Reinforcement Learning

Temporal Difference Algorithms

Lecturer: *Nicolò Cesa-Bianchi* version July 17, 2025

This material is partially based on the book draft "Reinforcement Learning: Foundations" by Shie Mannor, Yishay Mansour, and Aviv Tamar.

We consider the discounted infinite horizon criterion and focus on MDP with finite state space $\mathcal{S}$, finite action space $\mathcal{A}$ such that $\mathcal{A}(s) = \mathcal{A}$ for all $s \in \mathcal{S}$, transition kernel $\{p(\cdot \mid s, a) \,:\, s \in \mathcal{S},\, a \in \mathcal{A}\}$, and time-independent reward function $r : \mathcal{S} \times \mathcal{A} \to [-1, 1]$.

Fix a stationary deterministic policy π and consider the problem of estimating the state-value function V^π. If the MDP were known, we could simply use fixed-policy value iteration or linear programming. When the MDP is unknown, we must use samples from the trajectory generated by π. Recall the system of linear equations that V^π satisfies,

$$V^\pi(s) = r\big(s, \pi(s)\big) + \gamma\, \mathbb{E}\big[V^\pi(s') \mid s\big] \qquad s \in \mathcal{S}$$

where $s' \sim p\big(\cdot \mid s, \pi(s)\big)$. Now, similarly to what we did in Q-learning, we can obtain a sequence $V_0, V_1, \ldots$ of approximations to V^π by running gradient descent on the square loss

$$\ell_t(x) = \frac{1}{2}\Big(x - r\big(s_t, \pi(s_t)\big) - \gamma\, \mathbb{E}\big[V_t(s') \mid s\big]\Big)^2$$

for $x = V_t(s_t)$, which amounts to the update

$$V_{t+1}(s_t) = (1 - \eta_t) V_t(s_t) + \eta_t \Big(r\big(s_t, \pi(s_t)\big) + \gamma\, \mathbb{E}\big[V_t(s') \mid s_t\big]\Big)$$

Since, however, $\mathbb{E}\big[V_t(s') \mid s\big]$ is not directly accessible, we run gradient descent on a perturbed gradient,

$$V_{t+1}(s_t) = (1 - \eta_t) V_t(s_t) + \eta_t \Big(r\big(s_t, \pi(s_t)\big) + \gamma\, V_t(s_{t+1})\Big)$$

where $s_{t+1} \sim p\big(\cdot \mid s_t, \pi(s_t)\big)$. We call **temporal difference** the quantity

$$\Delta_t = r\big(s_t, \pi(s_t)\big) + \gamma\, V_t(s_{t+1}) - V_t(s_t)$$

and write the above update equivalently as

$$V_{t+1}(s_t) = V_t(s_t) + \eta_t \Delta_t$$

The algorithm based on this update is known as TD(0). Similarly to what we did for Q-learning, we can prove the convergence of TD(0) when η_t is a function $\eta_t : \mathcal{S} \to [0, 1]$ of the states defined by

$$\eta_t(s) = \frac{\mathbb{I}\{s = s_t\}}{N_t(s)} \qquad \text{where} \qquad N_t(s) = \sum_{\tau=0}^{t} \mathbb{I}\{s_\tau = s\}$$

Because we focus on deterministic policies, the learning rate η_t can depend only on states rather than on state-action pairs.

Theorem 1 *Assume that* TD(0) *is run with a stationary deterministic policy* π *inducing an irreducible Markov chain on the underlying MDP. Then*

$$\lim_{t\to\infty} V_t(s) = V^\pi(s) \qquad s \in \mathcal{S}$$

with probability 1*.*

The update of TD(0) is based on a 1-step lookahead $R_t^{(1)}(s_t) = r(s_t, a_t) + \gamma\, V_t(s_{t+1})$ so that $\Delta_t = R_t^{(1)}(s_t) - V_t(s_t)$. Using the identity

$$V^\pi(s) = r\big(s, \pi(s)\big) + \mathbb{E}\left[\sum_{\tau=1}^{\infty} \gamma^\tau r\big(s_\tau, \pi(s_\tau)\big) \,\middle|\, s\right] \qquad s \in \mathcal{S}$$

where $s_\tau \sim p\big(\cdot \mid s_{\tau-1}, \pi(s_{\tau-1})\big)$ and $s_0 = s$, TD(0) can be easily generalized to a n-step lookahead

$$R_t^{(n)}(s_t) = \sum_{\tau=0}^{n-1} \gamma^\tau r\big(s_{t+\tau}, \pi(s_{t+\tau})\big) + \gamma^n\, V_t(s_{t+n})$$

The corresponding updates are $V_{t+1}(s_t) = V_t(s_t) + \eta_t \Delta_t^{(n)}$ where $\Delta_t^{(n)} = R_t^{(n)}(s_t) - V_t(s_t)$. Note that

$$\Delta_t^{(n)} = \sum_{\tau=0}^{n-1} \gamma^\tau \Delta_{t+\tau} \tag{1}$$

Indeed,

$$\begin{aligned}
\sum_{\tau=0}^{n-1} \gamma^\tau \Delta_{t+\tau} &= \sum_{\tau=0}^{n-1} \gamma^\tau \Big(r\big(s_{t+\tau}, \pi(s_{t+\tau})\big) + \gamma\, V_t(s_{t+\tau+1}) - V_t(s_{t+\tau}) \Big) \\
&= \sum_{\tau=0}^{n-1} \gamma^\tau\, r\big(s_{t+\tau}, \pi(s_{t+\tau})\big) + \sum_{\tau=0}^{n-1} \Big(\gamma^{\tau+1}\, V_t(s_{t+\tau+1}) - \gamma^\tau\, V_t(s_{t+\tau}) \Big) \\
&= \sum_{\tau=0}^{n-1} \gamma^\tau\, r\big(s_{t+\tau}, \pi(s_{t+\tau})\big) + \gamma^n\, V_t(s_{t+n}) - V_t(s_t) \\
&= R_t^{(n)}(s_t) - V_t(s_t) = \Delta_t^{(n)}
\end{aligned}$$

It can be shown that if we run TD(0) with a n-step lookahead (for any given $n \geq 1$), then $V_t(s)$ converges to $V^\pi(s)$ for all $s \in \mathcal{S}$.

In case of deterministic T (finite horizon), we can choose $n = T$ and set $\gamma = 1$. The resulting algorithm is known as **Monte-Carlo sampling** (see Algorithm 1 below here).

Note that the estimate V_N of the state-value function V^π satisfies

$$V_N(s) = \frac{1}{n(s)} \sum_{i=1}^{N} R_i(s)$$

where $n(s)$ is the number of episodes in which the state s has been visited at least once. For $\alpha > 0$, let $\mathcal{S}_\alpha \subseteq \mathcal{S}$ the set of states such that the probability that π visits $s \in \mathcal{S}_\alpha$ in any given episode is at least α. Then we have the following result.

Algorithm 1 Monte-Carlo sampling for MDP with deterministic horizon T

Input: Stationary deterministic policy π, initial state $s_0 \in \mathcal{S}$, number N of episodes

Set $V_0(s) = 0$ and $n(s) = 0$ for all $s \in \mathcal{S}$; set $s_1 = s_0$
for $i = 1 \dots, N$ **do**
  Use π to generate $(s_1, a_1, r_1), \dots, (s_T, a_T, r_T)$
  for $t = T-1, \dots, 0$ **do**
    if s_t does not appear in $s_0, \dots, s_{t-1}$ **then**
      $R_i(s_t) = r_t + \cdots + r_T$
      $n(s) \leftarrow n(s) + 1$
      Update $V_i(s_t) = V_{i-1}(s_t) + R_i(s_t)$
    end if
  end for
  $s_1 = s_T$
end for
$V_N(s) \leftarrow \dfrac{V_N(s)}{n(s)}$ for all $s \in \mathcal{S}$

Output: $V_N : \mathcal{S} \to \mathbb{R}$

Theorem 2 *Assume that we execute N episodes using policy π and each episode has length at most T. Then, with probability at least $1-\delta$, for any $s \in \mathcal{S}_\alpha$, we have $\big|V_N(s) - V^\pi(s)\big| \le \varepsilon$ for*

$$N \ge \frac{2m}{\alpha} \ln \frac{2|\mathcal{S}|}{\delta} \qquad \text{and} \qquad m = \frac{T^2}{\varepsilon^2} \ln \frac{2|\mathcal{S}|}{\delta}$$

In the discounted setting, the choice of n may impact the quality of the policy evaluation process. Instead of choosing a single value for n, we may average over all positive integers. A simple way of implementing this idea is through exponential averaging with a parameter $\lambda \in (0,1)$. This implies that the weight assigned to each parameter n is $(1-\lambda)\lambda^{n-1}$. This leads to the TD(λ) algorithm.

Recall $\Delta_t^{(n)} = R_t^{(n)}(s_t) - V_t(s_t)$. The TD($\lambda$) update is defined by

$$V_{t+1}(s_t) = V_t(s_t) + (1-\lambda)\eta_t \sum_{n=1}^{\infty} \lambda^{n-1} \Delta_t^{(n)}$$

The problem with this approach is that we have to compute an infinite sum to make a single update. Luckily, there is an equivalent formulation that avoids this problem. The trick is to use the notion of **eligibility trace**

$$e_t(s) = \sum_{k=0}^{t} (\lambda\gamma)^{t-k} \eta_k \mathbb{I}\{s = s_k\}$$

Note that e_t can be recursively computed from $e_0(s) = 0$ and $e_t(s) = (\lambda\gamma) e_{t-1}(s) + \eta_t \mathbb{I}\{s = s_t\}$ for all $s \in \mathcal{S}$.

Now recall the definition of temporal difference,

$$\Delta_t = r\big(s_t, \pi(s_t)\big) + \gamma\, V_t(s_{t+1}) - V_t(s_t)$$

The backward temporal difference is just the standard temporal difference multiplied by the eligibility trace, $\Delta_t^{\mathrm{B}}(s) = \Delta_t\, e_t(s)$. See Algorithm 2 for the pseudocode. Note that in the backward

view all states s get updated at each time step t. The backward view solves the temporal assignment problem: how we assign credit or blame to past actions based on the reward obtained for the current action.

Algorithm 2 TD(λ)

Input: Stationary deterministic policy π, initial state $s_0 \in \mathcal{S}$, parameter $\lambda \in (0,1)$

Set $V_0(s) = 0$ and $e_0(s) = 0$ for all $s \in \mathcal{S}$
for $t = 0, 1, \ldots$ **do**
  Get $a_t = \pi(s_t)$ and observe $r(s_t, a_t)$, $s_{t+1} \sim p(\cdot \mid s_t, a_t)$
  Compute $\Delta_t = r(s_t, a_t) + \gamma\, V_t(s_{t+1}) - V_t(s_t)$
  for $s \in \mathcal{S}$ **do**
    Compute $e_t(s) = (\lambda\gamma) e_{t-1}(s) + \eta_t \mathbb{I}\{s = s_t\}$
    Update $V_{t+1}(s) = V_t(s) + e_t(s)\Delta_t$
  end for
end for

The following result shows that the forward and backward updates

$$V_{t+1}^{\mathrm{F}}(s_t) = V_t^{\mathrm{F}}(s_t) + (1-\lambda)\eta_t \sum_{n=1}^{\infty} \lambda^{n-1} \Delta_t^{(n)} \qquad \text{and} \qquad V_{t+1}^{\mathrm{B}}(s) = V_t^{\mathrm{B}}(s) + \Delta_t^{\mathrm{B}}(s) \quad s \in \mathcal{S}$$

converge to the same limit.

Theorem 3 *Assume*

$$\lim_{t\to\infty} \mathbb{I}\{s_t = s\} = \infty \qquad \textit{and} \qquad \lim_{t\to\infty} V_t^{\mathrm{F}}(s) = V^{\pi}(s)$$

for all $s \in \mathcal{S}$ with probability 1. Let $V_0^{\mathrm{F}}(s) = 0$ and $V_0^{\mathrm{B}}(s) = 0$ for all $s \in \mathcal{S}$. Then

$$\lim_{t\to\infty} V_t^{\mathrm{B}}(s) = V^{\pi}(s) \qquad s \in \mathcal{S}$$

Proof. Fix any $s \in \mathcal{S}$. Since $V_0^{\mathrm{F}}(s) = 0$, $s_t = s$ occurs for infinitely many t with probability 1, and $V_{t+1}^{\mathrm{F}}(s) = V_t^{\mathrm{F}}(s)$ when $s_t \neq s$, we have that

$$\lim_{t\to\infty} V_t^{\mathrm{F}}(s) = \sum_{t=0}^{\infty} \Big(V_{t+1}^{\mathrm{F}}(s) - V_t^{\mathrm{F}}(s) \Big) \mathbb{I}\{s_t = s\}$$

Likewise, using $V_0^{\mathrm{B}}(s) = 0$,

$$\lim_{t\to\infty} V_t^{\mathrm{B}}(s) = \sum_{t=0}^{\infty} \Big(V_{t+1}^{\mathrm{B}}(s) - V_t^{\mathrm{B}}(s) \Big)$$

Therefore, we are left to prove that

$$\sum_{t=0}^{\infty} \Big(V_{t+1}^{\mathrm{F}}(s) - V_t^{\mathrm{F}}(s) \Big) \mathbb{I}\{s_t = s\} = \sum_{t=0}^{\infty} \Big(V_{t+1}^{\mathrm{B}}(s) - V_t^{\mathrm{B}}(s) \Big)$$

We have the following chain of equalities

$$\begin{aligned}
\sum_{t=0}^{\infty}\Big(V_{t+1}^{\mathrm{F}}(s)-V_t^{\mathrm{F}}(s)\Big)\mathbb{I}\{s_t=s\} &= (1-\lambda)\sum_{t=0}^{\infty}\eta_t\mathbb{I}\{s_t=s\}\sum_{n=1}^{\infty}\lambda^{n-1}\Delta_t^{(n)} \\
&= (1-\lambda)\sum_{t=0}^{\infty}\eta_t\mathbb{I}\{s_t=s\}\sum_{n=1}^{\infty}\lambda^{n-1}\sum_{\tau=0}^{n-1}\gamma^{\tau}\Delta_{t+\tau} && \text{(using (1))} \\
&= (1-\lambda)\sum_{t=0}^{\infty}\eta_t\mathbb{I}\{s_t=s\}\sum_{n=0}^{\infty}\lambda^{n}\sum_{\tau=0}^{n}\gamma^{\tau}\Delta_{t+\tau} && \text{(change of variable: } n\leftarrow n-1\text{)} \\
&= (1-\lambda)\sum_{t=0}^{\infty}\eta_t\mathbb{I}\{s_t=s\}\sum_{\tau=0}^{\infty}\sum_{n=\tau}^{\infty}\lambda^{n}\gamma^{\tau}\Delta_{t+\tau} && \text{(exchanging sums)} \\
&= (1-\lambda)\sum_{t=0}^{\infty}\eta_t\mathbb{I}\{s_t=s\}\sum_{k=t}^{\infty}\sum_{n=k-t}^{\infty}\lambda^{n}\gamma^{k-t}\Delta_{k} && \text{(change of variable: } k\leftarrow t+\tau\text{)} \\
&= (1-\lambda)\sum_{t=0}^{\infty}\eta_t\mathbb{I}\{s_t=s\}\sum_{k=t}^{\infty}(\lambda\gamma)^{k-t}\Delta_k\sum_{n=k-t}^{\infty}\lambda^{n-k+t} \\
&= \sum_{t=0}^{\infty}\eta_t\mathbb{I}\{s_t=s\}\sum_{k=t}^{\infty}(\lambda\gamma)^{k-t}\Delta_k && \text{(change of variable: } n\leftarrow n-k+t \text{ and geometric sum)} \\
&= \sum_{k=0}^{\infty}\Delta_k\sum_{t=0}^{k}\eta_t\mathbb{I}\{s_t=s\}(\lambda\gamma)^{k-t} && \text{(exchanging sums)} \\
&= \sum_{k=0}^{\infty}\Delta_k^{\mathrm{B}}(s)=\sum_{k=0}^{\infty}\Big(V_{k+1}^{\mathrm{B}}(s)-V_k^{\mathrm{B}}(s)\Big)
\end{aligned}$$

Since we chose s arbitrarily, the proof is concluded. □

Recall that the actor-critic approach is a method for performing policy iteration without knowning the MDP.

1. Policy evaluation: Run π_t to evaluate $V_t = V^{\pi_t}$
2. Policy improvement: Perform the update $\pi_t \to \pi_{t+1}$

While we can use TD(λ) for the policy evaluation step, the corresponding policy improvement step

$$\pi_{t+1}(s) \in \operatorname*{argmax}_{a\in\mathcal{A}}\Big(r(s,a)+\gamma\mathbb{E}\left[V^{\pi_t}(s')\,\middle|\,s\right]\Big)$$

requires knowing the transition kernel. Since in a model-free setting policy improvement is easy when using the state-action function Q, we can use the version of SARSA (Algorithm 3) to perform policy evaluation. The actor-critic approach then takes the following form.

1. Policy evaluation: Run SARSA on π_t to evaluate $Q_t = Q^{\pi_t}$
2. Policy improvement: Perform the update

$$\pi_{t+1}(s) \in \operatorname*{argmax}_{a\in\mathcal{A}} Q^{\pi_t}(s,a) \qquad s\in\mathcal{S}$$

Algorithm 3 (SARSA for policy evaluation)

Input: Policy π to evaluate, initial state $s_0 \in \mathcal{S}$

Set $Q_0(s,a) = 0$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$
Draw $a_0 \sim \pi(\cdot \mid s_0)$
for $t = 0, 1, \ldots$ **do**
    Observe reward $r_t = r(s_t, a_t)$ and next state s_{t+1} drawn from $p(\cdot \mid s_t, a_t)$
    Draw action $a_{t+1} \sim \pi(\cdot \mid s_{t+1})$
    Update $Q_{t+1}(s_t, a_t) = (1 - \eta_t) Q_t(s_t, a_t) + \eta_t \big(r(s_t, a_t) + \gamma Q_t(s_{t+1}, a_{t+1}) \big)$
end for

Note that the SARSA update step uses a 1-step lookahead: $Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \eta_t \Delta_t$ where $\Delta_t = r(s_t, a_t) + \gamma\, Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t)$. Using the temporal difference idea to increase the lookahead, we define

$$R_t^{(n)}(s_t, a_t) = \sum_{\tau=0}^{n-1} \gamma^\tau r\big(s_{t+\tau}, a_{t+\tau}\big) + \gamma^n\, Q_t(s_{t+n}, a_{t+n})$$

The corresponding updates then are

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \eta_t \Delta_t^{(n)} \qquad \text{where} \qquad \Delta_t^{(n)} = R_t^{(n)}(s_t, a_t) - Q_t(s_t, a_t)$$

We can now define SARSA(λ) using exponential averaging with parameter λ,

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + (1 - \lambda) \eta_t \sum_{n=1}^{\infty} \lambda^{n-1} \Delta_t^{(n)}$$

Now, similarly to TD(λ), we can define a backward view using eligibility traces

$$e_0(s,a) = 0 \qquad \text{and} \qquad e_t(s) = (\lambda\gamma) e_{t-1}(s,a) + \eta_t \mathbb{I}\{s = s_t,\, a = a_t\} \qquad \text{for all } (s,a) \in \mathcal{S} \times \mathcal{A}$$

We can now implement the actor-critic approach using SARSA(λ) instead of SARSA for policy evaluation. Alternatively, we can apply the temporal difference technique to the original version of SARSA to perform on-policy value iteration. The resulting procedure is Algorithm 4 (see next page). Note that in the backward view all state and action pairs (s,a) get updated at each time step t.

Convergence of SARSA(λ) to Q^* is guaranteed when $\pi_{t+1}(\cdot \mid s_{t+1}, Q_t)$ is the ε-greedy policy, where ε_t is chosen as explained in SARSA convergence theorem.

Algorithm 4 SARSA(λ)

Input: Initial random policy π_0, initial state $s_0 \in \mathcal{S}$, parameter $\lambda \in (0, 1)$

Set $Q_0(s, a) = 0$ and $e_0(s, a) = 0$ for all $(s, a) \in \mathcal{S} \times \mathcal{A}$
Draw $a_0 \sim \pi_0(\cdot \mid s_0)$
for $t = 0, 1, \ldots$ **do**
    Play a_t and observe $r(s_t, a_t)$, $s_{t+1} \sim p(\cdot \mid s_t, a_t)$
    Draw $a_{t+1} \sim \pi_{t+1}(\cdot \mid s_{t+1}, Q_t)$
    Compute $\Delta_t = r(s_t, a_t) + \gamma\, Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t)$
    for $(s, a) \in \mathcal{S} \times \mathcal{A}$ **do**
        Compute $e_t(s, a) = (\lambda\gamma) e_{t-1}(s, a) + \eta_t \mathbb{I}\{s = s_t,\, a = a_t\}$
        Update $Q_{t+1}(s, a) = Q_t(s, a) + e_t(s, a)\Delta_t$
    end for
end for