Reinforcement Learning

Prof. Volkan Cevher
volkan.cevher@epfl.ch

Lecture 2: Dynamic Programming

Laboratory for Information and Inference Systems (LIONS)
École Polytechnique Fédérale de Lausanne (EPFL)

EE-568 (Spring 2024)

License Information for Reinforcement Learning (EE-568)

A refresher on Markov chains – I

Definition (Markov Chain)

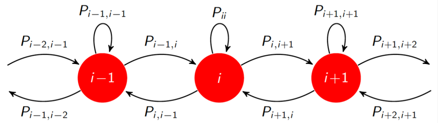
A (time-homogeneous) Markov chain is a stochastic process $\{X_0, X_1, \ldots\}$, *taking values on a countable number of states, satisfying the so-called Markov property, i.e.,*

$$\mathbb{P}[X_{t+1} = j | X_t = i, X_{t-1}, \ldots, X_0] = \mathbb{P}[X_{t+1} = j | X_t = i] = P_{ij}.$$

Markov Process

Markov process is a triple $\langle \mathcal{S}, \mathrm{P}, \mu \rangle$, where

- $\mathcal{S}$ is the set of all possible states
- The matrix P with entries $[P]_{ss'} = \mathrm{P}(s'|s)$ is the transition matrix over $\mathcal{S}$
- μ is the initial state distribution: $s_0 \sim \mu \in \Delta(\mathcal{S})$



A refresher on Markov chains – II

Definition (Stationary distribution)

If a Markov chain is irreducible and aperiodic with finite states (i.e., ergodic), then there exists a unique stationary distribution $\mathrm{d}^\star$ *and* $\{X_t\}$ *converges to it, i.e.,* $\lim_{t\to\infty}[P^t]_{ij} = \mathrm{d}^\star_j, \forall i, j$. *We can represent this via* $\mathrm{d}^\star = \mathrm{d}^\star P$ *where* $[P]_{ij} = P_{ij}$ *and* $\mathrm{d}^\star$ *is a row vector. Hence,* $\mathrm{d}^\star$ *is the left principal eigenvector of* P.

Remarks:

- Irreducibility:
  - A Markov chain is *irreducible* if it is possible to reach any state from any state.
  - Ensures the chain forms a single communicating graphical model.
- Aperiodicity:
  - A Markov chain is *aperiodic* if every state has a period of 1.
  - Prevents the chain from getting stuck in cycles, allowing thorough mixing.
- Practical Implications:
  - Convergence: Irreducible and aperiodic chains converge to a unique stationary distribution.
  - Ergodicity: Enables estimation of long-term averages by simulation.
  - Mixing Time: Affects the efficiency of simulations and probabilistic modeling.

Markov Decision Processes (MDPs)

Markov Decision Process

An MDP is a tuple $(\mathcal{S}, \mathcal{A}, \mathsf{P}, r, \mu, \gamma)$, where

- $\mathcal{S}$ is the set of all possible states.
- $\mathcal{A}$ is the set of all possible actions.
- For each action a, the matrix P^a with entries $[P^a]_{ss'} = \mathsf{P}(s'|s,a)$ is the transition matrix $(\mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S}))$.
- $r \colon \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function. We assume $r \in [0,1]$.
- μ is the initial state distribution: $s_0 \sim \mu \in \Delta(\mathcal{S})$.
- γ is the discount factor: $\gamma \in (0,1)$.

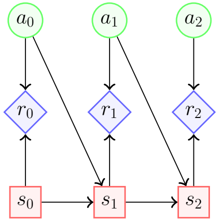
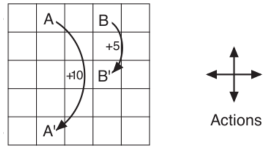


Figure: An MDP graphical model

Example: Gridworld

- State $\mathcal{S}$: the agent's position
- Action $\mathcal{A}$: moving north/south/east/west
- Reward r:
  - -1 if moving outside the world
  - $+10$ if moving to A
  - $+5$ if moving to B
  - 0 otherwise
- Transition model P:
  - move to the adjacent grid according to the direction
  - stay unchanged if moving toward the wall
  - transit to A' if moving into A, transit to B' if moving into B



MDPs: policies

What is our goal?

Find a behaviour or rule to make decisions that maximize the expected return.

∘ In general, a policy selects an action based on the history $h_t := (s_{0:t}, a_{0:t-1}) := (s_0, a_0, \ldots, s_{t-1}, a_{t-1}, s_t)$

- A stationary Markov policy is a mapping $\pi : \mathcal{S} \to \mathcal{A}$ or $\pi : \mathcal{S} \to \Delta(\mathcal{A})$,
- Δ is the appropriate probability simplex.

Deterministic Policy

- Stationary policy $\pi : \mathcal{S} \to \mathcal{A}$, $a_t = \pi(s_t)$
- Markov policy $\pi_t : \mathcal{S} \to \mathcal{A}$, $a_t = \pi_t(s_t)$
- History-dependent policy $\pi_t : \mathcal{H}_t \to \mathcal{A}$
  - $\mathcal{H}_t$ is the set of histories up to time t.
  - $a_t = \pi_t(h_t)$

Randomized Policy:

- Stationary policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$, $a_t \sim \pi(\cdot|s_t)$
- Markov policy $\pi_t : \mathcal{S} \to \Delta(\mathcal{A})$, $a_t \sim \pi_t(\cdot|s_t)$
- History-dependent policy $\pi_t : \mathcal{H}_t \to \Delta(\mathcal{A})$
  - $\mathcal{H}_t$ is the set of histories up to time t.
  - $a_t \sim \pi_t(\cdot|h_t)$

Remarks: ∘ The infinite horizon objective can be maximized by a *stationary deterministic policy*.

∘ The finite horizon objective needs instead a (nonstationary) *deterministic Markov policy*.

From MDPs to performance criteria

Reminder:

- We have described the role of MDPs while establishing a performance criterion.
  - Finite Horizon: Cumulative reward and average reward.
  - Infinite Horizon: Discounted reward and average reward.
- In this course, we mainly focus on discounted infinite-horizon MDPs:

$$J(\pi) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \middle| s_0 \sim \mu,\ \pi\right].$$

- We use $\gamma \in (0,1)$ to trade off past and present rewards.

Observations:

- If $\gamma = 1$, the total reward may be infinite, e.g., when the Markov process is cyclic.
- With $\gamma \in (0,1)$, assuming bounded rewards, i.e., $r < \infty$, the total return will always be finite.

Value functions

Definition (State-Value Function)

$$V^\pi(s) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s,\ \pi\right]$$

Value functions

Definition (State-Value Function)

$$V^\pi(s) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s,\ \pi\right]$$

Definition (Quality Function / State-Action Value Function)

$$Q^\pi(s, a) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s,\ a_0 = a,\ \pi\right]$$

Observations:

- ◦ $V^\pi(s)$ represents the total expected return starting at state s under policy π.
- ◦ $Q^\pi(s, a)$ represents the total expected return when choosing action a in state s under policy π.
- ◦ For convenience, we may drop the π in RHS when it is clear from the context.

Remark:

- ◦ In the literature, state-value function and value function are used interchangeably.

Value functions (cont'd)

Pop quiz: ◦ What is the relation between V^π and Q^π?

Value functions (cont'd)

Pop quiz: ◦ What is the relation between V^π and Q^π?

Answer: ◦ For any policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$, it holds that

$$Q^\pi(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a) V^\pi(s') \tag{1}$$

$$V^\pi(s) = \sum_{a \in \mathcal{A}} \pi(a \,|\, s)\, Q^\pi(s,a) \tag{2}$$

Proof of equation (1)

Derivation:

$$\begin{aligned} Q^\pi(s,a) &= \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s,\, a_0 = a,\, \pi\right] \\ &= r(s,a) + \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s,\, a_0 = a,\, \pi\right] \end{aligned}$$

Proof of equation (1)

Derivation:

$$\begin{aligned}
Q^\pi(s,a) &= \mathbb{E}\left[\sum\nolimits_{t=0}^{\infty} \gamma^t r(s_t,a_t) \mid s_0 = s,\, a_0 = a,\, \pi\right] \\
&= r(s,a) + \mathbb{E}\left[\sum\nolimits_{t=1}^{\infty} \gamma^t r(s_t,a_t) \mid s_0 = s,\, a_0 = a,\, \pi\right] \\
&= r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a)\, \mathbb{E}\left[\sum\nolimits_{t=1}^{\infty} \gamma^{t-1} r(s_t,a_t) \mid s_0 = s,\, s_0 = s',\, a_0 = a,\, \pi\right]
\end{aligned}$$

Proof of equation (1)

Derivation:

$$
\begin{aligned}
Q^\pi(s,a) &= \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t,a_t) \mid s_0 = s,\, a_0 = a,\, \pi\right] \\
&= r(s,a) + \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^t r(s_t,a_t) \mid s_0 = s,\, a_0 = a,\, \pi\right] \\
&= r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a)\, \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t,a_t) \mid s_0 = s,\, s_0 = s',\, a_0 = a,\, \pi\right] \\
&= r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a)\, \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t,a_t) \mid s_0 = s',\, \pi\right] \qquad \text{(Markov assumption)}
\end{aligned}
$$

Proof of equation (1)

Derivation:

$$
\begin{aligned}
Q^\pi(s,a) &= \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t r(s_t,a_t) \mid s_0 = s,\, a_0 = a,\, \pi\Big] \\
&= r(s,a) + \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^t r(s_t,a_t) \mid s_0 = s,\, a_0 = a,\, \pi\Big] \\
&= r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a)\, \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t,a_t) \mid s_0 = s,\, s_0 = s',\, a_0 = a,\, \pi\Big] \\
&= r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a)\, \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t,a_t) \mid s_0 = s',\, \pi\Big] \qquad \text{(Markov assumption)} \\
&= r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a)\, \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^{t} r(s_t,a_t) \mid s_0 = s',\, \pi\Big] \qquad \big(\text{i.e., } V^\pi(s')\big)
\end{aligned}
$$

Proof of equation (1)

Derivation:

$$
\begin{aligned}
Q^\pi(s,a) &= \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t,a_t) \mid s_0 = s,\, a_0 = a,\, \pi\right] \\
&= r(s,a) + \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^t r(s_t,a_t) \mid s_0 = s,\, a_0 = a,\, \pi\right] \\
&= r(s,a) + \gamma \sum_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a)\ \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t,a_t) \mid s_0 = s,\, s_0 = s',\, a_0 = a,\, \pi\right] \\
&= r(s,a) + \gamma \sum_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a)\ \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t,a_t) \mid s_0 = s',\, \pi\right] \qquad \text{(Markov assumption)} \\
&= r(s,a) + \gamma \sum_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a)\ \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^{t} r(s_t,a_t) \mid s_0 = s',\, \pi\right] \qquad \left(\text{i.e., } V^\pi(s')\right) \\
&= r(s,a) + \gamma \sum_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a)\ V^\pi(s') \square
\end{aligned}
$$

Optimal value functions

◦ Let Π be the set of all (possibly non-stationary and randomized) policies.

Definition (Optimal Value Function)

$$V^\star(s) := \max_{\pi \in \Pi} V^\pi(s)$$

Definition (Optimal Action-Value Function)

$$Q^\star(s,a) := \max_{\pi \in \Pi} Q^\pi(s,a)$$

Pop quiz: ◦ What is the relation between $V^\star$ and $Q^\star$?

Optimal value functions

∘ Let Π be the set of all (possibly non-stationary and randomized) policies.

Definition (Optimal Value Function)

$$V^\star(s) := \max_{\pi \in \Pi} V^\pi(s)$$

Definition (Optimal Action-Value Function)

$$Q^\star(s,a) := \max_{\pi \in \Pi} Q^\pi(s,a)$$

Pop quiz: ∘ What is the relation between $V^\star$ and $Q^\star$?

Answer:

$$Q^\star(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a) V^\star(s') \tag{3}$$

$$V^\star(s) = \max_{a \in \mathcal{A}} Q^\star(s,a) \tag{4}$$

∘ *Self-exercise*: prove equation (4).

Solving MDPs: find the optimal policy

Goal

Roughly speaking, the ultimate goal in RL can be summed up to finding an optimal policy $\pi^\star \in \Pi$ such that

$$V^{\pi^\star}(s) = V^\star(s) := \max_{\pi \in \Pi} V^\pi(s), \forall s \in \mathcal{S}.$$

Remark: ○ The optimal policy may not be unique, while $V^\star$ is unique.

Key Questions

- **Q1**: Does the optimal policy $\pi^\star$ exist?
- **Q2**: How to evaluate my current policy π, i.e., how to compute $V^\pi(s)$? *–policy evaluation*
- **Q3**: If $\pi^\star$ exists, how to improve my current policy π, i.e., how to find $\pi^\star$? *–policy improvement*

Bellman optimality conditions

◦ The optimal value function $V^\star$ is the unique fixed point of the following equation:

$$V^\star(s) = \max_{a \in \mathcal{A}} \left[r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a) V^\star(s') \right].$$

Remarks:

◦ This requirement is also known as the Bellman optimality equation.

◦ We will show that there exists a deterministic optimal policy.

◦ Fixed-point perspective motivates value iteration (VI) and policy iteration (PI) methodologies.

Existence of an optimal policy

Theorem (Existence of an optimal policy [1] [14])

For an infinite horizon MDP $M = (\mathcal{S}, \mathcal{A}, P, t, \mu, \gamma)$, there exists a stationary and deterministic policy π such that for any $s \in \mathcal{S}$ and $a \in \mathcal{A}$, we have

$$V^{\pi}(s) = V^{\star}(s), \quad Q^{\pi}(s,a) = Q^{\star}(s,a).$$

Remarks:

- ◦ Finding $\pi^{\star}$ can be done by first computing $V^{\star}$ or $Q^{\star}$.
- ◦ Note that we can directly get a (deterministic and stationary) optimal policy from $Q^{\star}$:

$$\pi^{\star}(s) = \arg\max_{a \in \mathcal{A}} Q^{\star}(s,a).$$

- ◦ Note: Proof of the theorem can be found the supplementary slides #8.

Bellman consistency equation

Richard Ernest Bellman
(August 26, 1920 – March 19, 1984)

Theorem (Bellman Consistency Equation)

$$V^{\pi}(s) = \mathbb{E}_{a\sim\pi(\cdot|s)}\left[r(s,a) + \gamma \sum_{s'\in\mathcal{S}} P(s'|s,a)V^{\pi}(s')\right] \qquad \text{(BCE)}$$

Bellman consistency equation

Richard Ernest Bellman
(August 26, 1920 – March 19, 1984)

Theorem (Bellman Consistency Equation)

$$V^\pi(s) = \mathbb{E}_{a\sim\pi(\cdot|s)}\left[r(s,a) + \gamma \sum_{s'\in\mathcal{S}} P(s'|s,a)V^\pi(s')\right] \qquad \text{(BCE)}$$

Remarks:

- BCE is also known as Bellman expectation equation.
- BCE states the value of a state under a given policy π, which is
  - the expected return starting from that state, taking an action according to the policy,
  - ... and thereafter following the policy....

Bellman consistency equation

Richard Ernest Bellman
(August 26, 1920 – March 19, 1984)

Theorem (Bellman Consistency Equation)

$$V^\pi(s) = \mathbb{E}_{a\sim\pi(\cdot|s)}\left[r(s,a) + \gamma \sum_{s'\in\mathcal{S}} P(s'|s,a)V^\pi(s')\right] \qquad \text{(BCE)}$$

Matrix Form

We can concisely represent the Bellman consistency equation in the following matrix form: $V^\pi = R^\pi + \gamma P^\pi V^\pi$.

- We can derive from equations (1) and (2):

$$Q^\pi(s,a) = r(s,a) + \gamma \sum_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a)V^\pi(s') \quad (1)$$

$$V^\pi(s) = \sum_{a\in\mathcal{A}} \pi(a\,|\,s)\, Q^\pi(s,a) \quad (2)$$

- We can write, with $|\mathcal{S}|$ being the cardinality of $\mathcal{S}$:

$$V^\pi \in \mathbb{R}^{|\mathcal{S}|} : V^\pi_s = V^\pi(s);$$

$$R^\pi \in \mathbb{R}^{|\mathcal{S}|}, R^\pi_s := \sum_{a\in\mathcal{A}} \pi(a|s)r(s,a);$$

$$P^\pi \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|} : P^\pi_{s,s'} := \sum_{a\in\mathcal{A}} \pi(a|s)\mathsf{P}(s'\,|\,s,a).$$

Closed-form solution for policy evaluation

Closed-Form Solution of V^π

Given the matrix form of BCE, we have the following closed-form solution: $V^\pi = (I - \gamma P^\pi)^{-1} R^\pi$.

Remarks:

- This is one of exact solution methods for policy evaluation.
- Note that the matrix $I - \gamma P^\pi$ is always invertible for $\gamma \in (0,1)$.
- The solution of Bellman equation is always unique.
- Computation cost: $\mathcal{O}(|\mathcal{S}|^3 + |\mathcal{S}|^2|\mathcal{A}|)$, which can be expensive for large state spaces.

Bellman expectation operator and fixed-point perspective

Definition (Bellman expectation operator)

The Bellman expectation operator $\mathcal{T}^\pi : \mathbb{R}^{|\mathcal{S}|} \to \mathbb{R}^{|\mathcal{S}|}$ *is defined by the following expression*

$$\mathcal{T}^\pi V := R^\pi + \gamma P^\pi V. \tag{5}$$

Remarks:

- ◦ BCE implies that V^π is the fixed point of $\mathcal{T}^\pi$: $\mathcal{T}^\pi V^\pi = V^\pi$.
- ◦ $\mathcal{T}^\pi$ is a linear operator and is a γ-contraction mapping.
- ◦ The solution of BCE is always unique.
- ◦ For the following iteration invariant $V_{t+1} = \mathcal{T}^\pi V_t$, $t = 0, 1, \ldots$, it holds that

$$\lim_{t\to\infty} (\mathcal{T}^\pi)^t V_0 = V^\pi.$$

Bellman optimality conditions

Theorem (Bellman optimality equation)

The optimal value and action-value functions satisfy the following equations:

$$V^\star(s) = \max_{a\in\mathcal{A}} \left[r(s,a) + \gamma \sum_{s'\in\mathcal{S}} P(s'|s,a) V^\star(s') \right],$$

$$Q^\star(s,a) = r(s,a) + \gamma \left[\sum_{s'\in\mathcal{S}} P(s'|s,a) \left(\max_{a'\in\mathcal{A}} Q^\star(s',a') \right) \right].$$

Remarks:
- ◦ These requirements are also known as Bellman optimality conditions.
- ◦ Obtained by combining equations (3) and (4).
- ◦ Fixed-point perspective motivates value iteration (VI) and policy iteration (PI) methodologies.

Bellman optimality operator

Definition (Bellman optimality operator)

We define the following operator $\mathcal{T}$, which will be useful when discussing value-iteration in the sequel:

$$(\mathcal{T}V)(s) := \max_{a\in\mathcal{A}} \left[r(s,a) + \gamma \sum_{s'\in\mathcal{S}} P(s'|s,a)V(s') \right]. \qquad \text{(Bellman operator)}$$

Remarks:

- The optimal value function $V^\star$ is the **fixed point** of $\mathcal{T}$, i.e.,

$$\mathcal{T}V^\star = V^\star.$$

- The Bellman optimality operator is a γ-contraction mapping w.r.t. ℓ_∞-norm:

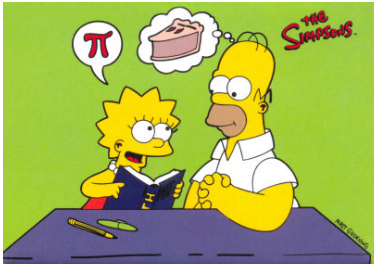
$$\left\|\mathcal{T}V' - \mathcal{T}V\right\|_\infty \le \gamma \left\|V' - V\right\|_\infty.$$

- The Bellman operator is also monotonic (component-wise): $V' \le V \;\Rightarrow\; \mathcal{T}V' \le \mathcal{T}V$.
- We can define a similar Bellman operator on the Q-function and show similar properties.

Pause and reflect

◦ Before we move on, take a minute to reflect on these important notations:

▷ π, $\pi^\star$, $V^\pi(s)$, $V^\star(s)$, $Q^\pi(s,a)$, $Q^\star(s,a)$, $\mathcal{T}^\pi$, $\mathcal{T}$



Solving MDPs

◦ What we talked about:

- Optimal state-value function ($V^\star(s)$) and optimal action-value Function ($Q^\star(s,a)$).
- Bellman consistency equation ($V^\pi = R^\pi + \gamma P^\pi V^\pi$).
- Bellman expectation operator and fixed-point perspective ($\mathcal{T}^\pi V := R^\pi + \gamma P^\pi V$).
- Bellman optimality equations and Bellman optimality operator.

◦ How do we use this to do "planning," i.e., finding an optimal policy via MDPs (our goal)?

Algorithm	**Component**	**Output**
Value Iteration (VI)	Bellman Optimality Operator $\mathcal{T}$	V_T such that $\|V_T - V^\star\| \leq \epsilon$
Policy Iteration (PI).	Bellman Operator $\mathcal{T}^\pi$ + Greedy Policy	$V^\star$ and $\pi^\star$

Observation: ◦ These solutions require, and we assume throughout, that the transitions dynamics are known.

Value iteration (VI)

Algorithm: Value Iteration (VI) for solving MDPs

```
Start with an arbitrary guess $V_0$ (e.g., $V_0(s) = 0$ for any $s$)
for each iteration $t$ do
  Apply the BELLMAN OPERATOR $\mathcal{T}$ to the current value estimate $V_t$:
      $$V_{t+1} = \mathcal{T} V_t.$$
end for
```

Remarks:

- ◦ Finding $V^\star$ or $\pi^\star$ is equivalent to finding a fixed point of $\mathcal{T}$.
- ◦ Value iteration can be therefore viewed as a fixed-point iteration.

Discussion on value iteration

◦ After obtaining $V^\star$ via VI, we can obtain an optimal policy from the greedy policy:

$$\pi^\star(s) = \arg\max_{a\in\mathcal{A}} \left[r(s,a) + \gamma \sum_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a) V^\star(s') \right].$$

Discussion on value iteration

- After obtaining $V^\star$ via VI, we can obtain an optimal policy from the greedy policy:

$$\pi^\star(s) = \arg\max_{a\in\mathcal{A}} \left[r(s,a) + \gamma \sum_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a) V^\star(s') \right].$$

- Alternatively, we can run Q-value iteration and compute $\pi^\star$ via

$$\pi^\star(s) = \arg\max_{a\in\mathcal{A}} \; Q^\star(s,a).$$

Remarks:

- Q-value iteration uses the following update derived from equations (1) and (2):

$$Q_{t+1}(s,a) = r(s,a) + \gamma \sum_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a) \max_{a'\in\mathcal{A}} Q_t(s',a'), \tag{1}$$

- The Q-value iteration does not require knowledge of P to extract the policy $\pi^\star$.
- This observation is the starting point to develop "model-free" algorithms in the sequel.

Convergence of value iteration

Theorem (Linear Convergence of Value Iteration)

The value iteration algorithm attains a linear convergence rate, i.e.,

$$\|V_t - V^\star\|_\infty \leq \gamma^t \|V_0 - V^\star\|_\infty.$$

Convergence of value iteration

Theorem (Linear Convergence of Value Iteration)

The value iteration algorithm attains a linear convergence rate, i.e.,

$$\|V_t - V^\star\|_\infty \le \gamma^t \|V_0 - V^\star\|_\infty.$$

Proof.

$$\|V_t - V^\star\|_\infty = \|\mathcal{T}V_{t-1} - \mathcal{T}V^\star\|_\infty \le \gamma\|V_{t-1} - V^\star\|_\infty \le \cdots \le \gamma^t\|V_0 - V^\star\|_\infty.$$

□

Remarks:

- The complexity of applying $\mathcal{T}$ is $\mathcal{O}\left(|\mathcal{S}|^2|\mathcal{A}|\right)$.
- The number of iterations to reach ϵ accuracy is $\mathcal{O}\left(\log \epsilon^{-1}\right)$ due to linear convergence.

Directly update the policy

∘ Value iteration first finds $V^\star$, then computes the optimal policy $\pi^\star$ by the greedy policy.

Directly update the policy

◦ Value iteration first finds $V^\star$, then computes the optimal policy $\pi^\star$ by the greedy policy.

◦ We can also directly search for the optimal policy $\pi^\star$.

Some intuition: ◦ Starting with an initial guess π, we can iteratively perform the following motions:

1. Evaluate policy: compute the value function V^π of the current policy

   $\Rightarrow$ Policy evaluation

2. Improve policy: update the guess by the greedy policy w.r.t. V^π

   $\Rightarrow$ Policy improvement

Policy improvement theorem

Theorem (Policy Improvement)

If a (deterministic) policy π' satisfies the following

$$Q^{\pi}(s, \pi'(s)) \geq V^{\pi}(s) \quad \forall\, s \in \mathcal{S}, \tag{6}$$

then we have $V^{\pi'}(s) \geq V^{\pi}(s)$ for any $s \in \mathcal{S}$.

Remarks:

- The same result holds for a stochastic policy π' if $\mathbb{E}_{a\sim\pi'(\cdot|s)} Q^{\pi}(s,a) \geq V^{\pi}(s) \quad \forall\, s \in \mathcal{S}$.
- Improving the current policy by one step everywhere, we can improve the whole policy.
- It suggests a natural way of improving the current policy via

$$\pi_{t+1}(s) \leftarrow \arg\max_{a\in\mathcal{A}} Q^{\pi_t}(s,a).$$

- Indeed, $V^{\pi_{t+1}}(s) \geq V^{\pi_t}(s), \forall\, s \in \mathcal{S}$, and the inequality is strict if π_t is suboptimal.

Policy iteration

Algorithm: Policy Iteration (PI) for solving MDPs

```
Start with an arbitrary policy guess $\pi_0$
for each iteration $t$ do
  (Step 1: Policy evaluation) Compute $V^{\pi_t}$:
      (Option 1) Iteratively apply policy value iteration, $V_t \leftarrow \mathcal{T}^{\pi_t} V_t$, until convergence
      (Option 2) Use the closed-form solution: $V^{\pi_t} = (I - \gamma P^{\pi_t})^{-1} R^{\pi_t}$
  (Step 2: Policy improvement) Update the current policy $\pi_t$ by the greedy policy
      $\pi_{t+1}(s) = \arg\max_{a \in \mathcal{A}} \left[ r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'|s,a) V^{\pi_t}(s') \right].$   (7)
end for
```

Remarks:

- ◦ Recall that we assume that there exists a deterministic optimal policy.
- ◦ Greedy policy achieves the optimal deterministic policy.

Comparison

Algorithm	Value Update	Policy Update
Value Iteration (VI)	$V_{t+1} = \mathcal{T} V_t.$	None
Policy Iteration (PI)	$V_{t+1} = \mathbb{E}\left[r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathsf{P}(s'\|s,a) V(s') \| \pi_t\right]$	Greedy Policy

Algorithm	Per iteration cost	Number of iterations	Output
Value Iteration (VI)	$\mathcal{O}\left(\|\mathcal{S}\|^2 \|\mathcal{A}\|\right)$	$T = \mathcal{O}\left(\frac{\log(\epsilon(1-\gamma))}{\log \gamma}\right)$	V_T such that $\\|V_T - V^\star\\| \leq \epsilon$
Policy Iteration (PI)	$\mathcal{O}\left(\|\mathcal{S}\|^3 + \|\mathcal{S}\|^2 \|\mathcal{A}\|\right)$	$T = \mathcal{O}\left(\frac{\|\mathcal{S}\|(\|\mathcal{A}\|-1)}{1-\gamma}\right)$	$V^\star$ and $\pi^\star$

Observations:
- ◦ VI and PI are broadly dynamic programming approaches.
- ◦ PI converges in finite number of iterations [16] whereas VI does not [15].
- ◦ These solution mythologies are broadly known as model-based RL.
- ◦ Modified policy iteration [17] performs limited value-function updates for speed-ups.

Convergence of policy iteration (PI)

Theorem (Linear Convergence of Policy Iteration)

Policy iteration outputs the optimal policy after $\mathcal{O}\left(\frac{|\mathcal{S}||\mathcal{A}|}{1-\gamma}\right)$ iterations.

Proof.

- For simplicity, we provide just the proof sketch.
- The first step is to prove that PI identifies a suboptimal action at a certain state every $\mathcal{O}\left(\frac{1}{1-\gamma}\right)$.
- The proof is concluded noticing that there exists at most $|\mathcal{S}|\left(|\mathcal{A}|-1\right)$ suboptimal actions. □

Summary I

◦ Basic concepts of Markov decision process (MDP)

- Policy, value functions, optimal value functions
- Bellman equations and Bellman operators
- Fixed point viewpoints
- Existence and construction of optimal policy

Summary I

- Basic concepts of Markov decision process (MDP)
  - Policy, value functions, optimal value functions
  - Bellman equations and Bellman operators
  - Fixed point viewpoints
  - Existence and construction of optimal policy
- Exact solution methods for policy evaluation

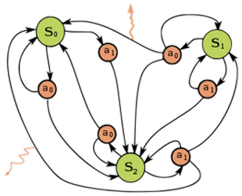
Summary I

- Basic concepts of Markov decision process (MDP)
  - Policy, value functions, optimal value functions
  - Bellman equations and Bellman operators
  - Fixed point viewpoints
  - Existence and construction of optimal policy
- Exact solution methods for policy evaluation
- Exact solution methods for solving MDPs
  - Value iteration: iteratively apply Bellman operator
  - Policy iteration: alternatively execute policy evaluation and policy improvement

From planning to reinforcement learning

Fundamental Challenge 1

The dynamic programming approaches (VI and PI) as well as the linear programming approach all require the full knowledge of the transition model P and the reward.

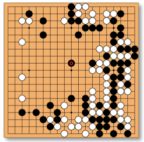


Fundamental Challenge 2

The computation and memory cost can be very expensive for large scale MDP problems.

Chess: 10^{120}

Go: 3^{361}

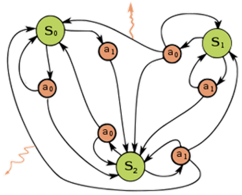


From planning to reinforcement learning

Fundamental Challenge 1

The dynamic programming approaches (VI and PI) as well as the linear programming approach all require the full knowledge of the transition model P and the reward.

⇒Need sampling approaches

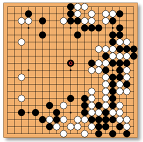


Fundamental Challenge 2

The computation and memory cost can be very expensive for large scale MDP problems.

Chess: 10^{120}

Go: 3^{361}

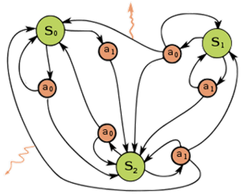


From planning to reinforcement learning

Fundamental Challenge 1

The dynamic programming approaches (VI and PI) as well as the linear programming approach all require the full knowledge of the transition model P and the reward.

$\Rightarrow$Need sampling approaches



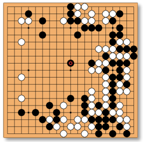
Fundamental Challenge 2

The computation and memory cost can be very expensive for large scale MDP problems.

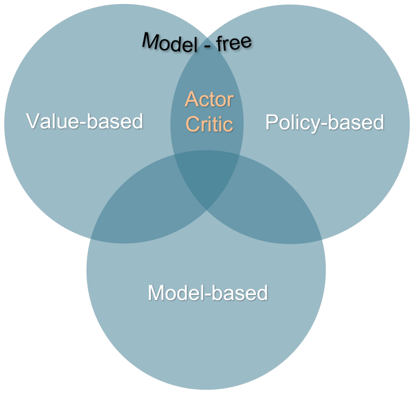
$\Rightarrow$Need new representations

Chess: 10^{120}

Go: 3^{361}



Overview of reinforcement learning approaches



- **Value-based RL**
  - Learn the optimal value functions $V^\star, Q^\star$
- **Policy-based RL**
  - Learn the optimal policy $\pi^\star$
- **Model-based RL**
  - Learn the model P, r and then do planning

Model-based vs model-free methods

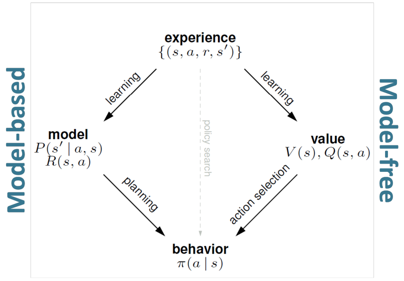


Figure: [8]

- Make full use of “experiences”
- Can reason about model uncertainty
- Sample efficient for easy dynamics

- Direct and simple
- Not affected by poor model estimation
- Not sample efficient

Online vs offline reinforcement learning

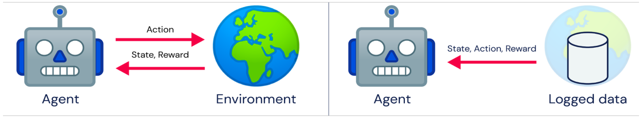


Figure: [4]

Online RL

- Collect data by interacting with environment
- Exploitation-exploration tradeoff

Offline/Batch RL

- Use previously collected data
- Data is static, no online data collection

On-policy vs. Off-policy reinforcement learning

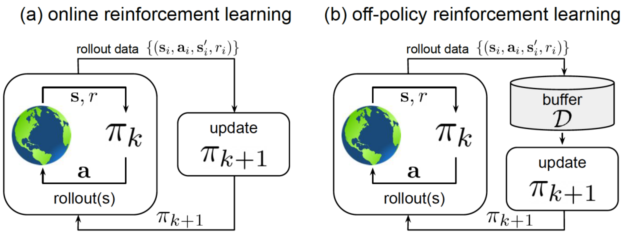

(a) online reinforcement learning
rollout data $\{(\mathbf{s}_i, \mathbf{a}_i, \mathbf{s}'_i, r_i)\}$
$\mathbf{s}, r$
π_k
$\mathbf{a}$
rollout(s)
update
π_{k+1}
π_{k+1}
(b) off-policy reinforcement learning
rollout data $\{(\mathbf{s}_i, \mathbf{a}_i, \mathbf{s}'_i, r_i)\}$
buffer
$\mathcal{D}$
$\mathbf{s}, r$
π_k
$\mathbf{a}$
rollout(s)
update
π_{k+1}
π_{k+1}

Figure: [12]

On-policy RL

- Learn based on data from current policy
- Always online

Off-policy RL

- Learn based on data from other policies
- Can be online or offline

Representation learning

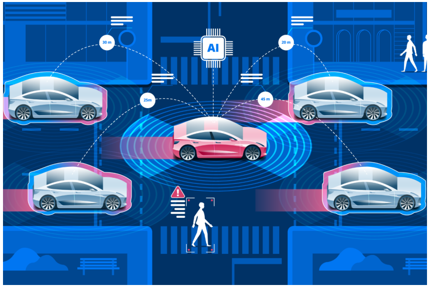


Figure: `http://selfdrivingcars.space/?p=68`

Large or continuous state and action spaces

$\Longrightarrow$

Function approximation

$$V(s) \approx V_\theta(s)$$
$$Q(s,a) \approx Q_\theta(s,a)$$
$$\pi(a|s) \approx \pi_\theta(a|s)$$
$$\mathsf{P}(s'|s,a) \approx \mathsf{P}_\theta(s'|s,a)$$

Value function representations

Linear function approximations

- The value function can be represented linearly using some known basis functions ϕ as follows:

$$V_\theta(s) = [\phi_1(s), \ldots, \phi_d(s)] \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_d \end{bmatrix}$$

- Reproducing kernel Hilbert space (RKHS) [18]
- Neural tangent kernel [7]

Nonlinear Function Approximation

- Fully connected neural networks [11]
- Convolutional neural networks [10]
- Residual networks [5]
- Recurrent networks [6]
- Self-attention [25]
- Generative adversarial networks [3]

Remarks: ◦ In continuous d dimensional state space linear function approx (LFA) is better then discretizing.

◦ Discretizing we would end up with a number of states which is exponential in d.

◦ Using features we will have algorithms that find an almost optimal policy in poly(d)-time.

The advantage of LFAs

- ◦ The improvement is possible because each state is treated independently in the tabular setting.
- ◦ Instead, with LFA we can exploit similarities between states.

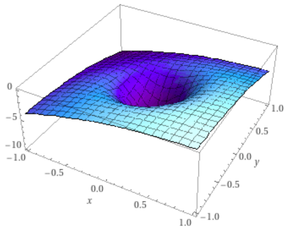


Figure: Reward function in a continuous grid world example

Example: ◦ Reward in the figure changes smoothly between neighboring states.

◦ After discretization, the states are independent.

◦ In contrast, LFA allows to exploit similarities.

Towards non-linear function approximations

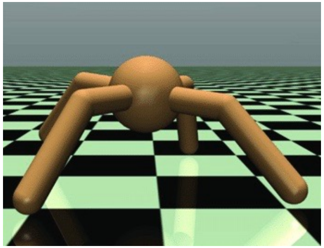


Figure: The ant environment in MuJoCo.

- In some important applications, like robotics, LFA is not expressive enough.
- For instance, in MuJoCo [23], we usually use a 3 layers neural network to parameterize the value function.
- The input is a vector containing position and velocity of the “ant” joints.

Our first goal: Model-free prediction

Goal:

Estimate $V^\pi(s)$ or $Q^\pi(s,a)$ from trajectories $\tau = \{s_0, a_0, r_0, s_0, \ldots\}$ collected with a given $\pi : \mathcal{S} \to \Delta(\mathcal{A})$:

$$V^\pi(s) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | s_0 = s, \pi\right], \qquad (V^\pi)$$

where the expectation is taken with respect to the randomness of the environment and the randomness of the actions due to π. We assume that the transition dynamics is not available.

Observations:
- ◦ Similar to the policy evaluation phase in Policy Iteration.
- ◦ But without knowledge of the dynamics!
- ◦ We will explain and analyze two methods:
  - ▸ Temporal Differences (TD).
  - ▸ Monte Carlo (MC).

Monte Carlo: the optimization problem

◦ Let us consider a state distribution $\rho \in \Delta_{\mathcal{S}}$.

◦ In MC, this is typically the initial state distribution.

Monte-Carlo optimization problem

We will use the following optimization problem to explain the main ideas in Monte-Carlo approaches:

$$\min_{V} \mathcal{L}(V) \tag{MC}$$

where $\mathcal{L}(V) = \frac{1}{2} \|V^{\pi} - V\|_{\rho}^{2}$ is a ρ-weighted norm loss.

Observations:

◦ The gradient is simply $\nabla \mathcal{L}(V) = \mathrm{diag}(\rho)(V - V^{\pi})$.

◦ The role of the distribution ρ will be made clearer in the next slides.

◦ We will apply stochastic gradient descent method to solve this problem.

Solving the problem via stochastic gradient descent (SGD)

∘ Ingredients of SGD algorithm:

- an unbiased, bounded estimate g_t of the gradient of the loss function, i.e., $\mathbb{E}g_t = \nabla\mathcal{L}(V)$;
- a step-size (i.e., learning rate) η_t;
- a simple iteration invariant: $V_{t+1} = V_t - \eta_t g_t$.

Solving the problem via stochastic gradient descent (SGD)

- Ingredients of SGD algorithm:
  - an unbiased, bounded estimate g_t of the gradient of the loss function, i.e., $\mathbb{E}g_t = \nabla\mathcal{L}(V)$;
  - a step-size (i.e., learning rate) η_t;
  - a simple iteration invariant: $V_{t+1} = V_t - \eta_t g_t$.
- Let us define a (random) vector $G^\pi \in \mathbb{R}^{|\mathcal{S}|}$ with entries $G^\pi(s_0) = \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)$ with $s_0 \sim \rho$, given π.
  - G^π is unbiased estimator of V^π. Indeed, by the definition of (V^π), we have $\mathbb{E}[G^\pi(s_0)] = V^\pi(s_0)$.
  - Recall the randomness of the expectation in (V^π).
  - Let $\mathbf{e}_s \in \{0,1\}^{|\mathcal{S}|}$ be the vector such that the s^{th} entry equals 1, and is zero, elsewhere.
  - For any V, $g_t = (V(s_0) - \mathbb{E}[G^\pi(s_0)])\mathbf{e}_{s_0}$ is an unbiased gradient estimator:

$$\mathbb{E}_{s_0\sim\rho}[(V(s_0) - \mathbb{E}[G^\pi(s_0)])\mathbf{e}_{s_0}] = \mathbb{E}_{s_0\sim\rho}[(V(s_0) - V^\pi(s_0))\mathbf{e}_{s_0}] = \text{diag}(\rho)(V - V^\pi) = \nabla\mathcal{L}(V).$$

  - Note that g_t has bounded second moment, $\mathbb{E}[\|g_t\|^2] \leq \frac{1}{(1-\gamma)^2}$:

$$\mathbb{E}[\|(G^\pi(s_0) - V^\pi(s_o))\mathbf{e}_{s_0}\|^2] = (G^\pi(s_0) - V^\pi(s_0))^2 \leq \frac{1}{(1-\gamma)^2}.$$

Solving the problem via stochastic gradient descent (SGD)

∘ Let us define a (random) vector $G^\pi \in \mathbb{R}^{|\mathcal{S}|}$ with entries $G^\pi(s_0) = \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)$ with $s_0 \sim \rho$, given π.

- G^π is unbiased estimator of V^π. Indeed, by the definition of (V^π), we have $\mathbb{E}[G^\pi(s_0)] = V^\pi(s_0)$.
- Recall the randomness of the expectation in (V^π).
- Let $\mathbf{e}_s \in \{0,1\}^{|\mathcal{S}|}$ be the vector such that the s^{th} entry equals 1, and is zero, elsewhere.
- For any V, $g_t = (V(s_0) - \mathbb{E}[G^\pi(s_0)])\mathbf{e}_{s_0}$ is an unbiased gradient estimator. Indeed,

$$\mathbb{E}_{s_0\sim\rho}[(V(s_0) - \mathbb{E}[G^\pi(s_0)])\mathbf{e}_{s_0}] = \mathbb{E}_{s_0\sim\rho}[(V(s_0) - V^\pi(s_0))\mathbf{e}_{s_0}] = \operatorname{diag}(\rho)(V - V^\pi) = \nabla\mathcal{L}(V).$$

- Note that g_t has bounded second moment, $\mathbb{E}[\|g_t\|^2] \leq \frac{1}{(1-\gamma)^2}$.

$$\mathbb{E}[\|g_t\|^2] = \mathbb{E}[\|(G^\pi(s_0) - V^\pi(s_0))\mathbf{e}_{s_0}\|^2] = (G^\pi(s_0) - V^\pi(s_0))^2 \leq \frac{1}{(1-\gamma)^2}.$$

Stochastic gradient descent applied to (MC)

We can use SGD updates as follows. Sample an initial state $s_0 \sim \rho$ and sample a trajectory $\{s_0, a_0, r_0, s_0, \ldots\}$, then

$$V_{t+1}(s_0) = V_t(s_0) - \eta_t(V_t(s_0) - G^\pi(s_0)),$$

where the step-size η_t has to be chosen appropriately.

SGD analysis via strong convexity

∘ Recall that g_t has bounded second moment, $\mathbb{E}[\|g_t\|^2] \leq \frac{1}{(1-\gamma)^2}$.

∘ Moreover, the loss function $\mathcal{L}(V)$ is $(\min_s \rho(s))$-strongly convex (recall the Hessian $\nabla^2 \mathcal{L}(V) = \mathrm{diag}(\rho)$).

- Below, we will assume that $(\min_s \rho(s)) > 0$.

Bound via SGD

Plugging into the SGD bound for strongly convex smooth functions (see Math of Data) with

- $L = 1$, $\sigma^2 = \frac{1}{(1-\gamma)^2}$, strong convexity $= \min_s \rho(s)$, and $\eta_t = \frac{1}{\mu t}$,

we can obtain the following guarantee for SGD:

$$\frac{1}{T} \sum_{t=1}^{T} \mathbb{E}[\|V^\pi - V_t\|_\rho^2] \leq \frac{2\,|\mathcal{S}| \log T}{T \min_s \rho(s)(1-\gamma)^2}. \tag{8}$$

By Jensen's inequality, the above guarantee holds true for the average iterate $\bar{V}_T = \frac{1}{T}\sum_{t=1}^{T} V_t$ as well.

Question*: ∘ What happens $(\min_s \rho(s)) = 0$? Which setting is better? This is a nuanced question!

Monte Carlo: The algorithm

Idea: ◦ Estimate $V^\pi(s)$ by the average of returns following all visits to s.

The Monte Carlo Algorithm [13, 22]

```
for $t = 1, \ldots, T$ do
  Collect an episode $\tau = \{s_0, a_0, r_0, s_1, \ldots, s_\mathbb{T}, a_\mathbb{T}, r_\mathbb{T}\}$ generated following $\pi$
  for each state $s_h$ do
    Compute return $G^\pi(s_t) = r_t + \gamma r_{t+1} + \cdots$
    Update $V_{t+1}(s_h) \leftarrow V_t(s_h) + \eta_t(G^\pi(s_h) - V_t(s_h))$
  end for
end for
```

Observations: ◦ This is **not** the SGD method with the step-size η_t (why?)

Monte Carlo: The algorithm

Idea: ○ Estimate $V^\pi(s)$ by the average of returns following all visits to s.

The Monte Carlo Algorithm [13, 22]

```
for $t = 1, \dots, T$ do
    Collect an episode $\tau = \{s_0, a_0, r_0, s_1, \dots, s_{\mathbb{T}}, a_{\mathbb{T}}, r_{\mathbb{T}}\}$ generated following $\pi$
    for each state $s_h$ do
        Compute return $G^\pi(s_t) = r_t + \gamma r_{t+1} + \cdots$
        Update $V_{t+1}(s_h) \leftarrow V_t(s_h) + \eta_t(G^\pi(s_h) - V_t(s_h))$
    end for
end for
```

Observations:

○ This is **not** the SGD method with the step-size η_t (why?) Due to the second loop!

○ Notice that in this implementation the updates G^π are correlated in the second loop.

○ The value estimates do not build on the other states even if they may be correlated.

○ Learning can be slow when the episodes are long.

○ The terminology of episode and trajectories are equivalent in the literature.

Monte Carlo with linear function approximation (LFA)

∘ We can parameterize the value function, i.e., $V_\theta = \Phi\theta$ with $\theta \in \mathbb{R}^d, \Phi \in \mathbb{R}^{|\mathcal{S}| \times d}$.

Monte Carlo optimization problem with LFA

For the case of linear function approximation, we look at the following optimization problem

$$\min_\theta \mathcal{L}(\theta) \tag{MC LFA}$$

with $\mathcal{L}(\theta) = \frac{1}{2}\|V^\pi - \Phi\theta\|_\rho^2$.

Observations: ∘ The gradient of the loss in $(\mathrm{MC\ LFA})$ is $\nabla\mathcal{L}(\theta) = \Phi^T \mathrm{diag}(\rho)(\Phi\theta - V^\pi)$.

∘ Similar to the SGD approach in (MC), we can have an unbiased estimator of the gradient.

SGD with linear function approximation

Let $s \sim \rho$ and let $\phi(s)$ denote the s^{th} row of Φ. Then, the SGD updates are as follows

$$\theta_{t+1} = \theta_t - \eta_t \phi(s)((\Phi\theta_t)(s) - G^\pi(s)),$$

where $\phi(s)((\Phi\theta_t)(s) - G^\pi(s))$ is an unbiased gradient estimate (why?) and η_t has to be chosen appropriately.

Monte Carlo analysis

◦ We can analyze MC with LFA via SGD.

◦ The loss function is smooth in the following sense.

Smoothness

If f is smooth then it holds that $\|\nabla f(x) - \nabla f(y)\| \leq L \|x - y\|$.

◦ By using the SGD convergence on Lipschitz smooth functions, we obtain the following guarantee.

SGD convergence bound with LFA

Let $\theta^\star$ be the minimizer of the loss function in (MC LFA) and let us assume realizability, i.e. $V_{\theta^\star} = V^\pi$. Let us, run Monte Carlo for T iterations with step size $\eta = \frac{\sqrt{d}(1-\gamma)}{\sqrt{t}}$, then it holds that

$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}[\|V_{\theta_t}(s) - V^\pi(s)\|_\rho^2] \leq \frac{2\sqrt{d}\log T}{(1-\gamma)\sqrt{T}}$$

By Jensen's inequality, the above guarantee holds true for the average iterate $\bar{V}_T = \frac{1}{T}\sum_{t=1}^{T} V_{\theta_t}$ as well.

Derivation: ◦ It follows from the descent lemma.

◦ The complete proof is in the appendix.

Monte Carlo cannot handle infinite horizon problems

- Unfortunately, we can make an update only after reaching the end of an episode.
  - Recall the definition $G^{\pi}(s_0) := \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)$.
  - All of our unbiased gradient estimators used this quantity.
- If the problem has an infinite horizon, we cannot compute G!
- We need to artificially introduce a finite horizon truncating the trajectories.
- The truncation introduces some bias making the above analysis invalid.
- These problems can be overcome by the temporal difference method that we look at next.

Temporal difference (TD) learning

- Recall LFA: We parameterize the value function, i.e., $V_\theta(s) = \phi(s)^T\theta$.
- Recall the Bellman operator: $(\mathcal{T}^\pi V)(s) = \sum_a \pi(a|s)\left(r(s,a) + \gamma \sum_{s'} \mathsf{P}(s'|s,a)V(s')\right)$.
- Recall that V^π is a fixed point of $\mathcal{T}^\pi$: i.e., $V^\pi = \mathcal{T}^\pi V^\pi$.

Optimization program for TD learning

The TD learning solves the following convex program

$$\min_{\theta\in\mathbb{R}^d} \mathcal{L}(\theta) \tag{TD}$$

where the objective $\mathcal{L}(\theta) = \frac{1}{2}\|V_\theta - \mathcal{T}^\pi V_\theta\|_\rho^2$ measures the violation of the fixed-point condition.

- In TD, we will consider ρ as the limit distribution below (in contrast to MC, where it is the initial distribution):

$$\rho(s) = \lim_{t\to\infty} \mathbb{P}_\pi[s_t = s].$$

- We can write the gradient of the loss function (TD) as

$$\nabla_\theta\mathcal{L}(\theta) = \Phi^T(I - \mathcal{T}^\pi)^T\mathrm{diag}(\rho)(I - \mathcal{T}^\pi)\Phi\theta. \tag{TD-Gradient}$$

Can we find an unbiased estimator, i.e. $\mathbb{E}[g_t] = \nabla_\theta \mathcal{L}(\theta)$?

- Challenge: We do not know the transition dynamics P in the model-free setting.
  - but we can sample from it!
- The TD-Gradient can be numerically computed via the following expression:

$$\nabla_\theta \mathcal{L}(\theta) = \sum_{s\in\mathcal{S}} \rho(s) \left(V_\theta(s) - \sum_{a\in\mathcal{A}} \pi(a|s)(r(s,a) + \gamma \mathbb{E}_{s'\sim \mathsf{P}(\cdot|s,a)}[V_\theta(s')]) \right) \cdot \left(\nabla_\theta V_\theta(s) - \gamma \mathbb{E}_{s'\sim \mathsf{P}(\cdot|s,a)}[\nabla_\theta V_\theta(s')] \right)$$

- So to approximate it, we can sample $S \sim \rho$, $A \sim \pi(\cdot|S)$ and $S' \sim \mathsf{P}(\cdot|S,A)$, and propose the estimator

$$g_t = \left(V_\theta(S) - r(S,A) - \gamma V_\theta(S') \right) \cdot \left(\nabla_\theta V_\theta(S) - \gamma \nabla_\theta V_\theta(S') \right)$$

- But this is clearly biased because the expectation does not distribute over products:

$$\mathbb{E}_{S'\sim \mathsf{P}(\cdot|S,A)}[V_\theta(S') \nabla_\theta V_\theta(S')] \neq \mathbb{E}_{S'\sim \mathsf{P}(\cdot|S,A)}[V_\theta(S')] \mathbb{E}_{S'\sim \mathsf{P}(\cdot|S,A)}[\nabla_\theta V_\theta(S')].$$

Avoiding the bias: the temporal difference (TD) method

∘ We will judiciously ignore the second term depending on θ, i.e.,

$$\Big(V_\theta(S) - r(S,A) - \gamma V_\theta(S')\Big) \cdot \Big(\nabla_\theta V_\theta(S) - \cancel{\gamma \nabla_\theta V_\theta(S')}\Big)$$

and we define

$$g_t = \Big(V_\theta(S) - r(S,A) - \gamma V_\theta(S')\Big) \cdot (\nabla_\theta V_\theta(S))$$

∘ This suggestion for g_t is an unbiased estimator for the following partial computation of TD-Gradient

$$F(\theta) = \sum_{s\in\mathcal{S}} \rho(s) \left(V_\theta(s) - \sum_{a\in\mathcal{A}} \pi(a|s)(r(s,a) + \gamma \mathbb{E}_{s'\sim \mathsf{P}(\cdot|s,a)}[V_\theta(s')]) \right) \cdot \nabla_\theta V_\theta(s)$$

∘ Is $F(\theta)$ a good update direction?

- Yes, it holds that $\langle F(\theta), \theta^\star - \theta\rangle \geq (1-\gamma)\left\|V_\theta - V_{\theta^\star}\right\|_\rho^2$, i.e., it is a *descent* direction (*cf.*, proof in [2]).
- This means that the expected direction of the update forms an acute angle with the vector pointing to $\theta^\star$.
- In addition, $F(\theta)$ satisfies $\|F(\theta)\|_2 \leq 2\left\|V_\theta - V_{\theta^\star}\right\|_\rho$ (*cf.*, proof in Lemma 4 of [2]).

Analysis of TD

Finite time bound for TD

Running TD with a step-size $\eta_t = \frac{1-\gamma}{4\sqrt{t}}$, we obtain

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[\|V_{\theta_t} - V_{\theta^\star}\|_\rho^2] \leq \frac{9d\log T}{2(1-\gamma)^2\sqrt{T}},$$

where $\theta^\star$ is the minimizer of the loss function in (TD).

Remarks:

- It is slightly worst than MC while being much easier to implement!
- The proof is in the appendix and simplifies the derivations in [2].

Temporal difference learning

Idea:

- ∘ Incrementally estimate $V^\pi(s)$ by the intermediate return plus estimated return at next state.
- ∘ In the linear case, $g_t = \big(V(s_t) - r_t - \gamma V(s_{t+1})\big)\nabla_\theta V_{\theta_t}(s_t)$ and the TD update is

$$\theta \leftarrow \theta - \eta_t g_t$$

TD Learning / TD(0)

```
for $t = 1, \ldots, T$ do
    for each step of an episode $\tau = \{s_0, a_0, r_0, s_1, \ldots, s_\mathbb{T}, a_\mathbb{T}, r_\mathbb{T}\}$ following $\pi$ do
        Compute update direction $g_t = \big(V(s_t) - r_t - \gamma V(s_{t+1})\big)\nabla_\theta V_{\theta_t}(s_t)$
        Update $\theta_{t+1} \leftarrow \theta_t - \eta_t g_t$
    end for
end for
```

Observations:

- ∘ Note that above implementation (and practitioners) do not sample from ρ!
  - ▶ [2] proves that you only suffer a small additive bias in the convergence guarantee.
- ∘ Similar to MC: learn directly from episodes of experiences without the MDP knowledge.
- ∘ Unlike MC: learn from incomplete episodes, and applicable to non-terminating environment.
- ∘ In the supplementary material we cover the generalization of TD which uses eligibility traces.

Convergence in the misspecified setting: Monte Carlo

∘ In the misspecified case, we have that $\min_\theta \|\Phi\theta - V^\pi\|_\rho = \epsilon_{\text{approx}} > 0$.

∘ If $\epsilon_{\text{approx}} = 0$ MC and TD converge to the same point.

∘ Otherwise, MC and TD converges to different points.

A property of the MC solution

For MC, the optimality condition $\nabla_\theta \mathcal{L}(\theta^\star_{\text{MC}}) = 0$ implies that

$$\Phi^T \text{diag}(\rho)\Phi\theta^\star_{\text{MC}} = \Phi^T \text{diag}(\rho) V^\pi,$$

which implies that $\theta^\star_{\text{MC}}$ is the projection of V^π in the feature span.

Derivation: ∘ From the stationarity of the solution, we have

$$\begin{aligned} &\nabla_\theta \mathcal{L}(\theta^\star_{\text{MC}}) = 0 \\ \implies &\Phi^T \text{diag}(\rho)(V_{\theta^\star_{\text{MC}}} - V^\pi) = 0 \\ \implies &\Phi^T \text{diag}(\rho)(\Phi\theta^\star_{\text{MC}} - V^\pi) = 0 \end{aligned}$$

Therefore, rearranging the terms gives $\Phi^T \text{diag}(\rho)\Phi\theta^\star_{\text{MC}} = \Phi^T \text{diag}(\rho) V^\pi$.

Convergence in the misspecified setting: TD

- For TD, we have that the stationary point $\theta^\star_{\mathrm{TD}}$ satisfies $F(\theta^\star_{\mathrm{TD}}) = 0$, i.e.,

$$\theta^\star_{\mathrm{TD}} = (\Phi^T \mathrm{diag}(\rho)\Phi)^{-1}\Phi^T \mathrm{diag}(\rho) T^\pi \Phi \theta^\star_{\mathrm{TD}}.$$

  - the derivation can be found in the next slide.
- Hence, $\theta^\star_{\mathrm{TD}}$ is not directly related to V^π but it is the fixed point of a projected Bellman equation.
- [24] has shown that

$$\left\| V_{\theta^\star_{\mathrm{TD}}} - V^\pi \right\|_\rho \leq \frac{1}{\sqrt{1-\gamma^2}} \|\Phi\theta^\star_{\mathrm{MC}} - V^\pi\|_\rho$$

Remarks:

- $\frac{1}{\sqrt{1-\gamma^2}}$ is an inflation factor that TD pays w.r.t. the minimum possible approximation error.
- The minimum possible approximation error is achieved by the MC method.

⋆Stationarity condition in TD

∘ In the previous slides, we used the fact

$$\theta^\star_{\mathrm{TD}} = (\Phi^T \mathrm{diag}(\rho)\Phi)^{-1}\Phi^T \mathrm{diag}(\rho) T^\pi \Phi\theta^\star_{\mathrm{TD}}.$$

Derivation: ∘ The stationarity condition $F(\theta^\star_{\mathrm{TD}}) = 0$ can be written in vector form, i.e.

$$\Phi^T \mathrm{diag}(\rho)(I - \mathcal{T}^\pi)\Phi\theta^\star_{\mathrm{TD}} = 0.$$

∘ Then, we just need the following steps:

$$\begin{aligned} &\Phi^T \mathrm{diag}(\rho)(\Phi\theta^\star_{\mathrm{TD}} - \mathcal{T}^\pi \Phi\theta^\star_{\mathrm{TD}}) = 0 \\ \Longrightarrow\ &\Phi^T \mathrm{diag}(\rho)\Phi\theta^\star_{\mathrm{TD}} = \Phi^T \mathrm{diag}(\rho)\mathcal{T}^\pi \Phi\theta^\star_{\mathrm{TD}} \\ \Longrightarrow\ &\theta^\star_{\mathrm{TD}} = (\Phi^T \mathrm{diag}(\rho)\Phi)^{-1}\Phi^T \mathrm{diag}(\rho)\mathcal{T}^\pi \Phi\theta^\star_{\mathrm{TD}} \end{aligned}$$

State-Action-Reward-State-Action (SARSA) for Q-value estimation

- In VI or PI, we often require the evaluation of Q-function to compute the greedy policy or optimal policy.
- How do we estimate Q^π?

SARSA optimization problem

The algorithm SARSA solves the following program:

$$\min_Q \mathcal{L}(Q) := \frac{1}{2} \|Q - \mathcal{T}^\pi Q\|_\rho^2 ,$$

where $\mathcal{T}^\pi$ is in this case the Bellman operator for Q value functions, that is,

$$(\mathcal{T}^\pi Q)(s,a) = r(s,a) + \gamma \sum_{s',a'} \mathsf{P}(s'|s,a)\pi(a'|s')Q(s',a').$$

Remarks:
- This is essentially the analog of the fixed point the TD loss but for the Q-function.
- We again use ρ as the stationary distribution (in contrast to initial distribution).

SARSA: the algorithm

◦ If we solve the SARSA program with SGD we obtain the following algorithm.

◦ SARSA should be understood as the TD method applied to the action-value function.

SARSA [19]

```
for $t = 1, \dots, T$ do
    for each step of an episode $\tau = \{s_0, a_0, r_0, s_1, \dots, s_{\mathbb{T}}, a_{\mathbb{T}}, r_{\mathbb{T}}\}$ following $\pi$ do
        Update $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha_t(r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$
    end for
end for
```

◦ With the same steps presented for the case of TD(0), one can prove convergence of SARSA.

Using model-free prediction for control

◦ We can look at MC and TD as approximate policy evaluation routines and use them in policy iteration.

Algorithm: Policy Iteration (PI) for solving MDPs

```
Start with an arbitrary policy guess $\pi_0$
for each iteration $t$ do
    (Step 1: Policy evaluation) Compute $V^{\pi_t}$:
        (Option 1) Iteratively apply policy value iteration, $V_t \leftarrow \mathcal{T}^{\pi_t} V_t$, until convergence. (Exact)
        (Option 2) Use the closed-form solution: $V^{\pi_t} = (I - \gamma P^{\pi_t})^{-1} R^{\pi_t}$. (Exact)
        (Option 3) Approximate $V^{\pi_t}$ with Monte Carlo. (Inexact)
        (Option 4) Approximate $V^{\pi_t}$ with Temporal Differences. (Inexact)
    (Step 2: Policy improvement) Update the current policy $\pi_t$ by the greedy policy
        $$\pi_{t+1}(s) = \arg\max_{a \in \mathcal{A}} \left[ r(s,a) + \gamma \sum_{s' \in \mathcal{S}} \mathrm{P}(s'|s,a) V^{\pi_t}(s') \right]. \quad (9)$$
end for
```

Using model-free prediction for control

◦ The advantage of Options 3 and 4 is that they do not require knowledge of P.

◦ The disadvantage is that the value functions computed in these ways are *inexact*.

How the errors propagate in the PI analysis?

This question is answered by [17] that gives the bound

$$\sum_{s\in\mathcal{S}} \rho(s)(V^\star(s) - V^{\pi^k}(s)) \leq \mathcal{O}\left(\frac{1}{(1-\gamma)^2} \max_{1\leq j\leq k} \underbrace{\left\| V_{\theta_j} - V^{\pi_j} \right\|_\rho}_{\text{evaluation error}} \right) + \mathcal{O}(\gamma^k).$$

Remarks:

◦ The evaluation error can be controlled with the results derived before for MC or DP.

◦ The PI with these inexact options is also known as Least Square Policy Iteration [9].

Control while learning: Q-Learning

- We can use the same idea to estimate directly the optimal action-value function.

Q-Learning [26]

```
for $t = 1, \dots, T$ do
    for each step of an episode $\tau = \{s_0, a_0, r_0, s_1, \dots, s_{\mathbb{T}}, a_{\mathbb{T}}, r_{\mathbb{T}}\}$ following $\pi$ do
        Update $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha_t(r_t + \gamma \max_{b \in \mathcal{A}} Q(s_{t+1}, b) - Q(s_t, a_t))$
    end for
end for
```

- In Q-Learning, learning and control phases are not clearly separated.
- Running the notebooks, you will see that the choice of the policy π makes a big difference in practice.
- Actually also in theory but proving it is a bit too advanced for now.

Summary: Model-free prediction

Methods	**DP**	**MC**	**TD(0)**
Model knowledge	Need	No need	No need
Uses Bellman equation?	Yes	No	Yes
When to perform updates	After next step	After whole episode	After next step
Bias	-	Unbiased	Biased
Variance	-	Big	Small

Reference: [21]

Wrap Up

- PI and VI are dynamic programming methods applicable when the transition matrix is known.
- Monte Carlo methods are used to estimate value function when the transition matrix is unknown.
- Monte Carlo methods are an instance of stochastic approximation.
- TD is an application of dynamic programming when the transition matrix is unknown.
- The following week is about Linear Programming for RL!
- Next week is Jupiter Notebook $\#1$.

References I

[1] Alekh Agarwal, Nan Jiang, Sham M Kakade, and Wen Sun.
Reinforcement learning: Theory and algorithms.
CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep, 2019.
22

[2] Jalaj Bhandari, Daniel Russo, and Raghav Singal.
A finite time analysis of temporal difference learning with linear function approximation.
In *Conference On Learning Theory*, pages 1691–1692. PMLR, 2018.
70, 71, 72

[3] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio.
Generative Adversarial Networks.
ArXiv e-prints, June 2014.
54

[4] Caglar Gulcehre, Sergio Gómez Colmenarejo, Jakub Sygnowski, Thomas Paine, Konrad Zolna, Yutian Chen, Matthew Hoffman, Razvan Pascanu, Nando de Freitas, et al.
Addressing extrapolation error in deep offline reinforcement learning.
2020.
51

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun.
Deep residual learning for image recognition.
pages 770–778, 2016.
54

References II

[6] Sepp Hochreiter and Jürgen Schmidhuber.
Long short-term memory.
Neural computation, 9(8):1735–1780, 1997.
54

[7] Arthur Jacot, Franck Gabriel, and Clément Hongler.
Neural tangent kernel: Convergence and generalization in neural networks.
In *Advances in neural information processing systems*, pages 8571–8580, 2018.
54

[8] Mykel J. Kochenderfer, Tim A. Wheeler, and Kyle H. Wray.
Algorithms for Decision Making.
MIT press, 2022.
50, 97, 98, 99

[9] Michail G Lagoudakis and Ronald Parr.
Least-squares policy iteration.
The Journal of Machine Learning Research, 4:1107–1149, 2003.
79

[10] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner.
Gradient-based learning applied to document recognition.
Proceedings of the IEEE, 86(11):2278–2324, Nov 1998.
54

References III

[11] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton.
Deep learning.
Nature, 521(7553):436–444, 2015.
54

[12] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu.
Offline reinforcement learning: Tutorial, review, and perspectives on open problems.
arXiv preprint arXiv:2005.01643, 2020.
52

[13] Donald Michie and Roger A Chambers.
Boxes: An experiment in adaptive control.
Machine intelligence, 2(2):137–152, 1968.
63, 64

[14] Martin L Puterman.
Markov decision processes: discrete stochastic dynamic programming.
John Wiley & Sons, 2014.
22

[15] Satinder Singh Richard and Richard C. Yee.
An upper bound on the loss from approximate optimal-value functions.
In *Machine Learning*, pages 227–233, 1994.
41

References IV

[16] Bruno Scherrer.
Improved and generalized upper bounds on the complexity of policy iteration.
Mathematics of Operations Research, 41(3):758–774, 2016.
41

[17] Bruno Scherrer, Victor Gabillon, Mohammad Ghavamzadeh, and Matthieu Geist.
Approximate modified policy iteration.
arXiv preprint arXiv:1205.3054, 2012.
41, 79

[18] Bernhard Schölkopf and Alexander J. Smola.
Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.
MIT Press, Cambridge, MA, USA, 2001.
54

[19] Satinder Singh, Tommi Jaakkola, Michael L Littman, and Csaba Szepesvári.
Convergence results for single-step on-policy reinforcement-learning algorithms.
Machine learning, 38(3):287–308, 2000.
77

[20] Richard S. Sutton and A. G. Barto.
Reinforcement Learning: An Introduction.
MIT Press, Cambridge, MA, 1998.
104, 109

References V

[21] Richard S Sutton and Andrew G Barto.
Reinforcement learning: An introduction.
MIT press, 2018.
81, 107, 110

[22] Richard S Sutton, Andrew G Barto, et al.
Introduction to reinforcement learning, volume 135.
MIT press Cambridge, 1998.
63, 64

[23] Emanuel Todorov, Tom Erez, and Yuval Tassa.
Mujoco: A physics engine for model-based control.
In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5026–5033, 2012.
56

[24] John Tsitsiklis and Benjamin Van Roy.
Analysis of temporal-diffference learning with function approximation.
Advances in neural information processing systems, 9, 1996.
74

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin.
Attention is all you need.
2017.
54

References VI

[26] Christopher JCH Watkins and Peter Dayan.
Q-learning.
Machine learning, 8(3-4):279–292, 1992.
80

Supplementary material

Existence of an optimal policy (proof)

Proof Sketch

Assume a start from $(s_0, a_0, r_0, s_1) = (s, a, r, s')$, *then*

1. *Define "offset" policy* $\tilde{\pi}(a_t = a \,|\, h_t) := \pi(a_{t+1} = a \,|\, (s_0, a_0) = (s, a), h_t)$, *Markov property implies*

$$\mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^t r(s_t, a_t) \,|\, (s_0, a_0, r_0, s_1) = (s, a, r, s'),\ \pi\right] = \gamma V^{\tilde{\pi}}(s').$$

2. *With all* $(s_0, a_0, r_0) = (s, a, r)$, *the set* $\{\tilde{\pi} \,|\, \Pi\}$ *will just be* Π *itself.*
3. *Show that the optimal value from* s_1 *onward is independent of* $(s_0, a_0, r_0) = (s, a, r)$,

$$\max_{\pi \in \Pi} \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^t r(s_t, a_t) \,|\, (s_0, a_0, r_0, s_1) = (s, a, r, s'),\ \pi\right] = \gamma \max_{\pi \in \Pi} V^{\tilde{\pi}}(s') = \gamma \max_{\pi \in \Pi} V^{\pi}(s') = \gamma V^{\star}(s').$$

Existence of an optimal policy (proof)

Proof Sketch (cont.)

4. *Let* $\pi(s) = \arg\max_{a\in\mathcal{A}} \max_{\pi'\in\Pi} Q^{\pi'}(s,a)$, *show this deterministic and randomized policy is optimal*

$$
\begin{aligned}
V^\star(s_0) &= \max_{\pi'\in\Pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t,a_t) \mid s_0 = s,\ \pi\right] = \max_{\pi'\in\Pi} \mathbb{E}\left[r(s_0,a_0) + \sum_{t=1}^{\infty} \gamma^t r(s_t,a_t) \mid \pi\right] \\
&= \max_{\pi'\in\Pi} \mathbb{E}\left[r(s_0,a_0) + \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^t r(s_t,a_t) \mid (s_0,a_0,r_0,s_1), \pi\right]\right] \\
&\leq \max_{\pi'\in\Pi} \mathbb{E}\left[r(s_0,a_0) + V^\star(s_1)\right] \quad \Longleftarrow \quad \textit{Step 3 above} \\
&= \mathbb{E}\left[r(s_0,a_0) + V^\star(s_1) \mid \pi\right] \quad \Longleftarrow \quad \textit{Definition of } \pi \textit{ above}
\end{aligned}
\tag{10}
$$

5. $V^\star(s_0) \leq \mathbb{E}[r(s_0,a_0) + V^\star(s_1) \mid \pi] \leq \mathbb{E}\left[r(s_0,a_0) + \gamma r(s_1,a_1) + \gamma^2 V^\star(s_1) \mid \pi\right] \leq \cdots \leq V^\pi(s_0)$, *so* $\mathbf{V}^\pi = \mathbf{V}^\star$, *i.e., the proposed* π *is optimal.*

$\square$

Contraction of bellman optimality operator (proof)

Proof.

For any $\mathbf{V}', \mathbf{V} \in \mathbb{R}^{|\mathcal{S}|}$ and $s \in \mathcal{S}$, we have

$$\begin{aligned}
&\left|\left(\mathcal{T}\mathbf{V}'\right)(s) - (\mathcal{T}\mathbf{V})(s)\right| \\
&= \left|\max_{a\in\mathcal{A}} \left[r(s,a) + \gamma \sum\nolimits_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a)\mathbf{V}'(s')\right] - \max_{a'\in\mathcal{A}} \left[r(s,a') + \gamma \sum\nolimits_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a')\mathbf{V}(s')\right]\right| \\
&\leq \max_{a\in\mathcal{A}} \left|\left(r(s,a) + \gamma \sum\nolimits_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a)\mathbf{V}'(s')\right) - \left(r(s,a) + \gamma \sum\nolimits_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a)\mathbf{V}(s')\right)\right| \\
&\leq \max_{a\in\mathcal{A}} \gamma \sum\nolimits_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a)\left|\mathbf{V}'(s') - \mathbf{V}(s')\right| \\
&\leq \left\|\mathbf{V}' - \mathbf{V}\right\|_\infty \max_{a\in\mathcal{A}} \gamma \sum\nolimits_{s'\in\mathcal{S}} \mathsf{P}(s'|s,a) = \gamma \left\|\mathbf{V}' - \mathbf{V}\right\|_\infty,
\end{aligned}$$

which concludes the proof. □

Policy improvement theorem (proof)

Theorem (Policy Improvement)

If a (deterministic) policy π' satisfies that,

$$Q^\pi(s, \pi'(s)) \geq V^\pi(s) \quad \forall\ s \in \mathcal{S}, \tag{11}$$

then $V^{\pi'}(s) \geq V^\pi(s)$ for any $s \in \mathcal{S}$.

Proof.

Follow the property, for any $s \in \mathcal{S}$, (denote $s' \sim \mathsf{P}(\cdot|s, \pi'(s))$ as $s' \sim \pi'$)

$$\begin{aligned} V^\pi(s) \leq\ Q^\pi(s, \pi'(s)) &= \mathbb{E}_{\pi'}\left[r(s_0, \pi'(s_0)) + \gamma V^\pi(s_1) |\, s_0 = s\right] \\ &\leq \mathbb{E}_{\pi'}\left[r_0 + \gamma Q^\pi(s_1, \pi'(s_1)) \,|\, s_0 = s\right] \\ &\leq \mathbb{E}_{\pi'}[r_0 + \gamma r_1 + \gamma V^\pi(s_1) \,|\, s_0 = s] \\ &\leq \cdots \\ &\leq \mathbb{E}_{\pi'}\left[r_0 + \gamma r_1 + \gamma^2 r_2 + \cdots \,|\, s_0 = s\right] = V^{\pi'}(s). \end{aligned} \tag{12}$$

□

POMDPs

Partial observable Markov decision processes (POMDPs)

- $\mathcal{S}$ is the set of all possible states
- $\mathcal{A}$ is the set of all possible actions
- $\mathsf{P}(s'|s,a)$: $\mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is the transition model
- Ω is the set of observations: $o \in \Omega$.
- O is a set of conditional observation probabilities: $\mathsf{O}(o|s',a)$.
- $r(s,a)$: $\mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function
- μ is the initial state distribution: $s_0 \sim \mu \in \Delta(\mathcal{S})$
- γ is the discount factor: $\gamma \in [0,1]$

MDP vs POMDP:

- POMDPs are flexible: *We do not have to have perfect information about the states.*
- POMDPs are closer to the real world.

  Example: see a baby crying but do not know the true state (hungry, sleepy, etc).
- MDPs assume perfect knowledge of the states.

POMDPs

∘ When we do not observe the actual states, we construct the so-called *belief states* vector.

Definition (Belief states)

A belief state vector b_t is a distribution over states at time t that estimates the state distribution given the observation and the action history $h_t = \{o_0, a_0, \ldots, a_{t-1}, o_t\}$, i.e., $P(s_t = s|h_t)$:

$$b_t(s) := P(s_t = s|h_t).$$

Remarks:

∘ Via the Bayes rule, the belief states must satisfy:

$$\begin{aligned}\mathrm{P}(s_t = s|h_t) =& \frac{\mathrm{O}(o_t|s_t, a_{t-1}, h_{t-1})\mathrm{P}(s_t|a_{t-1}, h_{t-1})}{\mathrm{P}(o_t|a_{t-1}, h_{t-1})} \\ =& \frac{\mathrm{O}(o_t|s_t, a_{t-1}, h_{t-1}) \sum_{s_{t-1}} \mathrm{P}(s_t|s_{t-1}, a_{t-1})\mathrm{P}(s_{t-1}|h_{t-1})}{\sum_{s_t} \mathrm{O}(o_t|s_t, a_{t-1}, h_{t-1}) \sum_{s_{t-1}} \mathrm{P}(s_t|s_{t-1}, a_{t-1})\mathrm{P}(s_{t-1}|h_{t-1})}.\end{aligned}$$

∘ As a result, we have a recursion for the conditional probability $\mathrm{P}(s_t = s|h_t)$.

∘ We will represent this recursion via a "belief operator."

The belief operator

◦ We can concisely represent the recursion on $b_t(s)$ using the belief operator $U : \Delta(\mathcal{S}) \times \Omega \times \mathcal{A} \to \Delta(\mathcal{S})$:

$$b_{t+1}(s') = U(b_t; a, o)(s') = \frac{\mathsf{O}(o|s', a) \sum_{s \in S} \mathsf{P}(s'|s, a) b_t(s)}{\sum_{s'} \mathsf{O}(o|s', a) \sum_{s \in S} \mathsf{P}(s'|s, a) b_t(s)}.$$

Remarks:

◦ The expected (non-stationary) **reward** now also depends on our current belief state:

$$r_t(a) = \sum_{s \in S} r(a, s) b_t(s).$$

◦ We will focus more on MDPs and how to solve them optimally.

◦ Tools for MDPs translate readily to POMDPs once we have an estimate of $b_t(s)$.

Numerical example: Hex World

- Traverse a tile map to reach a goal state
- Each cell in the tile map represents a state; action is a move in any of the 6 directions
- Taking any action in certain cells gives a specified reward and transports to a **terminal state**

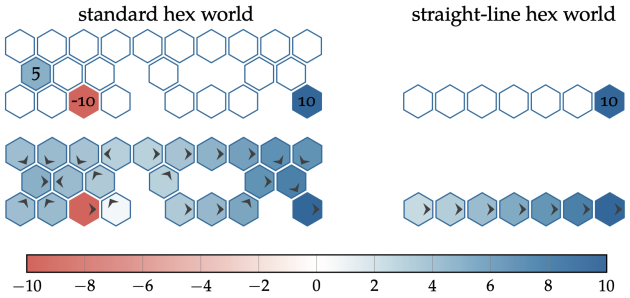


Figure: Top row shows the base problem setup and colors hexes with terminal rewards. Bottom row shows an optimal policy for each problem and colors the expected value. Arrows indicate the action to take in each state. [8]

Numerical example: Value iteration

- Initialized with the **east-moving** policy

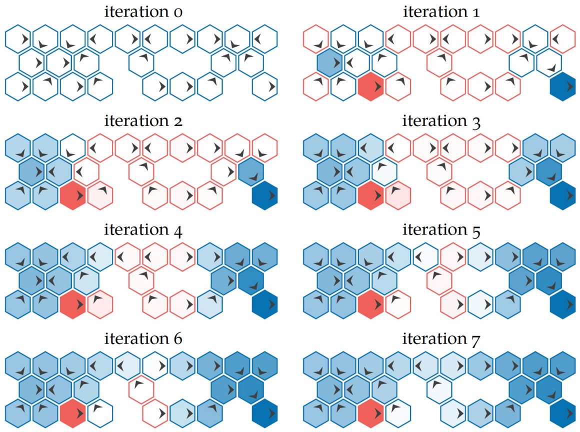


Figure: Value iteration for Hex World. [8]

Numerical example: Policy iteration

- Initialized with the **east-moving** policy
- An optimal policy is obtained (the algorithm converges) in four iterations

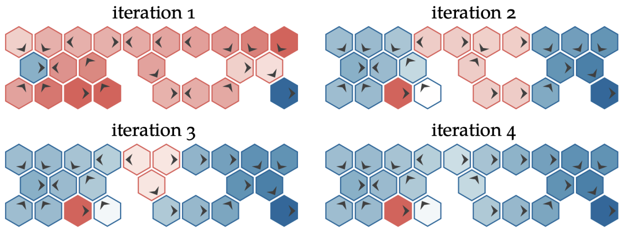


Figure: Policy iteration for Hex World. [8]

Proof for the Monte Carlo case.

Proof.

$$\begin{aligned}\|\theta_{t+1}-\theta^\star\|^2 &= \|\theta_t-\theta^\star\|^2 - \eta(\theta_t-\theta^\star)^T\phi(s)(V_{\theta_t}(s)-G(s)) + \eta^2\|\phi(s)(V_{\theta_t}(s)-G(s))\|^2\\ &= \|\theta_t-\theta^\star\|^2 - \eta(V_{\theta_t}(s)-V_{\theta^\star}(s))\cdot(V_{\theta_t}(s)-G(s)) + \eta^2\|\phi(s)(V_{\theta_t}(s)-G(s))\|^2\end{aligned}$$

Then, taking expectation and using that $\|\phi(s)(V_{\theta_t}(s)-G(s))\|^2 \leq \frac{1}{(1-\gamma)^2}$,

$$\begin{aligned}\mathbb{E}[\|\theta_{t+1}-\theta^\star\|^2] &\leq \mathbb{E}[\|\theta_t-\theta^\star\|^2] - \eta\mathbb{E}[(V_{\theta_t}(s)-V_{\theta^\star}(s))\cdot(V_{\theta_t}(s)-\mathbb{E}[G(s)|s])] + \frac{\eta^2}{(1-\gamma)^2}\\ &\leq \mathbb{E}[\|\theta_t-\theta^\star\|^2] - \eta\mathbb{E}_{s\sim\rho}\mathbb{E}[(V_{\theta_t}(s)-V^\pi(s))^2] + \frac{\eta^2}{(1-\gamma)^2}\\ &\leq \mathbb{E}[\|\theta_t-\theta^\star\|^2] - \eta\mathbb{E}[\|V_{\theta_t}(s)-V^\pi(s)\|_\rho^2] + \frac{\eta^2}{(1-\gamma)^2}\end{aligned}$$

Then, rearranging and dividing by η.

$$\mathbb{E}[\|V_{\theta_t}(s)-V^\pi(s)\|_\rho^2] \leq \frac{\mathbb{E}[\|\theta_t-\theta^\star\|^2]-\mathbb{E}[\|\theta_{t+1}-\theta^\star\|^2]}{\eta} + \frac{\eta}{(1-\gamma)^2}$$

□

Proof for the Monte Carlo case (continued).

Proof.

Summing over $t \in [T]$, we obtain

$$\begin{aligned}\sum_{t=1}^{T} \mathbb{E}[\|V_{\theta_t}(s) - V^{\pi}(s)\|_{\rho}^2] &\leq \frac{\mathbb{E}[\|\theta_1 - \theta^{\star}\|^2]}{\eta} + \frac{\eta}{(1-\gamma)^2}T \\ &\leq \frac{d}{\eta} + \frac{\eta}{(1-\gamma)^2}T\end{aligned}$$

Therefore, choosing $\eta = \frac{\sqrt{d}(1-\gamma)}{\sqrt{T}}$ and dividing by T.

$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}[\|V_{\theta_t}(s) - V^{\pi}(s)\|_{\rho}^2] \leq \frac{2\sqrt{d}}{(1-\gamma)\sqrt{T}}$$

$\square$

Proof of Finite Time Bound for TD

Proof.

- For the analysis we restart from the same step we did for the case without variance but we take expectation.
- Recall that g_t equals $F(\theta)$ plus zero mean noise, i.e. $\mathbb{E}[g_t] = F(\theta)$.
- And it holds $\mathbb{E}\,\|g_t - F(\theta)\| \leq \sigma$.

$$
\begin{aligned}
\mathbb{E}[\|\theta_{t+1} - \theta^\star\|^2] &= \mathbb{E}[\|\theta_t - \theta^\star\|^2] - 2\mathbb{E}[\eta g_t^T(\theta_t - \theta^\star)] + \eta^2\mathbb{E}[\|g_t\|^2] \\
&\leq \mathbb{E}[\|\theta_t - \theta^\star\|^2] - 2\eta(1-\gamma)\mathbb{E}[\|V_{\theta_t} - V_{\theta^\star}\|_\rho^2] + 4\eta^2\mathbb{E}[\|V_{\theta_t} - V_{\theta^\star}\|_\rho^2] + \eta^2\mathbb{E}[\|g_t - F(\theta_t)\|^2] \\
&\leq \mathbb{E}[\|\theta_t - \theta^\star\|^2] - 2\eta(1-\gamma)\mathbb{E}[\|V_{\theta_t} - V_{\theta^\star}\|_\rho^2] + 4\eta^2\mathbb{E}[\|V_{\theta_t} - V_{\theta^\star}\|_\rho^2] + \eta^2\sigma^2 \\
&\leq (1 - \frac{(1-\gamma)^2\lambda_{\min}^2}{4})\mathbb{E}[\|\theta_t - \theta^\star\|^2] + \frac{\lambda_{\min}^2}{4} \\
&\leq \left(1 - \frac{(1-\gamma)^2\lambda_{\min}^2}{4}\right)^t \mathbb{E}[\|\theta_1 - \theta^\star\|^2] + \frac{\lambda_{\min}^2}{4}\sum_{\ell=0}^{\infty}\left(1 - \frac{(1-\gamma)^2\lambda_{\min}^2}{4}\right)^\ell \\
&\leq \left(1 - \frac{(1-\gamma)^2\lambda_{\min}^2}{4}\right)^t \mathbb{E}[\|\theta_1 - \theta^\star\|^2] + \frac{1}{(1-\gamma)^2}
\end{aligned}
$$

□

Proof of $\lambda_{\min}$ independent Finite Time bound.

Proof.

◦ Restarting from

$$\mathbb{E}[\|\theta_{t+1} - \theta^\star\|^2] \leq \mathbb{E}[\|\theta_t - \theta^\star\|^2] - 2\eta(1-\gamma)\mathbb{E}[\|V_{\theta_t} - V_{\theta^\star}\|_\rho^2] + 4\eta^2\mathbb{E}[\|V_{\theta_t} - V_{\theta^\star}\|_\rho^2] + \eta^2\sigma^2$$

◦ We set $\eta \leq \frac{1-\gamma}{4}$ and we can rearrange the term as follows

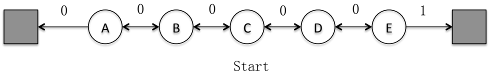
$$\mathbb{E}[\|V_{\theta_t} - V_{\theta^\star}\|_\rho^2] \leq \frac{1}{\eta(1-\gamma)}\left(\mathbb{E}[\|\theta_t - \theta^\star\|^2] - \mathbb{E}[\|\theta_{t+1} - \theta^\star\|^2]\right) + \eta\frac{\sigma^2}{1-\gamma}$$

◦ Finally averaging over $t = 1, \dots, T$, we obtain $\frac{1}{T}\sum_{t=1}^T \mathbb{E}[\|V_{\theta_t} - V_{\theta^\star}\|_\rho^2] \leq \frac{1}{\eta(1-\gamma)}\mathbb{E}[\|\theta_1 - \theta^\star\|^2] + \eta\frac{\sigma^2}{1-\gamma}$ ◦ Setting $\eta = \frac{1-\gamma}{4\sqrt{T}}$, we obtain

$$\frac{1}{T}\sum_{t=1}^T \mathbb{E}[\|V_{\theta_t} - V_{\theta^\star}\|_\rho^2] \leq \frac{4}{\sqrt{T}}\mathbb{E}[\|\theta_1 - \theta^\star\|^2] + \frac{1}{2(1-\gamma)^2\sqrt{T}} \leq \frac{9d}{2(1-\gamma)^2\sqrt{T}}$$

□

Numerical example: Random walk



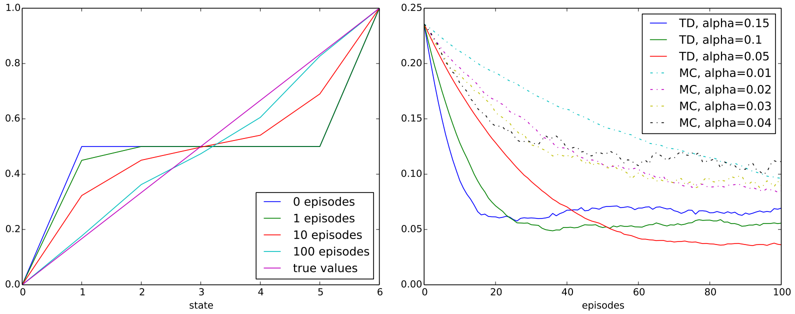


Figure: Left: values learned after various number of updates in a single run of TD(0). Right: the root mean-squared (RMS) error between the value functions learned and the true values. [20]

Bias-variance trade-off

- MC return is **unbiased**, but has **higher variance** since it relies on many random steps
- TD target is **biased**, but has **lower variance** since it only relies on the next step
- The MC error can be written as a sum of TD errors:

$$\begin{aligned}
G_t - V(s_t) &= r_{t+1} + \gamma G_{t+1} - V(s_t) + \gamma V(s_{t+1}) - \gamma V(s_{t+1}) \\
&= \delta_t + \gamma(G_{t+1} - V(s_{t+1})) \\
&= \delta_t + \gamma\delta_{t+1} + \gamma^2(G_{t+2} - V(s_{t+2})) \\
&= \cdots \\
&= \sum_{k=t}^{\infty} \gamma^{k-t}\delta_k
\end{aligned}$$

Multiple-step TD learning

Definition (n-step return)

Let T be the termination time step in a given episode, $\gamma \in [0,1]$.

$$
\begin{aligned}
G_t^{(1)} &= r_{t+1} + \gamma V(s_{t+1}) && \textit{TD(0)} \\
G_t^{(2)} &= r_{t+1} + \gamma r_{t+2} + \gamma^2 V(s_{t+2}) && \textit{(two-step return)} \\
G_t^{(n)} &= r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{n-1} r_{t+n} + \gamma^n V(s_{t+n}) && \textit{(n-step return)} \\
G_t^{(\infty)} &= r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{T-t-1} r_T && \textit{MC}
\end{aligned}
$$

Note that $G_t^{(n)} = G_t^{(\infty)}$ if $t + n \geq T$.

Multiple-step TD learning

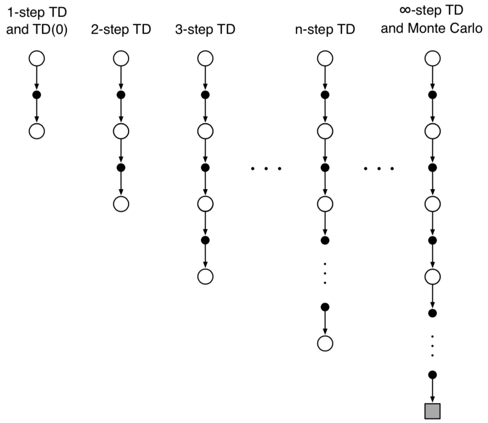


Figure: [21]

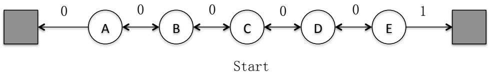
Multiple-step TD learning

Multi-step TD learning:

$$V(s_t) \leftarrow V(s_t) + \alpha_t \Big(\underbrace{G_t^{(n)} - V(s_t)}_{n\text{-step TD error}} \Big)$$

Observations:
- Unifies and combines TD(0) and MC: $n = 1$ recovers TD(0) and $n = \infty$ recovers MC.
- Trades-off bias and variance.
- However, we need to observe $r_{t+1}, \cdots, r_{t+n}$.

Numerical example: Longer random walk



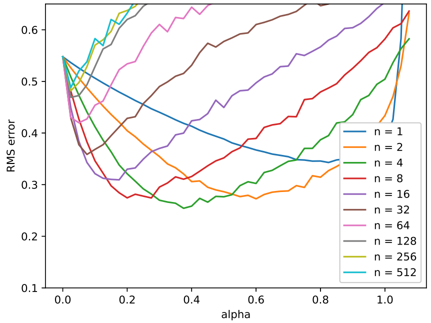


Figure: Performance of n-step TD methods as a function of α, for various values of n, on a 19-state random walk task. [20]

Further extension: TD(λ) with eligibility trace

λ-return (weighted average of all n-step returns)

$$G_t^{\lambda} = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)}$$

TD(λ)

$$V(s_t) \leftarrow V(s_t) + \alpha \left[G_t^{\lambda} - V(s_t) \right]$$

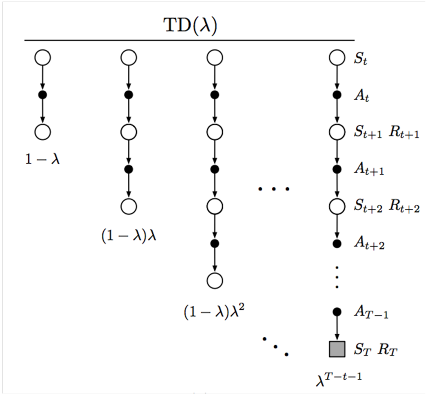


Figure: [21]

Further extension: TD(λ) with eligibility trace

TD(λ)

$$V(s_t) \leftarrow V(s_t) + \alpha \left[G_t^{\lambda} - V(s_t)\right]$$

Observations:

- $\lambda = 0$ reduces to TD(0); $\lambda = 1$ reduces to MC.
- Can be efficiently implemented:

$$\begin{aligned} V(s) &\leftarrow V(s) + \alpha \delta_t e_t(s) \\ e_t(s) &= \gamma \lambda e_{t-1}(s) + \mathbb{1}\{s_t = s\} \end{aligned}$$

- The term $e_t(s) = \sum_{k=0}^{t} \gamma^{t-k} \mathbb{1}\{s_t = s\}$ is called the eligibility trace.
- Converge faster than TD(0) when λ is appropriately chosen.