside that of the n -step TD methods. In both case, intermediate values of the bootstrapping parameter (λ or n) performed best. The results with the off-line λ -return algorithm are slightly better at the best values of α and λ , and at high α .

Figure 12.3 of "Reinforcement Learning: An Introduction, Second Edition".