# SAC, Eligibility Traces

Milan Straka

November 21, 2022

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

# Soft Actor Critic

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. Such an approach is an instance of *regularized policy optimization*.

# Soft Actor Critic Objective

Until now, our goal was to optimize

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

Assume the rewards are deterministic and that $\mu_\pi$ is on-policy distribution of a policy $\pi$.

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

$$\begin{aligned}\pi_* &= \arg\max_\pi \mathbb{E}_{s\sim\mu_\pi} \Big[\mathbb{E}_{a\sim\pi(s)} \big[r(s,a)\big] + \alpha H(\pi(\cdot|s))\Big] \\ &= \arg\max_\pi \mathbb{E}_{s\sim\mu_\pi, a\sim\pi(s)} \big[r(s,a) - \alpha \log \pi(a|s)\big].\end{aligned}$$

Note that the value of $\alpha$ 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

To maximize the regularized objective, we define the following augmented reward:

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

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

# Soft Policy Evaluation

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 $\mathcal{T}_\pi$ be defined as

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

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

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

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 $q_\pi$ 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 $\pi$, its action-value function $q_\pi$ 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 $\Pi$.

We define the improved policy $\pi'$ as

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

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 $\pi$ as the improved policy. Therefore,

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

which results in

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

We now finish the proof analogously to the original one:

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

# Soft Policy Iteration

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 $\pi_*$ using the soft policy improvement step. (It is not clear to me why the algorithm should converge in finite time, but we can make the rest of the slide conditional on "if the algorithm converges").

It remains to show that the $\pi_*$ 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 $q_{\pi_*}(s, a) \geq q_\pi(s, a)$.

# Soft Policy Improvement Derivation

The following derivation is not in the original paper, but it is my understanding of how the softmax of the action-value function arises. For simplicity, we assume finite number of actions.

Assuming we have a policy $\pi$ and its action-value function $q_\pi$, we usually improve the policy using

$$\begin{aligned} \nu(\cdot|s) &= \arg\max_\nu \mathbb{E}_{a\sim\nu(\cdot|s)}\big[q_\pi(s,a)\big] \\ &= \arg\max_\nu \sum\nolimits_a q_\pi(s,a)\nu(a|s) \\ &= \arg\max_\nu \boldsymbol{q}_\pi(s,\cdot)^T\boldsymbol{\nu}(\cdot|s), \end{aligned}$$

which results in a greedy improvement with the form of

$$\nu(s) = \arg\max\nolimits_a q_\pi(s,a).$$

# Soft Policy Improvement Derivation

Now consider instead the regularized objective

$$\begin{aligned}\nu(\cdot|s) &= \arg\max_\nu \left(\mathbb{E}_{a\sim\nu(\cdot|s)}\big[q_\pi(s,a)\big] + \alpha H(\nu(\cdot|s))\right)\\ &= \arg\max_\nu \left(\mathbb{E}_{a\sim\nu}\big[q_\pi(s,a) - \alpha\log\nu(a|s)\big]\right)\end{aligned}$$

To maximize it for a given $s$, we form a Lagrangian

$$\mathcal{L} = \Big(\sum\nolimits_a \nu(a|s)\big(q_\pi(s,a) - \alpha\log\nu(a|s)\big)\Big) - \lambda\Big(1 - \sum\nolimits_a \nu(a|s)\Big).$$

The derivative with respect to $\nu(a|s)$ is

$$\frac{\partial\mathcal{L}}{\partial\nu(a|s)} = q_\pi(s,a) - \alpha\log\nu(a|s) - \alpha + \lambda.$$

Setting it to zero, we get $\alpha\log\nu(a|s) = q_\pi(s,a) + \lambda - \alpha$, resulting in $\nu(a|s) \propto e^{\frac{1}{\alpha}q_\pi(s,a)}$.

# Soft Actor Critic

Our soft actor critic will be an off-policy algorithm with continuous action space. The model consist of two critics $q_{\boldsymbol\theta_1}$ and $q_{\boldsymbol\theta_2}$, two target critics $q_{\bar{\boldsymbol\theta}_1}$ and $q_{\bar{\boldsymbol\theta}_2}$ and finally a single actor $\pi_{\boldsymbol\varphi}$.

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".

# Soft Actor Critic – Critic Training

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

$$J_q(\boldsymbol\theta_i) = \mathbb{E}_{s\sim\mu_\pi, a\sim\pi_\varphi(s)}\Big[\Big(q_{\boldsymbol\theta_i}(s,a) - \big(r(s,a) + \gamma\mathbb{E}_{s'\sim p(s,a)}[v_{\min}(s')]\big)\Big)^2\Big],$$

where

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

The target critics are updated using exponentiation moving averages with momentum $\tau$.

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

$$J_\pi(\boldsymbol\varphi) = \mathbb{E}_{s\sim\mu_\pi, a\sim\pi_{\boldsymbol\varphi}(s)}\Big[\alpha \log\big(\pi_{\boldsymbol\varphi}(a, s)\big) - \min_i \big(q_{\boldsymbol\theta_i}(s, a)\big)\Big].$$

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

$$J_\pi(\boldsymbol\varphi) = \mathbb{E}_{s\sim\mu_\pi, \varepsilon\sim\mathcal{N}(0,1)}\Big[\alpha \log\big(\pi_{\boldsymbol\varphi}(f_{\boldsymbol\varphi}(s, \varepsilon), s)\big) - \min_i \big(q_{\boldsymbol\theta_i}(s, f_{\boldsymbol\varphi}(s, \varepsilon))\big)\Big].$$

# Soft Actor Critic – Bounding Actions

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

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

Therefore, the log-likelihood has quite a simple form

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

One of the most important hyperparameters is the entropy penalty $\alpha$.

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

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

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

We can then form a Lagrangian with a multiplier $\alpha$

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

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

# Soft Actor Critic – Automatic Entropy Adjustment

To optimize the Lagrangian, we perform *dual gradient descent*, where we alternate between maximization with respect to $\pi$ and minimization with respect to $\alpha$.

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 nonlinear function approximation.

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

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

**Algorithm 1** Soft Actor-Critic

**Input:** $\theta_1, \theta_2, \phi$ ▷ Initial parameters
$\bar{\theta}_1 \leftarrow \theta_1, \bar{\theta}_2 \leftarrow \theta_2$ ▷ Initialize target network weights
$\mathcal{D} \leftarrow \emptyset$ ▷ Initialize an empty replay pool
**for** each iteration **do**
  **for** each environment step **do**
    $\mathbf{a}_t \sim \pi_\phi(\mathbf{a}_t|\mathbf{s}_t)$ ▷ Sample action from the policy
    $\mathbf{s}_{t+1} \sim p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$ ▷ Sample transition from the environment
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{s}_t, \mathbf{a}_t, r(\mathbf{s}_t, \mathbf{a}_t), \mathbf{s}_{t+1})\}$ ▷ Store the transition in the replay pool
  **end for**
  **for** each gradient step **do**
    $\theta_i \leftarrow \theta_i - \lambda_Q \hat{\nabla}_{\theta_i} J_Q(\theta_i)$ for $i \in \{1, 2\}$ ▷ Update the Q-function parameters
    $\phi \leftarrow \phi - \lambda_\pi \hat{\nabla}_\phi J_\pi(\phi)$ ▷ Update policy weights
    $\alpha \leftarrow \alpha - \lambda \hat{\nabla}_\alpha J(\alpha)$ ▷ Adjust temperature
    $\bar{\theta}_i \leftarrow \tau\theta_i + (1 - \tau)\bar{\theta}_i$ for $i \in \{1, 2\}$ ▷ Update target network weights
  **end for**
**end for**
**Output:** $\theta_1, \theta_2, \phi$ ▷ Optimized parameters

*Algorithm 1 of "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 ($\gamma$) | 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 ($\tau$) | 0.005 |
| target update interval | 1 |
| gradient steps | 1 |

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

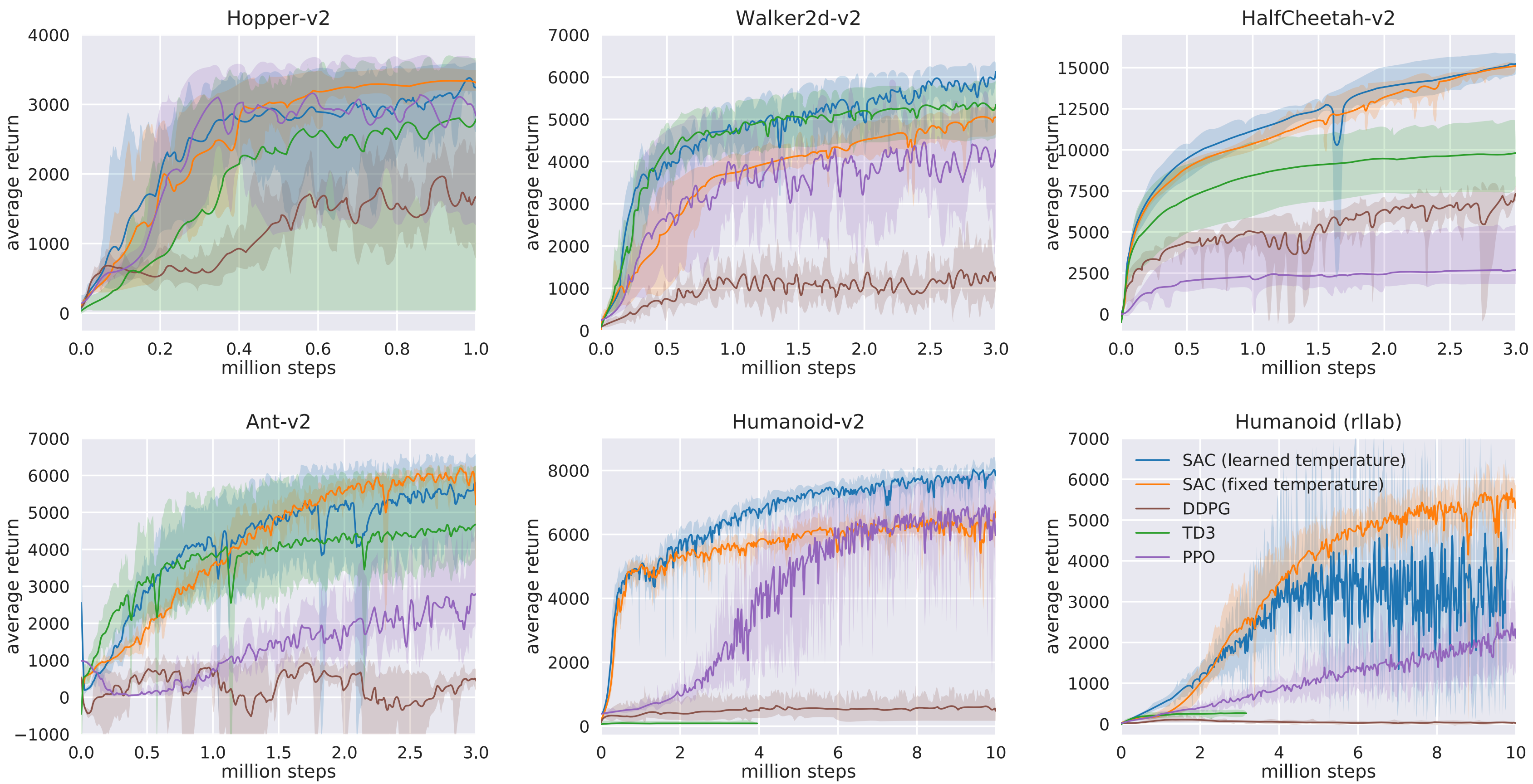
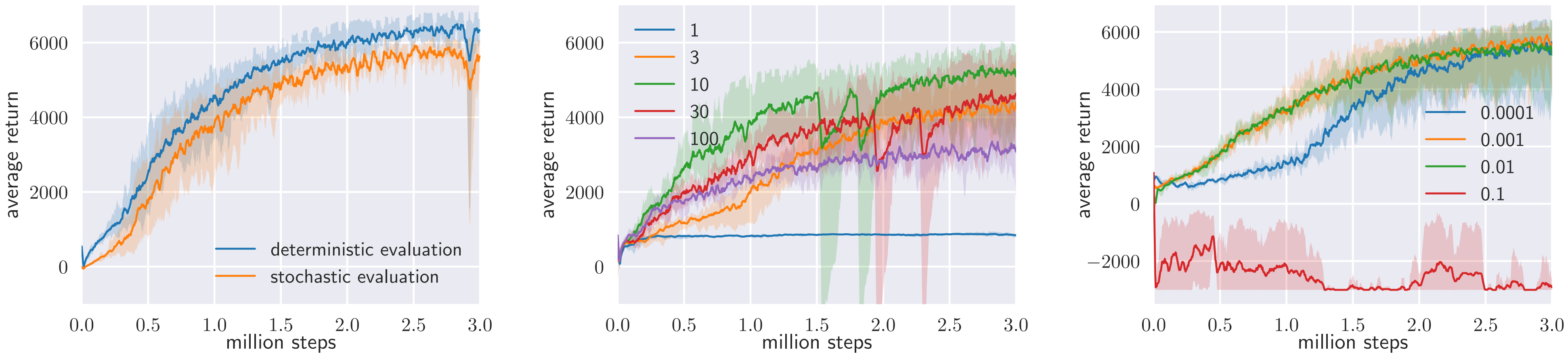



Figure 1: Training curves on continuous control benchmarks. Soft actor-critic (blue and yellow) performs consistently across all tasks and outperforming both on-policy and off-policy methods in the most challenging tasks.

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




(a) Evaluation (b) Reward Scale (c) Target Smoothing Coefficient ($\tau$)

*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 $\tau$ is used to stabilize training. Fast moving target (large $\tau$) can result in instabilities (red), whereas slow moving target (small $\tau$) makes training slower (blue).

*Figure 3 of "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

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

which can be written recursively as

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

For simplicity, we do not explicitly handle the first case ("the episode has already ended") in the following.

# Off-policy Correction Using Control Variates

Note that we can write

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

which yields

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

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:

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

Incidentally, to correctly handle the "the episode has already ended" case, it would be enough to define $\delta_t \stackrel{\text{def}}{=} R_{t+1} + [\neg\text{done}] \cdot \gamma V(S_{t+1}) - V(S_t)$.

# Return Formulations

| Recursive definition | Formulation with TD errors |
| --- | --- |
| $G_{t:t+n} \stackrel{\text{def}}{=} R_{t+1} + \gamma G_{t+1:t+n}$ | $V(S_t) + \sum_{i=0}^{n-1} \gamma^i \delta_{t+i}$ |

# 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

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

which can be extended to

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

# Off-policy Correction Using Control Variates

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 $G_{t:t+n}$

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

the $R_{t+1}$ depends only on $\rho_t$, 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

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

# Return Formulations

| Recursive definition | Formulation with TD errors |
|---|---|
| $G_{t:t+n} \stackrel{\text{def}}{=} R_{t+1} + \gamma G_{t+1:t+n}$ | $V(S_t) + \sum_{i=0}^{n-1} \gamma^i \delta_{t+i}$ |
| $G_{t:t+n}^{\text{IS}} \stackrel{\text{def}}{=} \rho_t \big(R_{t+1} + \gamma G_{t+1:t+n}^{\text{IS}}\big)$ | |

# Off-policy Correction Using Control Variates

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

To utilize this idea, we turn to **control variates**, which is a general method of reducing variance of Monte Carlo estimators. Let $\mu$ be an unknown expectation, which we estimate using an unbiased estimator $m$. Assume we have another **correlated** statistic $k$ with a known expectation $\kappa$.

We can then use an estimate $m^* \stackrel{\text{def}}{=} m - c(k - \kappa)$, which is also an unbiased estimator of $\mu$, with variance

$$\mathrm{Var}(m^*) = \mathrm{Var}(m) + c^2 \,\mathrm{Var}(k) - 2c \,\mathrm{Cov}(m, k).$$

To arrive at the optimal value of $c$, we can set the derivative of $\mathrm{Var}(m^*)$ to 0, obtaining

$$c = \frac{\mathrm{Cov}(m, k)}{\mathrm{Var}(k)}.$$

# Off-policy Correction Using Control Variates

In case of the value function estimate

$$G^{\text{IS}}_{t:t+n} = \rho_t\big(R_{t+1} + \gamma G^{\text{IS}}_{t+1:t+n}\big),$$

we might consider using $\rho_t$ as the correlated statistic $k$, with known expectation $\kappa = 1$, because if $\rho_t \gg 1$, then our return estimate is probably an overestimate, and vice versa.

The optimal value of $c$ should then be

$$c = \frac{\operatorname{Cov}(m, k)}{\operatorname{Var}(k)} = \frac{\mathbb{E}_b\big[(G^{\text{IS}}_{t:t+n} - v_\pi(S_t))(\rho_t - 1)\big]}{\mathbb{E}_b\big[(\rho_t - 1)^2\big]},$$

which is however difficult to compute. Instead, considering the estimate when $\rho_t = 0$, we get

$$\rho_t\big(R_{t+1} + \gamma G^{\text{IS}}_{t+1:t+n}\big) + c(1 - \rho_t) \xlongequal{\rho_t=0} c.$$

Because a reasonable estimate in case of $\rho_t = 0$ is $V(S_t)$, we use $c = V(S_t)$.

# Off-policy Correction Using Control Variates

The estimate with the **control variate** term is therefore

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

which adds no bias, since the expected value of $1 - \rho_t$ is zero and $\rho_t$ and $S_t$ are independent.

Similarly as before, rewriting to

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

results in

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

# Return Formulations

| Recursive definition | Formulation with TD errors |
|---|---|
| $G_{t:t+n} \stackrel{\text{def}}{=} R_{t+1} + \gamma G_{t+1:t+n}$ | $V(S_t) + \sum_{i=0}^{n-1} \gamma^i \delta_{t+i}$ |
| $G_{t:t+n}^{\text{IS}} \stackrel{\text{def}}{=} \rho_t \big(R_{t+1} + \gamma G_{t+1:t+n}^{\text{IS}}\big)$ | |
| $G_{t:t+n}^{\text{CV}} \stackrel{\text{def}}{=} \rho_t \big(R_{t+1} + \gamma G_{t+1:t+n}^{\text{CV}}\big) + (1 - \rho_t)V(S_t)$ | $V(S_t) + \sum_{i=0}^{n-1} \gamma^i \rho_{t:t+i} \delta_{t+i}$ |

# 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 $\frac{2}{3}G_{t:t+2} + \frac{1}{3}G_{t:t+4}$.

# $\lambda$-return

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

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

Alternatively, the $\lambda$-return can be written recursively as

$$\begin{aligned} G_t^\lambda &= (1 - \lambda) G_{t:t+1} \\ &\quad + \lambda (R_{t+1} + \gamma G_{t+1}^\lambda). \end{aligned}$$

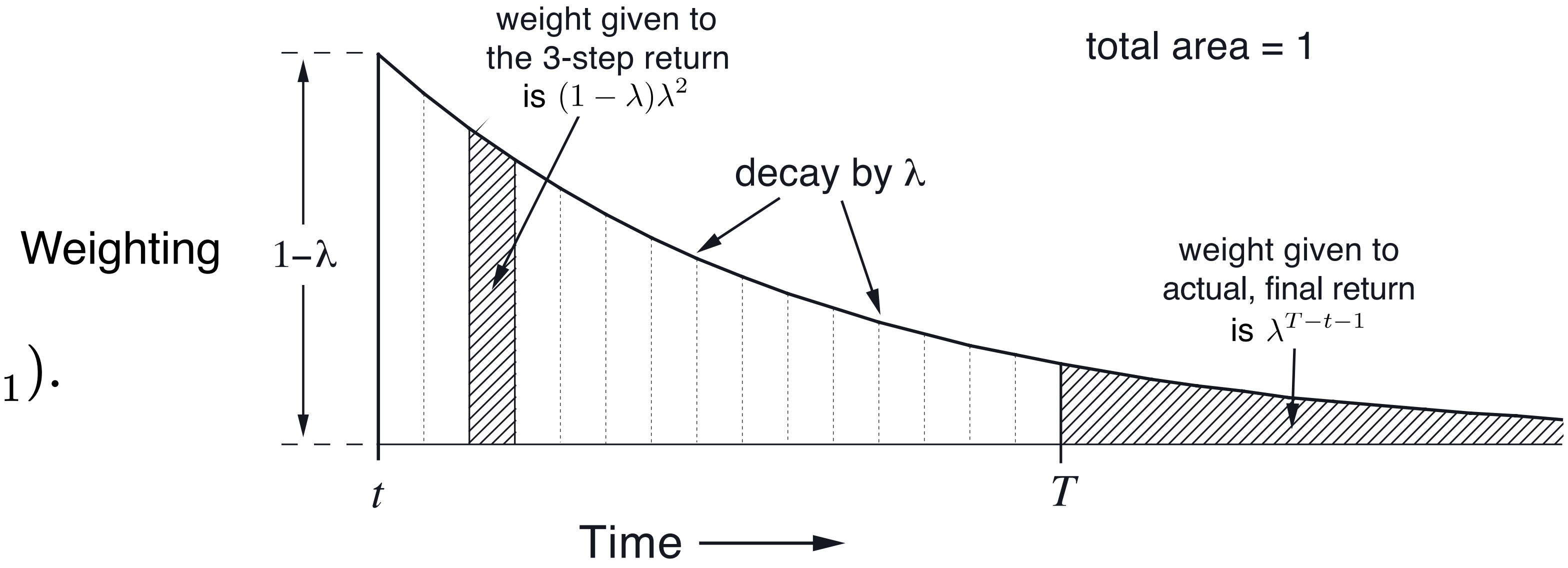



**Figure 12.2:** Weighting given in the $\lambda$-return to each of the $n$-step returns.

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

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

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

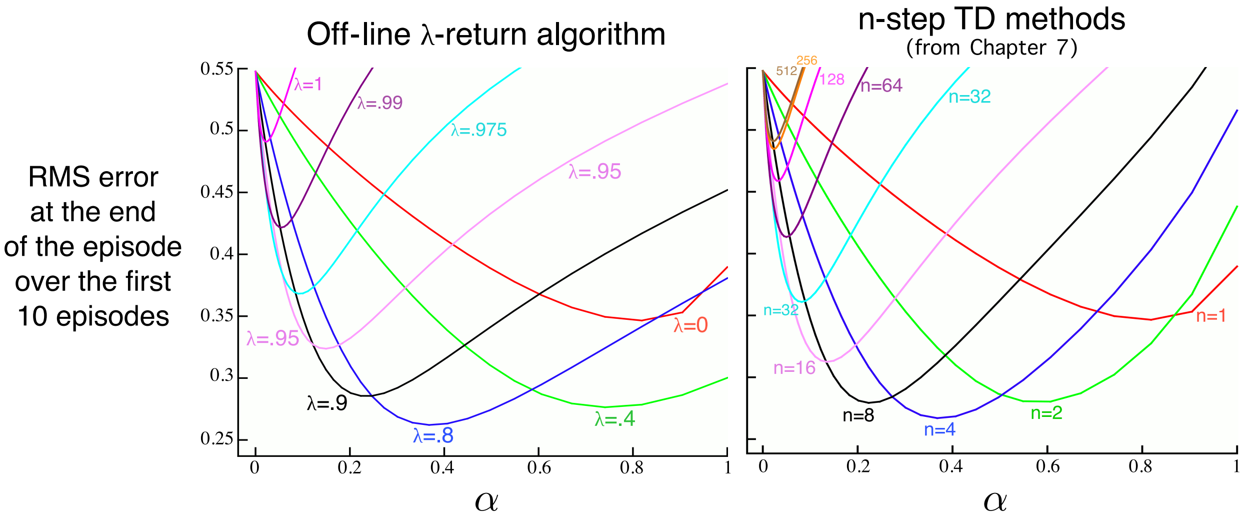



**Figure 12.3:** 19-state Random walk results (Example 7.1): Performance of the off-line $\lambda$-return algorithm alongside that of the $n$-step TD methods. In both case, intermediate values of the bootstrapping parameter ($\lambda$ or $n$) performed best. The results with the off-line $\lambda$-return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

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