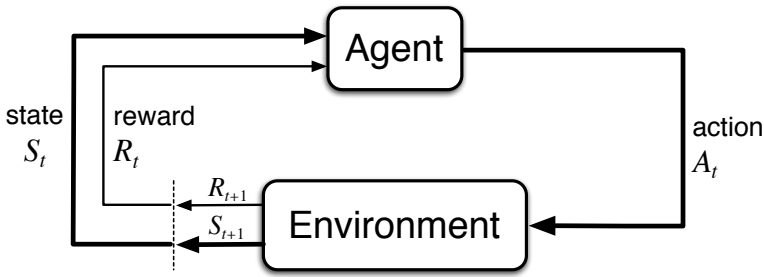
# Reinforcement Learning

# Reinforcement learning and applications

# What is reinforcement learning?

An agent learns how to act from the consequences of its decisions.




At step $t$: observe $S_t$, choose $A_t$, then receive the reward $R_{t+1}$ and the next state $S_{t+1}$.

The goal is to obtain a large **cumulative reward** over repeated interaction.

Original diagram: Sutton and Barto, Reinforcement Learning: An Introduction, Fig. 3.1 (second-edition draft).

# Games: choosing moves in Go




Choose moves that improve the chance of eventually winning.

Case: AlphaGo, Google DeepMind. Match photo: Google, AlphaGo challenge match.

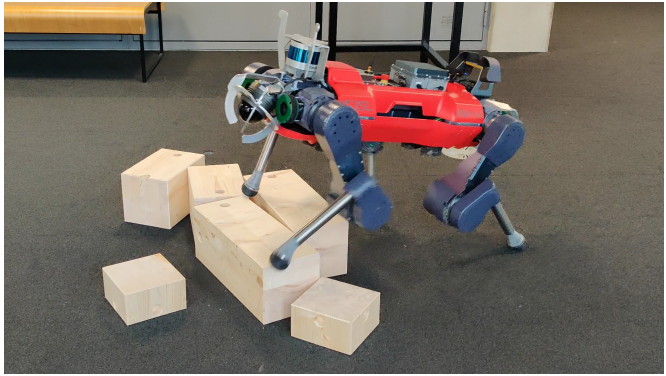
# Video games: learning actions from pixels



Observe the screen, choose controls, and learn to increase the game score.

Original Space Invaders screenshots extracted from Mnih et al. (2015), Fig. 4. Human-level control through deep reinforcement learning. Game: Taito.

# Robotics: learning to walk on varied terrain



Learn locomotion in simulation, then deploy the policy on a real robot.

Photo and case: Rudin et al., Learning to Walk in Minutes Using Massively Parallel Deep Reinforcement Learning.

# Infinite-horizon discounted MDPs

# The interaction at time $t$

The agent observes $S_t$, chooses $A_t$, then receives $R_{t+1}$ and observes $S_{t+1}$.

$$(S_0, A_0, R_1, S_1, A_1, R_2, S_2, \ldots).$$

The environment supplies a joint conditional law of $(R_{t+1}, S_{t+1})$:

$$\mathcal{L}(R_{t+1}, S_{t+1} \mid \text{history}, A_t) = \mathcal{L}(R_{t+1}, S_{t+1} \mid S_t, A_t).$$

The **Markov property** concerns the environment given the current state and action. It does not make successive states independent.

Puterman, Sec. 2.1. Sutton and Barto, Sec. 3.1.

# A finite discounted MDP

We use the model

$$(\mathcal{S}, \mathcal{A}, P, r, \gamma).$$

| | |
|---|---|
| $\mathcal{S}$ | Finite state space |
| $\mathcal{A}$ | Nonempty finite action space, shared by all states |
| $P(s' \mid s, a)$ | Probability of next state $s'$ |
| $r(s, a)$ | $\mathbb{E}[R_{t+1} \mid S_t = s, A_t = a]$ |
| $\gamma \in (0, 1)$ | Discount factor |

For later bounds, assume $0 \leq R_{t+1} \leq 1$. Rewards and next states may be dependent given $(S_t, A_t)$.

Puterman, Ch. 2. Nan Jiang, MDP Preliminaries, Sec. 1.

# Rewards and transition probabilities

The observed reward is random. Its conditional mean is the reward function:

$$r(s,a) = \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a].$$

The transition law describes the next state:

$$P(s' \mid s,a) = \mathbb{P}(S_{t+1} = s' \mid S_t = s, A_t = a), \qquad \sum_{s'} P(s' \mid s,a) = 1.$$

Some books write $R(s,a)$ for the expected reward. Here $R_{t+1}$ is a sample and $r(s,a)$ is its mean.

Reward and next state may share randomness. Their joint conditional law, together with the policy, determines the trajectory.

Puterman, Ch. 2. Sutton and Barto, Sec. 3.1.

# Policies

A general policy chooses a distribution of actions using the observed history:

$$\pi_t(a \mid H_t), \qquad H_t = (S_0, A_0, R_1, \ldots, S_t).$$

A **Markov policy** uses only the current time and state, $\pi_t(a \mid s)$.

A **stationary policy** uses the same rule at every time, $\pi(a \mid s)$.

A **deterministic stationary policy** chooses one action, $A_t = \pi(S_t)$.

We will prove that a deterministic stationary policy suffices for a finite discounted MDP.

Puterman, Sec. 2.1 and Ch. 6.

# A three-state MDP

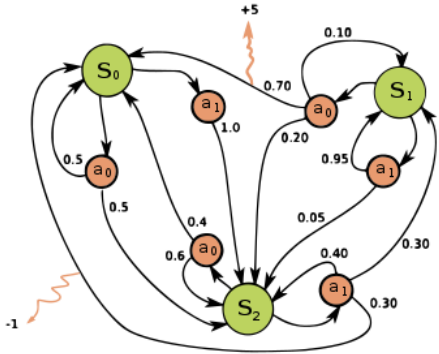
**States**: $\{s_0, s_1, s_2\}$

**Actions**: $\{a_0, a_1\}$

**From $s_2$, choose $a_0$:**

$$P(s_0 \mid s_2, a_0) = 0.4,$$
$$P(s_1 \mid s_2, a_0) = 0,$$
$$P(s_2 \mid s_2, a_0) = 0.6.$$

$+5, -1$: transition rewards




Example: Nan Jiang, CS 443, slide 4. Figure: waldoalvarez, Wikimedia Commons, CC BY-SA 4.0.

# Why discounting? Finite expected values

A loop with reward $1$ at every step:

$$1 + 1 + \cdots = \infty, \qquad 1 + \gamma + \gamma^2 + \cdots = \frac{1}{1-\gamma}.$$

**Bounded rewards**: $|R_{t+1}| \leq R_{\max}$

$$\left| \mathbb{E}_\pi \left[ \sum_{k=0}^{\infty} \gamma^k R_{k+1} \,\middle|\, S_0 = s \right] \right| \leq \frac{R_{\max}}{1-\gamma}.$$

Finite expected values even for policies that keep cycling.

Adapted from Nan Jiang, CS 443, slides 6–7.

# Discounting: interpretations and benefits

**Future rewards**: weight $\gamma^k$ after $k$ steps

Larger $\gamma$: more weight on the distant future

**Economic interpretation**: time value of money

**Guarantee of a stationary optimal policy**

$\pi^*(s)$ and $V^*(s)$: no time index

**Faster planning and learning**

Smaller $\gamma$: faster value iteration

Finite, time-homogeneous discounted MDPs. Nan Jiang, CS 443, slides 6–7. Economic view: Damodaran, The Time Value of Money.

# Discounted policy evaluation

# Expected discounted value

Fix a stationary policy $\pi$.

$$V^{\pi}(s) := \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R_{t+1} \,\middle|\, S_0 = s\right], \qquad 0 < \gamma < 1.$$

Expected total discounted reward starting from $s$

For $0 \leq R_{t+1} \leq 1$,

$$0 \leq V^{\pi}(s) \leq V_{\max} := \frac{1}{1-\gamma}.$$

Puterman, Ch. 6. Sutton and Barto, Sec. 3.5.

# Action values

First action $a$, then follow $\pi$.

$$Q^\pi(s,a) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R_{t+1} \,\middle|\, S_0 = s,\ A_0 = a,\ \pi \text{ thereafter}\right].$$

Average the first action under $\pi$:

$$V^\pi(s) = \sum_{a \in \mathcal{A}} \pi(a \mid s) Q^\pi(s,a).$$

Sutton and Barto, Sec. 3.5. Nan Jiang, MDP Preliminaries.

# How much of the distant future matters?

**Expected value of the omitted tail**

$$0 \le \mathbb{E}_\pi\left[\sum_{k=H}^{\infty} \gamma^k R_{k+1} \,\middle|\, S_0 = s\right] \le \frac{\gamma^H}{1-\gamma}.$$

For tail error at most $\varepsilon$, choose

$$H \ge \frac{\log \frac{1}{(1-\gamma)\varepsilon}}{-\log \gamma}, \qquad (1-\gamma)\varepsilon < 1.$$

As $\gamma \uparrow 1$, $-\log\gamma \sim 1-\gamma$ ($\varepsilon$ fixed):

$$\frac{\log \frac{1}{(1-\gamma)\varepsilon}}{-\log \gamma} \sim \frac{\log \frac{1}{(1-\gamma)\varepsilon}}{1-\gamma}.$$

# Deriving the Bellman equation

Fix a stationary policy $\pi(a \mid s)$.

$$
\begin{aligned}
V^\pi(s) &= \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{k+1} \,\middle|\, S_0 = s\right] \\
&= \mathbb{E}_\pi\left[R_1 + \gamma \sum_{k=0}^{\infty} \gamma^k R_{k+2} \,\middle|\, S_0 = s\right] \\
&= \mathbb{E}_\pi\left[R_1 + \gamma \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{k+2} \,\middle|\, S_0, S_1\right] \,\middle|\, S_0 = s\right] \\
&= \mathbb{E}_\pi\left[R_1 + \gamma \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{k+2} \,\middle|\, S_1\right] \,\middle|\, S_0 = s\right] \\
&= \mathbb{E}_\pi[R_1 + \gamma V^\pi(S_1) \mid S_0 = s] \\
&= \sum_a \pi(a \mid s) \left[r(s,a) + \gamma \sum_{s'} P(s' \mid s, a) V^\pi(s')\right].
\end{aligned}
$$

Adapted from Nan Jiang, CS 443, slides 9–11. Puterman, Sec. 6.1.

# Matrix-vector form

**Column vector** (fixed state ordering)

$$V^\pi = \left(V^\pi(s)\right)_{s\in\mathcal{S}} \in \mathbb{R}^{|\mathcal{S}|}.$$

$$r^\pi(s) = \sum_a \pi(a \mid s) r(s,a), \qquad P^\pi(s,s') = \sum_a \pi(a \mid s) P(s' \mid s,a).$$

**Bellman equation as a linear system**

$$V^\pi = r^\pi + \gamma P^\pi V^\pi \quad \Longleftrightarrow \quad (I - \gamma P^\pi) V^\pi = r^\pi.$$

**Unique solution**: $I - \gamma P^\pi$ is invertible for $0 < \gamma < 1$

$$V^\pi = (I - \gamma P^\pi)^{-1} r^\pi.$$

Puterman, Sec. 6.1.

# Discounted state occupancy

## Unnormalized discounted visits

$$\begin{aligned}\left[(I-\gamma P^\pi)^{-1}\right]_{ss'} &= \sum_{t=0}^{\infty} \gamma^t \left[(P^\pi)^t\right]_{ss'} \\ &= \sum_{t=0}^{\infty} \gamma^t \mathbb{P}_\pi(S_t = s' \mid S_0 = s).\end{aligned}$$

Expected discounted visits from $s$ to $s'$, including $t = 0$

## Normalized occupancy distribution

$$d_s^\pi(s') = (1-\gamma)\left[(I-\gamma P^\pi)^{-1}\right]_{ss'}, \qquad \sum_{s'} d_s^\pi(s') = 1.$$

$$V^\pi(s) = \frac{1}{1-\gamma} \sum_{s'} d_s^\pi(s') r^\pi(s').$$

Nan Jiang, CS 443, slides 13–14.

# Bellman equation for discounted occupancy

Fix the initial state $s$.

For every $s' \in \mathcal{S}$,

$$d_s^\pi(s') = (1-\gamma)\mathbf{1}\{s' = s\} + \gamma \sum_{x \in \mathcal{S}} P^\pi(x, s') d_s^\pi(x).$$

**Exercise**: prove this identity from the definition of $d_s^\pi$.

# A recurrent two-state calculation

State order: $(s_1, s_2)$. Under a fixed policy,

$$P^\pi = \begin{pmatrix} 1/2 & 1/2 \\ 0 & 1 \end{pmatrix}, \qquad r^\pi = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \qquad \gamma = 0.9.$$

The Bellman equations are

$$V^\pi(s_1) = 0.45V^\pi(s_1) + 0.45V^\pi(s_2), \qquad V^\pi(s_2) = 1 + 0.9V^\pi(s_2).$$

Solving gives

$$V^\pi(s_1) = \frac{90}{11} \approx 8.18, \qquad V^\pi(s_2) = 10.$$

State $s_1$ has no immediate reward. Its value comes from eventually reaching $s_2$.

# A Bellman operator for a fixed policy

For a proposed continuation value $v$, define

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

This is one exact lookahead step: follow $\pi$ now and assign value $v$ to the next state.

The correct value is a fixed point:

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

Iterative policy evaluation repeatedly applies this operator.

Puterman, Sec. 6.1.

# Value iteration for a fixed policy

Fix $\pi$. Initialize $v_0 = 0$.

For $k = 0, 1, \ldots$, update every state:

$$v_{k+1} = \mathcal{T}^\pi v_k = r^\pi + \gamma P^\pi v_k,$$

$$v_{k+1}(s) = \sum_a \pi(a \mid s) \left[ r(s,a) + \gamma \sum_{s'} P(s' \mid s, a) v_k(s') \right].$$

All right-hand sides use the old vector $v_k$.

**Geometric convergence**

$$\|v_k - V^\pi\|_\infty \leq \gamma^k \|v_0 - V^\pi\|_\infty \leq \frac{\gamma^k}{1-\gamma} \longrightarrow 0.$$

Iterative policy evaluation. Sutton and Barto, Sec. 4.1. Puterman, Sec. 6.1.

# Why fixed-policy value iteration converges

## $\gamma$-contraction

$$\begin{aligned}\left\|\mathcal{T}^\pi u - \mathcal{T}^\pi v\right\|_\infty &= \gamma \max_s \left|\sum_{s'} P^\pi(s,s')(u(s') - v(s'))\right| \\ &\leq \gamma \max_s \sum_{s'} P^\pi(s,s') \left\|u - v\right\|_\infty = \gamma \left\|u - v\right\|_\infty .\end{aligned}$$

**Unique fixed point**: $V^\pi = \mathcal{T}^\pi V^\pi$. For $w = \mathcal{T}^\pi w$,

$$\left\|w - V^\pi\right\|_\infty \leq \gamma \left\|w - V^\pi\right\|_\infty \quad \Longrightarrow \quad w = V^\pi.$$

**Convergence**: $v_{k+1} = \mathcal{T}^\pi v_k$

$$\begin{aligned}\left\|v_{k+1} - V^\pi\right\|_\infty &= \left\|\mathcal{T}^\pi v_k - \mathcal{T}^\pi V^\pi\right\|_\infty \leq \gamma \left\|v_k - V^\pi\right\|_\infty , \\ \left\|v_k - V^\pi\right\|_\infty &\leq \gamma^k \left\|v_0 - V^\pi\right\|_\infty \longrightarrow 0.\end{aligned}$$

Puterman, Sec. 6.1.

# The Bellman equation for a fixed policy's action values

Fix the first action $a$, then follow $\pi$:

$$Q^\pi(s,a) = r(s,a) + \gamma \sum_{s'} P(s' \mid s,a) V^\pi(s').$$

At the next state, average the next action under the same policy:

$$\boxed{Q^\pi(s,a) = r(s,a) + \gamma \sum_{s'} P(s' \mid s,a) \sum_{a'} \pi(a' \mid s') Q^\pi(s',a').}$$

The first action is specified by the argument $a$. Future actions follow $\pi$. There is no maximization in this evaluation equation.

Sutton and Barto, Sec. 3.5.

# Q-value iteration for a fixed policy

Fix $\pi$. Initialize $q_0 = 0$.

For $k = 0, 1, \ldots$, update every state-action pair:

$$q_{k+1} = \mathcal{T}_Q^\pi q_k,$$
$$(\mathcal{T}_Q^\pi q)(s,a) = r(s,a) + \gamma \sum_{s'} P(s' \mid s,a) \sum_{a' \in \mathcal{A}} \pi(a' \mid s') q(s',a').$$

All right-hand sides use the old table $q_k$.

**Geometric convergence**

$$\|q_k - Q^\pi\|_\infty \leq \gamma^k \|q_0 - Q^\pi\|_\infty \leq \frac{\gamma^k}{1-\gamma} \longrightarrow 0.$$

Sutton and Barto, Secs. 3.5 and 4.1.

# Why fixed-policy Q-value iteration converges

$\gamma$-**contraction**: for every $(s, a)$,

$$\begin{aligned}
&\left|(\mathcal{T}_Q^\pi q)(s,a) - (\mathcal{T}_Q^\pi u)(s,a)\right| \\
&\quad \le \gamma \sum_{s'} P(s' \mid s,a) \sum_{a' \in \mathcal{A}} \pi(a' \mid s') \left|q(s',a') - u(s',a')\right| \\
&\quad \le \gamma \left\|q - u\right\|_\infty .
\end{aligned}$$

Take the maximum over all $(s, a)$.

**Unique fixed point**: $Q^\pi = \mathcal{T}_Q^\pi Q^\pi$. If $w = \mathcal{T}_Q^\pi w$,

$$\left\|w - Q^\pi\right\|_\infty \le \gamma \left\|w - Q^\pi\right\|_\infty \quad \Longrightarrow \quad w = Q^\pi.$$

**Convergence**: $q_{k+1} = \mathcal{T}_Q^\pi q_k$

$$\begin{aligned}
\left\|q_{k+1} - Q^\pi\right\|_\infty &= \left\|\mathcal{T}_Q^\pi q_k - \mathcal{T}_Q^\pi Q^\pi\right\|_\infty \le \gamma \left\|q_k - Q^\pi\right\|_\infty , \\
\left\|q_k - Q^\pi\right\|_\infty &\le \gamma^k \left\|q_0 - Q^\pi\right\|_\infty \longrightarrow 0.
\end{aligned}$$

# Discounted state-action occupancy

Start at $s$ and follow $\pi$ from $t = 0$.

**Normalized occupancy distribution**

$$d_s^\pi(s', a') := (1-\gamma) \sum_{t=0}^{\infty} \gamma^t \mathbb{P}_\pi(S_t = s', A_t = a' \mid S_0 = s)$$
$$= d_s^\pi(s') \pi(a' \mid s'), \qquad \sum_{s', a'} d_s^\pi(s', a') = 1.$$

**Bellman equation**

$$d_s^\pi(s', a') = \pi(a' \mid s') \left[ (1-\gamma) \mathbf{1}\{s' = s\} + \gamma \sum_{\substack{x \in \mathcal{S} \\ b \in \mathcal{A}}} P(s' \mid x, b) d_s^\pi(x, b) \right].$$

**Exercise**: prove the Bellman equation.

# Values and state-action occupancy

**State values**

$$V^\pi(s) = \frac{1}{1-\gamma} \sum_{s',a'} d^\pi_s(s',a') r(s',a').$$

**Action values**: fix the first action $a$, then follow $\pi$

$$d^\pi_{s,a}(s',a') := (1-\gamma) \sum_{t=0}^{\infty} \gamma^t \mathbb{P}(S_t = s', A_t = a' \,|\, S_0 = s, A_0 = a, \pi \text{ thereafter}) .$$

$$Q^\pi(s,a) = \frac{1}{1-\gamma} \sum_{s',a'} d^\pi_{s,a}(s',a') r(s',a').$$

Discounted visits weighted by rewards

# Optimality

# Optimizing over policies

Policy evaluation fixes $\pi$. Optimality asks for the largest achievable value:

$$V^*(s) = \sup_\pi V^\pi(s), \qquad Q^*(s,a) = \sup_\pi Q^\pi(s,a).$$

For $Q^*$, the first action is $a$ and the supremum chooses future behavior.

The supremum may initially include randomized, history-dependent policies. We will show that one deterministic stationary policy attains $V^*(s)$ for every state simultaneously.

Puterman, Sec. 6.2.

# The Bellman optimality operator

Choose the best first action for continuation value $v$:

$$(\mathcal{T}v)(s) = \max_{a \in \mathcal{A}} \left[ r(s,a) + \gamma \sum_{s'} P(s' \mid s,a) v(s') \right].$$

For comparison,

$$(\mathcal{T}^{\pi}v)(s) = \sum_{a} \pi(a \mid s) \left[ r(s,a) + \gamma \sum_{s'} P(s' \mid s,a) v(s') \right].$$

The difference is the choice of the current action: maximize or average under $\pi$.

Puterman, Sec. 6.2.

# Two properties of the Bellman operators

**Monotonicity:** if $u(s) \le v(s)$ for every $s$, then

$$\mathcal{T}^\pi u \le \mathcal{T}^\pi v, \qquad \mathcal{T} u \le \mathcal{T} v.$$

**Contraction in the sup norm:**

$$\left\| \mathcal{T}^\pi u - \mathcal{T}^\pi v \right\|_\infty \le \gamma \left\| u - v \right\|_\infty,$$

$$\left\| \mathcal{T} u - \mathcal{T} v \right\|_\infty \le \gamma \left\| u - v \right\|_\infty.$$

Puterman, Sec. 6.2. Nan Jiang, MDP Preliminaries.

# Proof of contraction

For finite collections of numbers,

$$\left|\max_a x_a - \max_a y_a\right| \le \max_a |x_a - y_a|.$$

Hence, at any state $s$,

$$\begin{aligned}
|(\mathcal{T}u)(s) - (\mathcal{T}v)(s)| &\le \gamma \max_a \left|\sum_{s'} P(s' \mid s, a)(u(s') - v(s'))\right| \\
&\le \gamma \max_a \sum_{s'} P(s' \mid s, a) \left\|u - v\right\|_\infty \\
&= \gamma \left\|u - v\right\|_\infty .
\end{aligned}$$

Taking the maximum over $s$ proves the claim. The same argument with the average under $\pi$ proves it for $\mathcal{T}^\pi$.

# A contraction has one fixed point

Let $v_{k+1} = \mathcal{T} v_k$. Successive differences satisfy

$$\|v_{k+1} - v_k\|_\infty \leq \gamma^k \|v_1 - v_0\|_\infty .$$

Their geometric sum is finite, so $(v_k)$ is Cauchy and converges to a vector $v^*$. Continuity gives $\mathcal{T} v^* = v^*$.

If $u^*$ were another fixed point, then

$$\|u^* - v^*\|_\infty \leq \gamma \|u^* - v^*\|_\infty ,$$

so $u^* = v^*$.

It remains to show that this fixed point equals the optimal return.

# A greedy policy attains the fixed point

Choose a deterministic policy

$$\pi^*(s) \in \arg\max_a \left[ r(s,a) + \gamma \sum_{s'} P(s' \mid s,a) v^*(s') \right].$$

Then

$$\mathcal{T}^{\pi^*} v^* = \mathcal{T} v^* = v^*.$$

Since $\mathcal{T}^{\pi^*}$ has the unique fixed point $V^{\pi^*}$,

$$v^* = V^{\pi^*}.$$

Puterman, Sec. 6.2, including Thm. 6.2.7.

# Why history and randomization cannot improve the value

The fixed-point equation implies, for every action $a$,

$$v^*(s) \geq r(s,a) + \gamma \sum_{s'} P(s' \mid s,a) v^*(s').$$

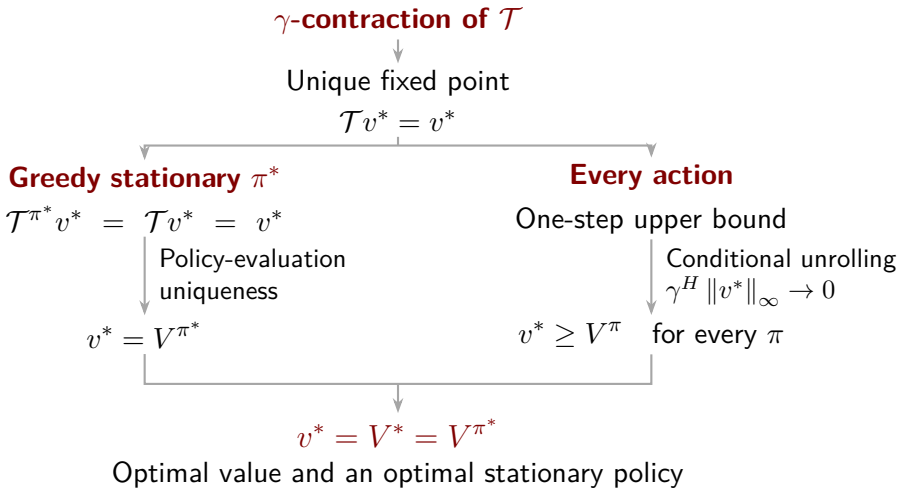
Under any admissible policy $\pi$, apply the bound conditionally at each step:

$$\begin{aligned} v^*(s) &\geq \mathbb{E}_\pi[R_1 + \gamma v^*(S_1) \,|\, S_0 = s] \\ &\geq \mathbb{E}_\pi\Big[R_1 + \gamma R_2 + \gamma^2 v^*(S_2) \,\Big|\, S_0 = s\Big] \\ &\geq \cdots \geq \mathbb{E}_\pi\left[\sum_{t=0}^{H-1} \gamma^t R_{t+1} + \gamma^H v^*(S_H) \,\middle|\, S_0 = s\right], \qquad H \geq 2. \end{aligned}$$

Because $v^*$ is bounded, the last term tends to zero as $H \to \infty$.

Thus $v^*(s) \geq V^\pi(s)$ for every policy, and equality holds for $\pi^*$. We may write $v^* = V^*$.

# Optimality proof sketch



# The optimal action-value equation

Once $V^*$ is known,

$$Q^*(s,a) = r(s,a) + \gamma \sum_{s'} P(s' \mid s,a) V^*(s').$$

Since $V^*(s') = \max_{a'} Q^*(s',a')$,

$$Q^*(s,a) = r(s,a) + \gamma \sum_{s'} P(s' \mid s,a) \max_{a'} Q^*(s',a').$$

Puterman, Ch. 6. Sutton and Barto, Sec. 3.6.

# The optimal action-value Bellman operator

For any action-value table $Q$, define

$$(\mathcal{T}_Q Q)(s,a) = r(s,a) + \gamma \sum_{s'} P(s' \mid s,a) \max_{a' \in \mathcal{A}} Q(s',a').$$

The optimal table satisfies $Q^\star = \mathcal{T}_Q Q^\star$. The maximum describes how the agent will act after the first action.

Exact iteration $Q_{k+1} = \mathcal{T}_Q Q_k$ is planning with a known model. A sampled version will later give Q-learning.

Puterman, Ch. 6. Sutton and Barto, Sec. 6.5.

# Planning algorithms

# Value iteration

Start with a value vector $v_0$, often zero. For $k = 0, 1, \ldots$, update every state:

$$v_{k+1}(s) = \max_a \left[ r(s,a) + \gamma \sum_{s'} P(s' \mid s, a) v_k(s') \right].$$

All right-hand sides use the old vector $v_k$.

Contraction gives the error bound

$$\|v_k - V^*\|_\infty \leq \gamma^k \|v_0 - V^*\|_\infty .$$

For $v_0 = 0$, the right-hand side is at most $\gamma^k/(1-\gamma)$.

Puterman, Sec. 6.3. Nan Jiang, CS 443 value-iteration slides.

# A stopping rule from the Bellman residual

We cannot inspect $\|v - V^*\|_\infty$ directly. We can compute

$$\varepsilon_{\mathrm{B}} = \|\mathcal{T} v - v\|_\infty .$$

By the triangle inequality and contraction,

$$\|v - V^*\|_\infty \leq \varepsilon_{\mathrm{B}} + \gamma \|v - V^*\|_\infty .$$

Therefore

$$\boxed{\|v - V^*\|_\infty \leq \frac{\varepsilon_{\mathrm{B}}}{1-\gamma}.}$$

In value iteration, $\varepsilon_{\mathrm{B}} = \|v_{k+1} - v_k\|_\infty$ for $v = v_k$.

Puterman, Sec. 6.3.

# An approximate value gives an approximate policy

Let $\pi_v$ be greedy with respect to $v$, so $\mathcal{T}^{\pi_v} v = \mathcal{T} v$. Write $e = \|v - V^*\|_\infty$ and $D = \|V^* - V^{\pi_v}\|_\infty$.

$$\begin{aligned} D &\le \|\mathcal{T} V^* - \mathcal{T} v\|_\infty + \|\mathcal{T}^{\pi_v} v - \mathcal{T}^{\pi_v} V^{\pi_v}\|_\infty \\ &\le \gamma e + \gamma(e + D). \end{aligned}$$

Consequently,

$$\boxed{\|V^* - V^{\pi_v}\|_\infty \le \frac{2\gamma}{1-\gamma} \|v - V^*\|_\infty .}$$

Together with the residual bound, this gives a computable policy certificate.

# Policy improvement

Evaluate a policy $\pi$, then choose $\pi'$ greedy with respect to $V^\pi$:

$$\pi'(s) \in \arg\max_a \left[ r(s,a) + \gamma \sum_{s'} P(s' \mid s,a) V^\pi(s') \right].$$

By greediness,

$$\mathcal{T}^{\pi'} V^\pi = \mathcal{T} V^\pi \geq \mathcal{T}^\pi V^\pi = V^\pi.$$

Monotonicity then gives

$$V^\pi \leq \mathcal{T}^{\pi'} V^\pi \leq (\mathcal{T}^{\pi'})^2 V^\pi \leq \cdots \longrightarrow V^{\pi'}.$$

Thus $V^{\pi'} \geq V^\pi$ at every state.

Puterman, Sec. 6.4.

# Policy iteration

Start with a deterministic policy $\pi_0$.

1. **Evaluation:** solve $(I - \gamma P^{\pi_k})V^{\pi_k} = r^{\pi_k}$.
2. **Improvement:** make $\pi_{k+1}$ greedy with respect to $V^{\pi_k}$.
3. Stop when $\pi_{k+1} = \pi_k$.

If the old action is already a maximizer, keep it.

This rule prevents changing the policy merely because several actions tie.

Puterman, Sec. 6.4.

# Strict improvement before optimality

Suppose the current policy is not optimal: $V^{\pi_k} \neq V^*$.

$$\mathcal{T}V^{\pi_k} \geq \mathcal{T}^{\pi_k}V^{\pi_k} = V^{\pi_k}.$$

If equality held everywhere, $V^{\pi_k}$ would be the fixed point $V^*$ of $\mathcal{T}$. Thus

$$\exists s: \quad (\mathcal{T}V^{\pi_k})(s) > V^{\pi_k}(s).$$

Greedy improvement and policy evaluation give

$$V^{\pi_{k+1}} \geq \mathcal{T}^{\pi_{k+1}}V^{\pi_k} = \mathcal{T}V^{\pi_k} \geq V^{\pi_k}.$$

At the state above, $V^{\pi_{k+1}}(s) > V^{\pi_k}(s)$.

In particular, $V^{\pi_{k+1}} \neq V^*$ implies $V^{\pi_k} \neq V^*$, so strict improvement applies.

Puterman, Sec. 6.4.

# Distinct policies and finite termination

**Corollary**: if $V^{\pi_k} \neq V^*$, then $\pi_0, \ldots, \pi_k$ are pairwise distinct.

**Finite termination**

With finitely many deterministic policies and no repetition, PI must reach optimality.

At $V^{\pi_k} = V^*$, every current action is greedy:

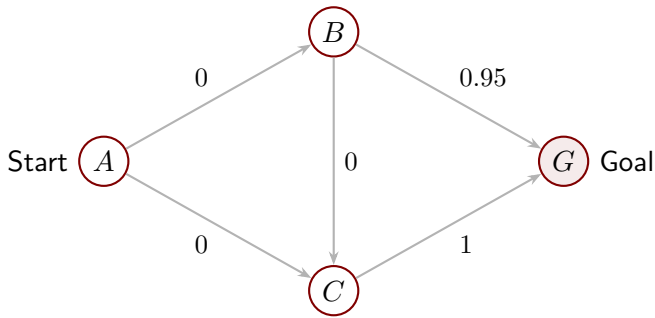
$$\mathcal{T}^{\pi_k} V^* = V^* = \mathcal{T} V^*.$$

Tie retention gives $\pi_{k+1} = \pi_k$, so PI stops.

Puterman, Sec. 6.4.

# Route planning in a discounted MDP

$\gamma = 0.9$. Arrows show transitions and their immediate rewards.



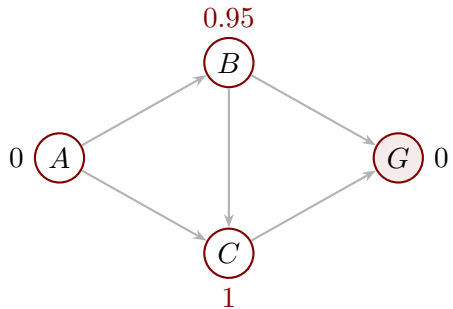

At $G$, stay forever with reward $0$.

Both reward size and arrival time matter.

# VI: successive value estimates

Initialize $v_0 = 0$. Each sweep uses the previous value table.

**After one sweep: $v_1$**




**Value estimates**

| | $A$ | $B$ | $C$ |
|---|---|---|---|
| $v_0$ | 0 | 0 | 0 |
| $v_1$ | 0 | 0.95 | 1 |
| $v_2$ | 0.9 | 0.95 | 1 |

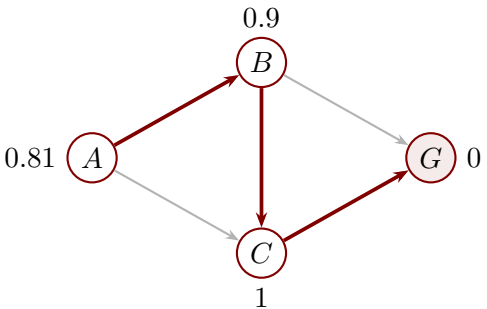
$$v_2(A) = 0.9 \max\{0.95, 1\} = 0.9.$$

The reward reaches $B, C$ first, then $A$.

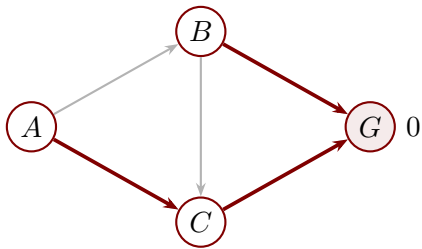
$v_1(A) = 0$: the reward two transitions away has not reached $A$ yet.

# PI: full evaluation and policy improvement

**1. Evaluate the whole policy $\pi_0$**




**2. Improve all states using $V^{\pi_0}$**




$V^{\pi_0}(A) = 0 + \gamma(0 + \gamma \cdot 1) = 0.81$

Maroon arrows show the chosen actions.

$\pi_1(A) = C$: $0.9 > 0.81$. $\pi_1(B) = G$: $0.95 > 0.9$.

**Then re-evaluate**: $V^{\pi_1} = (0.9, 0.95, 1)$ on $(A, B, C)$.

# Learning settings

# From planning to learning

So far, a backup used the full model:

$$r(s,a) + \gamma \sum_{s'} P(s' \mid s,a) v(s').$$

In learning, the model is unknown. Instead, we observe

$$(S_t, A_t, R_{t+1}, S_{t+1})$$

and construct a sampled target such as $R_{t+1} + \gamma v(S_{t+1})$.

We must specify which policy generates observations, which policy we want to evaluate or improve, and which states the data cover.

# A generative-model oracle

**Access**: query any $(s, a) \in \mathcal{S} \times \mathcal{A}$

$$\underbrace{(s,a)}_{\text{chosen query}} \xrightarrow{\mathcal{O}} \underbrace{(R, S')}_{\text{fresh sample}} .$$

$$0 \le R \le 1, \qquad \mathbb{E}[R \mid s, a] = r(s,a), \qquad S' \sim P(\cdot \mid s, a).$$

**Sampling design**: $n$ independent samples from each pair

$$S = |\mathcal{S}|, \qquad A = |\mathcal{A}|, \qquad N = SA\,n \text{ oracle calls.}$$

Known states, actions, and $\gamma$. Unknown $r$ and $P$.

Nan Jiang, CS 443, Learning Settings, pp. 7–9.

# An estimated MDP

At each $(s, a)$, write the $n$ samples as $(R_i, S_i')_{i=1}^n$.

**Empirical reward and transition probabilities**

$$\widehat{r}(s,a) = \frac{1}{n}\sum_{i=1}^{n} R_i, \qquad \widehat{P}(s' \mid s,a) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}\{S_i' = s'\}.$$

**Estimated model**

$$\widehat{M} = (\mathcal{S}, \mathcal{A}, \widehat{P}, \widehat{r}, \gamma).$$

$$0 \le \widehat{r}(s,a) \le 1, \qquad \widehat{P}(s' \mid s,a) \ge 0, \qquad \sum_{s'} \widehat{P}(s' \mid s,a) = 1.$$

Same states, actions, and discount. Estimate the unknown model from data.

Nan Jiang, Notes on Tabular Methods, Sec. 1.1.

# Certainty-equivalence planning

Treat $\widehat{M}$ as the model. Solve it exactly by policy iteration:

$$\widehat{\pi}^*(s) \in \arg\max_a \left[ \widehat{r}(s,a) + \gamma \sum_{s'} \widehat{P}(s' \mid s,a) \widehat{V}^*(s') \right].$$

$\widehat{V}^\pi$: value in $\widehat{M}$. $V^\pi$: value in the true MDP $M$.

$$\widehat{V}^{\widehat{\pi}^*} = \widehat{V}^*, \qquad V^{\pi^*} = V^*.$$

**Sample-complexity goal**: for $\varepsilon, \delta \in (0,1)$,

$$\mathbb{P}\left( \left\| V^* - V^{\widehat{\pi}^*} \right\|_\infty \le \varepsilon \right) \ge 1 - \delta.$$

Probability is over the oracle samples.

Nan Jiang, Notes on Tabular Methods, Secs. 1.1 and 2.

# Reward estimation

$$\varepsilon_r := \max_{s,a} |\widehat{r}(s,a) - r(s,a)|.$$

For one pair and $t > 0$, Hoeffding gives

$$\mathbb{P}(|\widehat{r}(s,a) - r(s,a)| > t) \leq 2e^{-2nt^2}.$$

A union bound over $SA$ pairs gives

$$\mathbb{P}(\varepsilon_r > t) \leq 2SAe^{-2nt^2}.$$

With probability at least $1 - \delta/2$,

$$\varepsilon_r \leq \sqrt{\frac{\log(4SA/\delta)}{2n}}.$$

Nan Jiang, Notes on Tabular Methods, Sec. 2.1.

# Coordinatewise transition estimation

For each $(s, a, s')$, Hoeffding applies to $\mathbf{1}\{S_i' = s'\}$:

$$\mathbb{P}\Big(|\widehat{P}(s' \mid s, a) - P(s' \mid s, a)| > t\Big) \le 2e^{-2nt^2}.$$

A union bound over $S^2A$ entries gives, with probability $\ge 1 - \delta/2$,

$$\max_{s,a,s'} |\widehat{P}(s' \mid s, a) - P(s' \mid s, a)| \le \sqrt{\frac{\log(4S^2A/\delta)}{2n}}.$$

On the same event,

$$\varepsilon_P := \max_{s,a} \Big\|\widehat{P}(\cdot \mid s, a) - P(\cdot \mid s, a)\Big\|_1 \le S\sqrt{\frac{\log(4S^2A/\delta)}{2n}}.$$

Joint reward and transition bounds: probability $\ge 1 - \delta$.

Nan Jiang, Notes on Tabular Methods, Sec. 2.1.

# Simulation lemma

**Reward error after averaging over the policy**

$$|\widehat{r}^{\pi}(s) - r^{\pi}(s)| \le \sum_{a} \pi(a \mid s) |\widehat{r}(s,a) - r(s,a)| \le \varepsilon_r.$$

**Transition error after averaging over the policy**

$$\sum_{s'} |\widehat{P}^{\pi}(s,s') - P^{\pi}(s,s')| \le \sum_{a} \pi(a \mid s) \sum_{s'} |\widehat{P}(s' \mid s,a) - P(s' \mid s,a)|$$
$$\le \varepsilon_P.$$

**Uniform value bound (including $\widehat{\pi}^*$)**

$$\left\| \widehat{V}^{\pi} - V^{\pi} \right\|_{\infty} \le \frac{\varepsilon_r}{1-\gamma} + \frac{\gamma \varepsilon_P}{2(1-\gamma)^2}.$$

Nan Jiang, Notes on Tabular Methods, Lemma 1.

# Simulation lemma: Bellman comparison

Fix $\pi$ and set $D = \left\|\widehat{V}^\pi - V^\pi\right\|_\infty$. The two Bellman equations are

$$V^\pi = r^\pi + \gamma P^\pi V^\pi, \qquad \widehat{V}^\pi = \widehat{r}^\pi + \gamma \widehat{P}^\pi \widehat{V}^\pi.$$

Subtract, then add and subtract $\gamma \widehat{P}^\pi V^\pi$:

$$\begin{aligned}\widehat{V}^\pi - V^\pi &= \widehat{r}^\pi - r^\pi + \gamma(\widehat{P}^\pi \widehat{V}^\pi - P^\pi V^\pi) \\ &= \widehat{r}^\pi - r^\pi + \gamma \widehat{P}^\pi(\widehat{V}^\pi - V^\pi) + \gamma(\widehat{P}^\pi - P^\pi)V^\pi.\end{aligned}$$

Take absolute values and use $\sum_{s'} \widehat{P}^\pi(s, s') = 1$:

$$\begin{aligned}D &\le \varepsilon_r + \gamma \max_s \sum_{s'} \widehat{P}^\pi(s, s')|\widehat{V}^\pi(s') - V^\pi(s')| + \gamma \left\|(\widehat{P}^\pi - P^\pi)V^\pi\right\|_\infty \\ &\le \varepsilon_r + \gamma D + \gamma \left\|(\widehat{P}^\pi - P^\pi)V^\pi\right\|_\infty.\end{aligned}$$

Nan Jiang, Notes on Tabular Methods, Lemma 1.

# Simulation lemma: bounding the transition term

Both transition matrices have row sums one, so $(\widehat{P}^\pi - P^\pi)\mathbf{1} = 0$.
Since $0 \le V^\pi(s') \le \frac{1}{1-\gamma}$, we have $\left|V^\pi(s') - \frac{1}{2(1-\gamma)}\right| \le \frac{1}{2(1-\gamma)}$.

$$\begin{aligned}\left|[(\widehat{P}^\pi - P^\pi)V^\pi](s)\right| &= \left|\sum_{s'}[\widehat{P}^\pi(s,s') - P^\pi(s,s')]\left[V^\pi(s') - \frac{1}{2(1-\gamma)}\right]\right| \\ &\le \frac{1}{2(1-\gamma)}\sum_{s'}|\widehat{P}^\pi(s,s') - P^\pi(s,s')| \le \frac{\varepsilon_P}{2(1-\gamma)}.\end{aligned}$$

Take $\max_s$, substitute into the previous inequality, and rearrange:

$$\begin{aligned}(1-\gamma)D &\le \varepsilon_r + \frac{\gamma\varepsilon_P}{2(1-\gamma)}, \\ D &\le \frac{\varepsilon_r}{1-\gamma} + \frac{\gamma\varepsilon_P}{2(1-\gamma)^2} =: e.\end{aligned}$$

Nan Jiang, Notes on Tabular Methods, Lemma 1.

# Evaluation error and policy loss

$\pi^*$ is optimal in $M$. $\widehat{\pi}^*$ is optimal in $\widehat{M}$. For every state $s$,

$$\begin{aligned} V^*(s) - V^{\widehat{\pi}^*}(s) &= [V^{\pi^*}(s) - \widehat{V}^{\pi^*}(s)] \\ &\quad + [\widehat{V}^{\pi^*}(s) - \widehat{V}^{\widehat{\pi}^*}(s)] \\ &\quad + [\widehat{V}^{\widehat{\pi}^*}(s) - V^{\widehat{\pi}^*}(s)] \\ &\leq e + 0 + e = 2e. \end{aligned}$$

The middle term is nonpositive by optimality in $\widehat{M}$.

$$\left\| V^* - V^{\widehat{\pi}^*} \right\|_\infty \leq \frac{2\varepsilon_r}{1-\gamma} + \frac{\gamma \varepsilon_P}{(1-\gamma)^2}.$$

Nan Jiang, Notes on Tabular Methods, Lemma 2.

# Sample complexity

Combining the reward and transition bounds, with probability $\geq 1-\delta$,

$$\left\|V^* - V^{\widehat{\pi}^*}\right\|_\infty \leq \frac{2(1-\gamma)+\gamma S}{(1-\gamma)^2}\sqrt{\frac{\log(4S^2A/\delta)}{2n}} \leq \frac{2S}{(1-\gamma)^2}\sqrt{\frac{\log(4S^2A/\delta)}{2n}}.$$

**Sufficient samples per pair**

$$n = \left\lceil \frac{2S^2\log(4S^2A/\delta)}{\varepsilon^2(1-\gamma)^4} \right\rceil.$$

**Total oracle calls** ($\widetilde{O}$ omits logarithmic factors)

$$N = SA\,n = \widetilde{O}\left(\frac{S^3A}{\varepsilon^2(1-\gamma)^4}\right).$$

Nan Jiang, Notes on Tabular Methods, Sec. 2.1.

# Sharper sample-complexity bounds

$n$: samples per pair. $N = SAn$: total oracle calls.

| Analysis | Total samples: $N = \widetilde{O}(\cdot)$ |
|---|---|
| Naive coordinate bounds | $\frac{S^3A}{\varepsilon^2(1-\gamma)^4}$ |
| Direct $\ell_1$ concentration | $\frac{S^2A}{\varepsilon^2(1-\gamma)^4}$ |
| Fixed-$Q^*$ comparison | $\frac{SA}{\varepsilon^2(1-\gamma)^6}$ |
| Bernstein inequality | $\frac{SA}{\varepsilon^2(1-\gamma)^3}$ |

Nan Jiang, Notes on Tabular Methods, Sec. 2. Bernstein rate: Agarwal, Kakade and Yang, Theorem 1.

# What data can the learner obtain?

| Access | Available observations | Main limitation |
|---|---|---|
| Online interaction | Follow a policy and observe a trajectory | Actions determine future data |
| Generative model | Request a fresh transition at a chosen $(s, a)$ | Assumes query access to chosen pairs |
| Offline data | A fixed collection of transitions | Cannot collect missing information |

In all three settings, a transition has the form $(S_t, A_t, R_{t+1}, S_{t+1})$. The learner does not receive the full transition distribution $P(\cdot \mid s, a)$.

# Prediction and control from samples

**Policy evaluation**: fix a policy $\pi$ and estimate its value.

$$V^\pi(s) = \mathbb{E}_\pi \left[ \sum_{k=0}^{\infty} \gamma^k R_{k+1} \,\middle|\, S_0 = s \right].$$

**Control**: learn how to choose actions to maximize the return.

$$Q^\star(s,a) = r(s,a) + \gamma \mathbb{E}_{S' \sim P(\cdot|s,a)} \left[ \max_{a'} Q^\star(S', a') \right].$$

Sutton and Barto, Chs. 5–6. Nan Jiang, CS 443: value prediction and control.

# On-policy and off-policy learning

The **behavior policy** $b$ generates the observed actions. The **target policy** $\pi$ is the policy whose value we want to learn.

| | Data-generating actions | Evaluation target |
|---|---|---|
| On-policy | $A_t \sim \pi(\cdot \mid S_t)$ | The same policy $\pi$ |
| Off-policy | $A_t \sim b(\cdot \mid S_t)$ | A different policy $\pi$ |

Sutton and Barto, Chs. 5–6.

# Several independent choices of learning setting

| Question | Possibilities | What it determines |
|---|---|---|
| What is the task? | Prediction or control | Evaluate a policy or improve it |
| What generates data? | Online, generative, or offline | How new information is obtained |
| Which policy is sampled? | On-policy or off-policy | Relation between behavior and target |

Online data can be off-policy. An offline dataset can be on-policy for the policy that generated it.

# A complete return as a target

**Policy**: fixed stationary $\pi$

**Data**: rollouts reaching zero-reward absorbing $\Delta$

$T < \infty$: absorption time

$$G_t = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{T-t-1} R_T, \qquad t < T.$$

Zero tail after $T$: complete discounted return

**Monte Carlo update**

$$V(S_t) \leftarrow V(S_t) + \alpha\big(G_t - V(S_t)\big).$$

Wait for the complete return.

Sutton and Barto, Sec. 5.1.

# First-visit Monte Carlo

Maintain a value $V(s)$ and a count $N(s)$ for every nonterminal state.

1. Generate an independent episode using the fixed policy $\pi$.
2. Compute returns backward: $G_T = 0$ and $G_t = R_{t+1} + \gamma G_{t+1}$.
3. For each state $s$ visited, let $t_s$ be its first visit in this episode.
4. Increase $N(s)$ by one and update

$$V(s) \leftarrow V(s) + \frac{1}{N(s)}(G_{t_s} - V(s)).$$

Each episode contributes at most one return per state. Unvisited entries stay unchanged.

Sutton and Barto, Sec. 5.1.

# Monte Carlo concentration

For $n$ independent complete returns $G_1, \dots, G_n$ starting at $s$, let $\widehat{V}_n(s) = n^{-1} \sum_{i=1}^{n} G_i$. Hoeffding gives

$$\mathbb{P}\Big( \Big| \widehat{V}_n(s) - V^{\pi}(s) \Big| \geq \varepsilon \Big) \leq 2 \exp\Big( -2n\varepsilon^2 (1-\gamma)^2 \Big).$$

To obtain error at most $\varepsilon$ with probability at least $1 - \delta$, it suffices that

$$n \geq \frac{\log(2/\delta)}{2\varepsilon^2 (1-\gamma)^2}.$$

Here $n$ counts independent returns. It does not count consecutive transitions from one trajectory.

Application of Hoeffding's inequality to bounded discounted returns.

# Temporal-difference learning

# TD(0): a one-step target

After observing a transition under the fixed policy $\pi$, form

$$Y_t = R_{t+1} + \gamma V_t(S_{t+1}).$$

Update the current state:

$$V_{t+1}(S_t) = V_t(S_t) + \alpha_t(R_{t+1} + \gamma V_t(S_{t+1}) - V_t(S_t)).$$

All other entries stay unchanged. Set $V_t(\Delta) = 0$ for a terminal state.

**Temporal difference** refers to the error between the current prediction and a reward plus the next prediction.

Sutton and Barto, Sec. 6.1.

# The expected TD target

Let $\mathcal{F}_t$ contain the history through $S_t$ and the current table $V_t$, before drawing $A_t \sim \pi(\cdot \mid S_t)$.

$$\begin{aligned}\mathbb{E}[Y_t \mid \mathcal{F}_t] &= \sum_a \pi(a \mid S_t) \left[ r(S_t, a) + \gamma \sum_{s'} P(s' \mid S_t, a) V_t(s') \right] \\ &= (\mathcal{T}^\pi V_t)(S_t).\end{aligned}$$

TD samples one component of a Bellman evaluation update. The expected target equals $\mathcal{T}^\pi V_t$, while the desired fixed point is $V^\pi$.

# Why bootstrapping can work

The target uses a current estimate of the future value. This is **bootstrapping**.

Its conditional mean generally differs from $V^\pi(S_t)$, but the whole Bellman map satisfies

$$\|\mathcal{T}^\pi V_t - V^\pi\|_\infty = \|\mathcal{T}^\pi V_t - \mathcal{T}^\pi V^\pi\|_\infty \leq \gamma \|V_t - V^\pi\|_\infty .$$

Thus exact Bellman updates approach the correct fixed point. TD replaces exact updates by noisy updates of visited coordinates.

Contraction alone does not show that every noisy TD step reduces the error. Step sizes and repeated visits control the noise.

# Online TD on the same episode

Use $\gamma = 0.9$, $\alpha = 1/2$, and $V_0(s) = V_0(w) = 0$.

| Observed transition | Target at that time | Updated entry |
|---|---|---|
| $(s, \text{wait}, 0, w)$ | $0 + 0.9 \cdot 0 = 0$ | $V(s) = 0$ |
| $(w, \text{finish}, 1, \Delta)$ | $1 + 0.9 \cdot 0 = 1$ | $V(w) = 0.5$ |

On the first transition of the next identical episode,

$$V(s) \leftarrow 0 + \tfrac{1}{2}\big(0 + 0.9 \cdot 0.5 - 0\big) = 0.225.$$

The later reward reaches earlier states through repeated bootstrapped updates.

# Monte Carlo and TD targets

| | Monte Carlo | TD(0) |
|---|---|---|
| Target | $G_t$ | $R_{t+1} + \gamma V_t(S_{t+1})$ |
| When available | After the remaining return | After one transition |
| Uses current estimates | No | Yes |
| Expected target | $V^\pi(S_t)$ | $(\mathcal{T}^\pi V_t)(S_t)$ |
| Needs a complete return | Yes | No |

Same policy $\pi$, different prediction targets.

Sutton and Barto, Sec. 6.2.

# Bounded values and local step sizes

Initialize $0 \le V_0(s) \le V_{\max}$ and choose $0 \le \alpha_t \le 1$. Then

$$0 \le R_{t+1} + \gamma V_t(S_{t+1}) \le 1 + \gamma V_{\max} = V_{\max}.$$

The new entry is a convex combination of the old entry and the target, so $0 \le V_t(s) \le V_{\max}$ for every $t$.

On the $k$th visit to state $s$, one possible choice is $\alpha_k(s) = 1/k$:

$$\sum_{k=1}^{\infty} \alpha_k(s) = \infty, \qquad \sum_{k=1}^{\infty} \alpha_k(s)^2 < \infty.$$

The visit count is local to $s$.

# Tabular TD convergence

For a finite discounted MDP and a fixed stationary policy $\pi$, assume:

- rewards are bounded and samples follow the MDP under $\pi$;
- every nonterminal state is visited infinitely often;
- per-state step sizes use past observations, lie in $[0, 1]$, and satisfy $\sum_k \alpha_k(s) = \infty$, $\sum_k \alpha_k(s)^2 < \infty$.

Then, for bounded initial values and zero terminal values,

$$V_t \xrightarrow{\text{a.s.}} V^\pi.$$

This is a stochastic approximation theorem. A constant step size does not satisfy its assumptions.

Tsitsiklis (1994), asynchronous stochastic approximation. Sutton and Barto, Ch. 6.

# The stochastic approximation structure

Write the TD target as its conditional mean plus noise:

$$M_{t+1} = Y_t - (\mathcal{T}^\pi V_t)(S_t), \qquad \mathbb{E}[M_{t+1} \mid \mathcal{F}_t] = 0.$$

Then the visited entry obeys

$$V_{t+1}(S_t) = V_t(S_t) + \alpha_t\big((\mathcal{T}^\pi V_t)(S_t) - V_t(S_t) + M_{t+1}\big).$$

The Bellman contraction supplies a stable fixed point. Repeated visits update all coordinates. Diminishing step sizes control accumulated centered noise.

The noise is conditionally centered even though successive transitions are dependent.

Tsitsiklis (1994), asynchronous stochastic approximation.

# Function approximation

# Value functions with shared parameters

A table stores a separate number for every state. For images, robot configurations, or continuous states, visiting each state many times is impractical.

Instead, represent the value by a function

$$V_w : \mathcal{S} \to \mathbb{R}, \qquad w \in \mathbb{R}^d.$$

| Representation | Parameters | How states share information |
|---|---|---|
| Table | One entry per state | Each entry is separate |
| Linear model | Feature weights | Shared features |
| Neural network | Network weights | Learned representations |

Sutton and Barto, Ch. 9.

# Linear features and the tabular special case

Choose a feature vector $\phi(s) \in \mathbb{R}^d$ and let

$$V_w(s) = w^\top \phi(s).$$

A feature can describe position, remaining resources, or a learned property of an observation. The same weight vector affects many states.

For a finite state space $\{s_1, \ldots, s_m\}$, choose one-hot features:

$$\phi(s_i) = e_i \in \mathbb{R}^m, \qquad V_w(s_i) = w_i.$$

This recovers the table. With fewer shared features, the function class restricts which value vectors can be represented.

Sutton and Barto, Secs. 9.4–9.5.

# Monte Carlo prediction becomes regression

Fix $\pi$. Sample starting states $S \sim \nu$, then roll out $\pi$ and observe returns $G$, with $\mathbb{E}[G \mid S = s] = V^\pi(s)$.

$$\mathcal{L}(w) = \frac{1}{2}\mathbb{E}[(V_w(S) - G)^2].$$

Conditioning on $S$ gives

$$\mathbb{E}[(V_w(S) - G)^2 \mid S] = (V_w(S) - V^\pi(S))^2 + \mathrm{Var}(G \mid S).$$

The variance term does not depend on $w$. Minimizing this loss fits $V^\pi$ in the state distribution $\nu$.

For sampled pairs $(s_i, G_i)$, use $\widehat{\mathcal{L}}(w) = \frac{1}{2n}\sum_i (V_w(s_i) - G_i)^2$.

Sutton and Barto, Sec. 9.3. Independent rollouts give the usual regression sampling model.

# A sample update can change several state values

A stochastic gradient step for the Monte Carlo regression loss is

$$w_{t+1} = w_t + \alpha_t(G_t - V_{w_t}(S_t))\nabla_w V_{w_t}(S_t).$$

For a linear model, $\nabla_w V_w(s) = \phi(s)$.

For example, let $w_0 = (0, 0)$, $\phi(s) = (1, 1)$, $\phi(z) = (1, 0)$. Observe a return $G = 0.9$ at $s$ and use $\alpha = 1/2$:

$$w_1 = (0, 0) + \tfrac{1}{2}(0.9)(1, 1) = (0.45, 0.45).$$

Then $V_{w_1}(s) = 0.9$ and $V_{w_1}(z) = 0.45$.

The observation at $s$ also changes the prediction at $z$ through the shared weights. The features determine how this generalization occurs.

Sutton and Barto, Sec. 9.3. Numerical illustration uses one observed return.

# TD with function approximation

After a transition under the fixed policy $\pi$, define

$$\delta_t = R_{t+1} + \gamma V_{w_t}(S_{t+1}) - V_{w_t}(S_t).$$

The TD update becomes

$$w_{t+1} = w_t + \alpha_t \delta_t \nabla_w V_{w_t}(S_t).$$

For a linear value function,

$$w_{t+1} = w_t + \alpha_t \big(R_{t+1} + \gamma w_t^\top \phi(S_{t+1}) - w_t^\top \phi(S_t)\big)\phi(S_t).$$

At a terminal next state, use zero continuation value. The update can now change predictions at states other than $S_t$.

Sutton and Barto, Sec. 9.3.

# Why this is a semi-gradient update

Hold the sampled target fixed at its current value $Y_t = R_{t+1} + \gamma V_{w_t}(S_{t+1})$. Differentiate the loss only with respect to the current-state prediction:

$$\nabla_w \frac{1}{2}(V_w(S_t) - Y_t)^2 \Big|_{w=w_t} = -\delta_t \nabla_w V_{w_t}(S_t).$$

Differentiating through both predictions would instead produce

$$\nabla_w \frac{1}{2}\big(R_{t+1} + \gamma V_w(S_{t+1}) - V_w(S_t)\big)^2 \Big|_{w=w_t} = \delta_t(\gamma \nabla_w V_{w_t}(S_{t+1}) - \nabla_w V_{w_t}(S_t)).$$

TD uses the first expression. In a neural-network implementation, the bootstrap target has no gradient during this update.

Sutton and Barto, Sec. 9.3. The target changes again on later updates.

# A Bellman update followed by a fit

Consider a batch version of policy evaluation. Freeze the current value function $V_{w_k}$ and compute its Bellman targets.

$$w_{k+1} \in \operatorname*{argmin}_{w} \mathbb{E}_{S \sim \nu}\left[\left(V_w(S) - (\mathcal{T}^\pi V_{w_k})(S)\right)^2\right].$$

Equivalently, write

$$V_{w_{k+1}} = \Pi_\nu \mathcal{T}^\pi V_{w_k},$$

where $\Pi_\nu$ denotes a best fit in the chosen function class under $\nu$.

Observed one-step targets can estimate this regression. An exact Bellman update may leave the function class, so the fit introduces another operation.

Sutton and Barto, Ch. 9. Fitted policy evaluation is distinct from an individual online TD step.

# Approximation changes the stability question

For the full value vector, planning gave

$$\left\|\mathcal{T}^{\pi}u - \mathcal{T}^{\pi}v\right\|_{\infty} \leq \gamma \left\|u - v\right\|_{\infty}.$$

With function approximation, the iteration also applies $\Pi_{\nu}$. This fitting step need not preserve contraction in the sup norm.

Even if the true value lies in the class, arbitrary fitted iterations need not converge to it.

On-policy linear TD has convergence results under appropriate visitation, feature, and step-size conditions. Nonlinear approximation and off-policy sampling require separate analysis.

Sutton and Barto, Chs. 9 and 11.

# A two-state fitted-iteration example

Fix a policy with one action per state. Both states transition to $v$, all rewards are zero, and $\gamma = 0.9$.

| State | Next state | Reward | Feature |
|---|---|---|---|
| $u$ | $v$ | $0$ | $\phi(u) = 1$ |
| $v$ | $v$ | $0$ | $\phi(v) = 2$ |

Use $V_w(u) = w$, $V_w(v) = 2w$. The true value $V^\pi = 0$ is represented by $w = 0$.

Fit the two states with equal weights, $\nu(u) = \nu(v) = 1/2$. From the old parameter $w_k$, both Bellman targets equal $2\gamma w_k$.

Adapted from Nan Jiang, Value Prediction with Function Approximation, two-state example.

# The fit can amplify the parameter

The next parameter minimizes

$$\frac{1}{2}(w - 2\gamma w_k)^2 + \frac{1}{2}(2w - 2\gamma w_k)^2.$$

Setting its derivative to zero gives

$$(w - 2\gamma w_k) + 2(2w - 2\gamma w_k) = 0, \qquad w_{k+1} = \frac{6\gamma}{5} w_k = 1.08 w_k.$$

For any $w_0 \neq 0$, this fitted iteration diverges, even though the exact Bellman operator is a contraction and $V^\pi$ is representable.

The uniform fitting distribution is not this chain's stationary distribution. This is a counterexample for the stated fitted iteration, not for all on-policy linear TD.

For control, we will similarly approximate action values by $Q_w(s, a)$.

Direct calculation for the preceding example. Nan Jiang, CS 443 function approximation slides.

# Control from sampled transitions

# Policy evaluation and policy improvement

Control combines two operations:

$$\text{estimate } Q^\pi \qquad \text{and} \qquad \pi_{\text{new}}(s) \in \arg\max_a Q^\pi(s, a).$$

With a known model, policy iteration performs these operations exactly. With samples, evaluation and improvement can proceed together. This is **generalized policy iteration**.

In the cash-or-wait example,

$$Q^\pi(s, \text{cash}) = 0.6, \qquad Q^\pi(s, \text{wait}) = 0.9$$

when $\pi$ chooses finish at $w$. An improved policy chooses wait.

Sutton and Barto, Sec. 4.6 and Ch. 6.

# SARSA: learning from the next action taken

For a fixed policy $\pi$, the next action also follows $\pi$. Observe the five quantities

$$(S_t, A_t, R_{t+1}, S_{t+1}, A_{t+1}).$$

The **SARSA** update is

$$Q_{t+1}(S_t, A_t) = Q_t(S_t, A_t) + \alpha_t[R_{t+1} + \gamma Q_t(S_{t+1}, A_{t+1}) - Q_t(S_t, A_t)].$$

All other entries stay unchanged. A terminal next state contributes zero. For a fixed $\pi$, this target samples the Bellman expectation equation for $Q^\pi$.

For control, choose actions using the current table and update that table as experience arrives.

Sutton and Barto, Sec. 6.4.

# Exploration during policy improvement

Let $a_t^*(s) \in \arg\max_a Q_t(s,a)$ be a chosen greedy action. An $\varepsilon_t$-greedy rule uses

$$b_t(a \mid s) = (1-\varepsilon_t)\mathbf{1}_{\{a=a_t^*(s)\}} + \frac{\varepsilon_t}{|\mathcal{A}|}.$$

SARSA with this behavior updates the value of the exploratory decisions being made. Its target policy changes as the table and exploration rate change.

Positive action probabilities help only at states that are reached. Optimal-control convergence needs sufficient visits and appropriate learning rates, together with a suitable policy-improvement schedule.

Sutton and Barto, Sec. 6.4.

# Expected SARSA

Instead of sampling a next action, average its value under a chosen policy $\pi$:

$$Y_t^\pi = R_{t+1} + \gamma \sum_{a' \in \mathcal{A}} \pi(a' \mid S_{t+1}) Q_t(S_{t+1}, a').$$

Then update

$$Q_{t+1}(S_t, A_t) = Q_t(S_t, A_t) + \alpha_t (Y_t^\pi - Q_t(S_t, A_t)).$$

The current pair $(S_t, A_t)$ can come from a different behavior policy. The target explicitly supplies the action distribution at the next state.

Choosing a greedy next-state policy makes the sum a maximum.

# Q-learning

Q-learning uses the optimality target

$$Y_t^Q = R_{t+1} + \gamma \max_{a'} Q_t(S_{t+1}, a').$$

Only the observed pair changes:

$$Q_{t+1}(S_t, A_t) = Q_t(S_t, A_t) + \alpha_t(Y_t^Q - Q_t(S_t, A_t)).$$

The behavior policy supplies experience. The maximum evaluates a greedy next action, even if the behavior policy will take a different action.

Use zero future value when $S_{t+1}$ is terminal.

Sutton and Barto, Sec. 6.5. Watkins and Dayan (1992).

# The conditional mean of a Q-learning target

Let $\mathcal{G}_t$ contain the history, the current table, and the selected action $A_t$, before the environment returns the next reward and state.

$$\begin{aligned}\mathbb{E}[Y_t^Q \mid \mathcal{G}_t] &= r(S_t, A_t) + \gamma \sum_{s'} P(s' \mid S_t, A_t) \max_{a'} Q_t(s', a') \\ &= (\mathcal{T}_Q Q_t)(S_t, A_t).\end{aligned}$$

The mean update targets the optimal Bellman operator. The behavior policy determines which coordinate receives that update.

This separation makes one-step Q-learning an off-policy control method.

Nan Jiang, Algorithms for control.

# Three targets for the same transition

Suppose $R_{t+1} = 0$, $\gamma = 0.9$, and at the observed next state $u$,

$$Q_t(u, a_1) = 0.8, \qquad Q_t(u, a_2) = 0.2.$$

The behavior policy chooses $a_1$ with probability $0.75$ and $a_2$ with probability $0.25$. Suppose the sampled next action is $a_2$.

| Method | Target on this observation |
|---|---|
| SARSA | $0.9(0.2) = 0.18$ |
| Expected SARSA under $b$ | $0.9(0.75 \cdot 0.8 + 0.25 \cdot 0.2) = 0.585$ |
| Q-learning | $0.9 \max(0.8, 0.2) = 0.72$ |

The targets describe different choices of future action.

# Tabular Q-learning convergence

Assume a finite discounted MDP, bounded rewards, and samples with the correct conditional reward–transition law. Choose each step size before observing that transition.

Every nonterminal pair must be updated infinitely often, with local step sizes

$$0 \le \alpha_k(s,a) \le 1, \qquad \sum_k \alpha_k(s,a) = \infty, \qquad \sum_k \alpha_k(s,a)^2 < \infty.$$

For bounded initial tables and terminal entries kept at zero,

$$Q_t \xrightarrow{\text{a.s.}} Q^*.$$

The contraction of $\mathcal{T}_Q$, bounded sampled noise, and repeated coordinate updates give the stochastic-approximation argument.

Watkins and Dayan (1992). Tsitsiklis (1994), asynchronous stochastic approximation.

# Control with an approximate action-value function

Represent values by $Q_w(s, a)$, with parameters $w$. Given a transition and a fixed target value $y_t$, use

$$w_{t+1} = w_t + \alpha_t(y_t - Q_{w_t}(S_t, A_t))\nabla_w Q_{w_t}(S_t, A_t).$$

Possible targets include

$$y_t^{\text{SARSA}} = R_{t+1} + \gamma Q_{w_t}(S_{t+1}, A_{t+1}),$$
$$y_t^{Q} = R_{t+1} + \gamma \max_{a'} Q_{w_t}(S_{t+1}, a').$$

The target is held fixed during this gradient step. Shared parameters can change estimates at many pairs simultaneously. Set the future contribution to zero at a terminal next state.

Sutton and Barto, Ch. 10.

# Fitted Q-iteration

Given a dataset $\mathcal{D} = \{(s_i, a_i, r_i, s_i')\}_{i=1}^N$ and the old estimate $Q_k$, form fixed regression targets

$$y_i = r_i + \gamma \max_{a'} Q_k(s_i', a').$$

Use $y_i = r_i$ when $s_i'$ is terminal. Fit the next function in a chosen class $\mathcal{Q}$:

$$Q_{k+1} \in \arg\min_{f \in \mathcal{Q}} \frac{1}{N} \sum_{i=1}^{N} \big(f(s_i, a_i) - y_i\big)^2.$$

Repeat target construction and regression. Finite data, restricted function classes, and incomplete optimization introduce errors beyond those of exact value iteration.

Nan Jiang, Algorithms for control.

# A deep Q-network

A **deep Q-network (DQN)** represents $Q_w$ by a neural network. Two components help manage changing targets and correlated observations.

**Experience replay:** store transitions and sample minibatches from that collection.

**Target network:** hold parameters $\bar{w}$ fixed for several optimization steps:

$$y_i = r_i + \gamma(1 - d_i) \max_{a'} Q_{\bar{w}}(s_i', a'), \qquad L(w) = \frac{1}{2N} \sum_{i=1}^{N} \big(Q_w(s_i, a_i) - y_i\big)^2.$$

Here $d_i = 1$ for a terminal transition and $d_i = 0$ otherwise. Periodically copy the current parameters into the target network, $\bar{w} \leftarrow w$.

These design choices do not inherit the finite-table convergence theorem.

Mnih et al. (2015), Human-level control through deep reinforcement learning. Simplified update.

# Off-policy learning and evaluation

# A target policy and a behavior policy

Fix a target policy $\pi$ to evaluate. Let $b$ generate the observed actions.

| Object | Role |
|---|---|
| Target $\pi(a \mid s)$ | Specifies the return we want to estimate |
| Behavior $b(a \mid s)$ | Specifies how the data are collected |

Importance sampling needs action coverage:

$$\pi(a \mid s) > 0 \quad \Longrightarrow \quad b(a \mid s) > 0.$$

This condition applies at states relevant to the target policy. It does not ensure that those states occur in the available data.

Sutton and Barto, Sec. 5.5.

# One-step importance sampling

Define the action ratio

$$\rho_t = \frac{\pi(A_t \mid S_t)}{b(A_t \mid S_t)}.$$

At a visited state $s$, a bounded fixed function $f$ satisfies

$$\begin{aligned}\mathbb{E}_b[\rho_t f(A_t, R_{t+1}, S_{t+1}) \mid S_t = s] &= \sum_{a:\, b(a|s)>0} b(a \mid s) \frac{\pi(a \mid s)}{b(a \mid s)} \mathbb{E}[f(a, R, S') \mid s, a] \\ &= \mathbb{E}_\pi[f(A_t, R_{t+1}, S_{t+1}) \mid S_t = s].\end{aligned}$$

The ratio changes the action distribution at the current state. The conditional environment law is the same under both policies.

# Off-policy TD at a visited state

Let $\mathcal{F}_t$ contain the history and table $V_t$ before drawing $A_t$ from $b$. The importance-sampling identity gives

$$\mathbb{E}_b\big[\rho_t(R_{t+1} + \gamma V_t(S_{t+1}) - V_t(S_t)) \,\big|\, \mathcal{F}_t\big] = (\mathcal{T}^\pi V_t)(S_t) - V_t(S_t).$$

This suggests the update

$$V_{t+1}(S_t) = V_t(S_t) + \alpha_t \rho_t(R_{t+1} + \gamma V_t(S_{t+1}) - V_t(S_t)).$$

Other entries stay unchanged. Action correction does not change how often the behavior visits each state. With function approximation, that state weighting still matters.

Sutton and Barto, Sec. 7.3 and Ch. 11.

# Importance sampling for a complete trajectory

Consider $H$ decisions, with the same initial state law $\mu$ and environment under $\pi$ and $b$. For an observed trajectory $\tau$,

$$W_H(\tau) = \frac{p_\pi(\tau)}{p_b(\tau)} = \prod_{t=0}^{H-1} \frac{\pi(A_t \mid S_t)}{b(A_t \mid S_t)}.$$

The joint reward–transition factors cancel. No reward–next-state independence is needed.

For $G_H = \sum_{t=0}^{H-1} \gamma^t R_{t+1}$,

$$J_H(\pi) = \mathbb{E}_\pi[G_H] = \mathbb{E}_b[W_H G_H].$$

Thus $N$ independent behavior trajectories give the unbiased estimate

$$\widehat{J}_H(\pi) = \frac{1}{N} \sum_{i=1}^{N} W_H(\tau_i) G_H(\tau_i).$$

Sutton and Barto, Sec. 5.5.

# The variance can grow with the horizon

There are two actions at each step. The target always chooses $a_1$; the behavior chooses either action with probability $1/2$.

A length-$H$ trajectory has

$$W_H = \begin{cases} 2^H, & \text{all } H \text{ actions equal } a_1, \\ 0, & \text{otherwise.} \end{cases}$$

The nonzero case has probability $2^{-H}$, so

$$\mathbb{E}_b[W_H] = 1, \qquad \mathrm{Var}_b(W_H) = 2^H - 1.$$

At $H = 10$, only one trajectory in $1024$ matches on average. Unbiasedness alone does not make the estimator accurate with a modest sample.

# Off-policy bootstrapping with function approximation

Three ingredients occur together in many useful algorithms:

1. **Function approximation:** one parameter update changes many predictions.
2. **Bootstrapping:** targets depend on current value estimates.
3. **Off-policy data:** sampled states and actions follow another policy.

Their combination can make value updates unstable, even for linear approximators. Sutton and Barto call this combination the **deadly triad**.

An unbiased conditional action correction does not establish a global contraction for the parameter update.

Sutton and Barto, Ch. 11.

# Offline data and replay

**On-policy/off-policy** describes the relationship between the data policy and the policy being evaluated or improved.

**Online/offline** describes whether the learner can collect more experience. Online Q-learning can be off-policy while it actively interacts with the environment. An offline dataset can also evaluate its own logging policy.

Replaying a finite dataset reuses information; it does not add new evidence about unobserved state–action pairs.

With a finite table and suitable updates, repeated replay targets the dataset's empirical transition law. Accuracy for the true environment still depends on data coverage and estimation error.

Nan Jiang, Algorithms for control, experience replay.

# Policy gradients

# Optimizing the policy directly

A value-based method learns action values and extracts a policy. We can instead choose a differentiable policy class $\{\pi_\theta : \theta \in \mathbb{R}^d\}$.

$$J_\mu(\theta) = \mathbb{E}_{\substack{S_0 \sim \mu \\ \pi_\theta}} \left[ \sum_{t=0}^{\infty} \gamma^t R_{t+1} \right], \qquad \theta \leftarrow \theta + \eta \widehat{\nabla J_\mu(\theta)}.$$

The initial distribution $\mu$ and the environment do not depend on $\theta$.

Assume $\pi_\theta(a \mid s) > 0$ and smooth dependence on $\theta$. Bounded policy scores justify the expectation and limit manipulations below.

The update uses gradient **ascent**: larger return is better. Direct policy parameterization also accommodates continuous action spaces.

Sutton and Barto, Ch. 13. Nan Jiang, Policy Gradient.

# A softmax policy and its score

For state–action features $\phi(s,a) \in \mathbb{R}^d$, define

$$\pi_\theta(a \mid s) = \frac{\exp(\theta^\top \phi(s,a))}{\sum_{b \in \mathcal{A}} \exp(\theta^\top \phi(s,b))}.$$

Differentiating its log gives the **policy score**:

$$g_\theta(s,a) := \nabla_\theta \log \pi_\theta(a \mid s) = \phi(s,a) - \sum_b \pi_\theta(b \mid s)\phi(s,b).$$

The score changes the relative probability of the chosen action. For every state,

$$\mathbb{E}_{A \sim \pi_\theta(\cdot|s)}[g_\theta(s,A)] = 0.$$

Linear softmax example. The same log-probability derivative applies to differentiable neural policies.

# The probability of a sampled trajectory

First consider $H$ decisions and

$$\tau = (S_0, A_0, R_1, S_1, \ldots, A_{H-1}, R_H, S_H).$$

Let $K(r, s' \mid s, a)$ denote the joint reward and next-state mass or density. Then

$$p_\theta(\tau) = \mu(S_0) \prod_{t=0}^{H-1} \pi_\theta(A_t \mid S_t)\, K(R_{t+1}, S_{t+1} \mid S_t, A_t).$$

Only the policy factors depend on $\theta$. On the trajectory support,

$$\nabla_\theta \log p_\theta(\tau) = \sum_{t=0}^{H-1} \nabla_\theta \log \pi_\theta(A_t \mid S_t).$$

No independence between reward and next state is needed.

Likelihood-ratio derivation. Nan Jiang, Policy Gradient.

# The likelihood-ratio identity

Write $Z_H(\tau) = \sum_{k=0}^{H-1} \gamma^k R_{k+1}$ and $J_H(\theta) = \mathbb{E}_\theta[Z_H]$. With $g_t = \nabla_\theta \log \pi_\theta(A_t \mid S_t)$,

$$\begin{aligned}\nabla_\theta J_H(\theta) &= \int Z_H(\tau)\, \nabla_\theta p_\theta(\tau)\, d\tau \\ &= \mathbb{E}_\theta[Z_H(\tau) \nabla_\theta \log p_\theta(\tau)] \\ &= \mathbb{E}_\theta\left[ Z_H(\tau) \sum_{t=0}^{H-1} g_t \right].\end{aligned}$$

We can estimate the gradient from sampled rewards and policy scores. We do not need to differentiate, or know, the environment kernel.

The score-function identity under differentiation and expectation interchange.

# Causality gives reward-to-go

Let $\mathcal{H}_t$ contain the history before choosing $A_t$. Past rewards are $\mathcal{H}_t$-measurable, and $\mathbb{E}[g_t \mid \mathcal{H}_t] = 0$. Therefore

$$\mathbb{E}\left[ g_t \sum_{k=0}^{t-1} \gamma^k R_{k+1} \right] = 0.$$

Define the return measured from time $t$ by

$$G_t^{(H)} = \sum_{k=t}^{H-1} \gamma^{k-t} R_{k+1}.$$

Removing those past rewards from the likelihood-ratio estimator gives

$$\boxed{\nabla_\theta J_H(\theta) = \mathbb{E}_\theta \left[ \sum_{t=0}^{H-1} \gamma^t G_t^{(H)} g_t \right].}$$

The factor $\gamma^t$ accounts for when the action occurs in the original objective.

# From sampled returns to action values

For the infinite-horizon problem, let $G_t := \lim_{H\to\infty} G_t^{(H)}$.

$$\mathbb{E}_\theta[G_t \mid S_t = s, A_t = a] = Q^{\pi_\theta}(s, a).$$

Taking the discounted limit of the reward-to-go identity and conditioning on $(S_t, A_t)$ gives

$$\nabla_\theta J_\mu(\theta) = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_\theta[Q^{\pi_\theta}(S_t, A_t)\, g_\theta(S_t, A_t)]\,.$$

This derivation already accounts for the policy's effect on future state distributions. It does not hold those distributions fixed while differentiating.

Bounded rewards and bounded policy scores justify passing to the discounted limit.

# The policy-gradient theorem

Define the normalized discounted state distribution

$$d_\mu^{\pi_\theta}(s) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t \mathbb{P}_{\mu,\pi_\theta}(S_t = s).$$

Its entries sum to one. Regrouping the previous time-indexed identity yields

$$\boxed{\nabla_\theta J_\mu(\theta) = \frac{1}{1-\gamma}\mathbb{E}_{\substack{S\sim d_\mu^{\pi_\theta} \\ A\sim\pi_\theta(\cdot|S)}}\left[Q^{\pi_\theta}(S,A)\,\nabla_\theta \log \pi_\theta(A \mid S)\right].}$$

The factor $1/(1-\gamma)$ matches the normalized state distribution and the unnormalized discounted-return objective.

Sutton, McAllester, Singh, and Mansour (1999).

# REINFORCE with discounted returns

**Data**: on-policy rollouts reaching $\Delta$ at finite $T$

$\theta$ fixed within each rollout. $\Delta$: zero-reward absorbing state

1. Compute returns backward: $G_T = 0$ and $G_t = R_{t+1} + \gamma G_{t+1}$.
2. Form the gradient estimate

$$\widehat{g} = \sum_{t=0}^{T-1} \gamma^t G_t \nabla_\theta \log \pi_\theta(A_t \mid S_t).$$

3. Update $\theta \leftarrow \theta + \eta \widehat{g}$. Start a new rollout.

Complete rollouts, stated regularity: $\mathbb{E}[\widehat{g}] = \nabla J_\mu(\theta)$.

Early cutoff: discounted tail omitted.

REINFORCE, in the discounted-return convention used throughout this lecture.

# A policy gradient on the cash-or-wait example

Let the initial state be $s$, and let

$$p_\theta = \pi_\theta(\text{wait} \mid s) = \frac{e^\theta}{1 + e^\theta}.$$

After wait, every action at $w$ finishes with reward $1$. With $\gamma = 0.9$, the two complete returns are $0.6$ and $0.9$.

$$J(\theta) = 0.6(1 - p_\theta) + 0.9p_\theta, \qquad J'(\theta) = 0.3p_\theta(1 - p_\theta).$$

At $\theta = 0$, $p_\theta = 1/2$ and $J'(\theta) = 0.075 > 0$.

Gradient ascent increases the probability of wait. REINFORCE reaches this conclusion using sampled returns rather than the model.

# A state baseline leaves the mean gradient unchanged

For a fixed action-independent baseline $b(s)$,

$$\begin{aligned}\mathbb{E}_{A\sim\pi_\theta(\cdot|s)}[b(s)\nabla_\theta \log \pi_\theta(A \mid s)] &= b(s)\sum_a \nabla_\theta \pi_\theta(a \mid s) \\ &= b(s)\nabla_\theta 1 = 0.\end{aligned}$$

Hence we may replace $G_t$ by $G_t - b(S_t)$ in REINFORCE:

$$\widehat{g}_b = \sum_{t=0}^{T-1} \gamma^t (G_t - b(S_t))\nabla_\theta \log \pi_\theta(A_t \mid S_t).$$

An appropriate baseline can reduce variance. For example, fit it from prior data and hold it fixed while collecting the actor's batch. Do not differentiate the baseline in the actor's gradient step.

Sutton and Barto, Sec. 13.4. Nan Jiang, Policy Gradient.

# Advantages compare an action with the current policy

Choose the baseline $b(s) = V^\pi(s)$ and define

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

Then

$$\nabla_\theta J_\mu(\theta) = \frac{1}{1-\gamma} \mathbb{E}_{\substack{S \sim d_\mu^{\pi_\theta} \\ A \sim \pi_\theta(\cdot|S)}} \left[ A^{\pi_\theta}(S,A) \nabla_\theta \log \pi_\theta(A \mid S) \right].$$

An action with positive advantage has higher expected return than the policy's average action at that state.

On the running example at $p_\theta = 1/2$, $V^\pi(s) = 0.75$: cash has advantage $-0.15$, while wait has advantage $0.15$.

# A critic supplies a one-step advantage estimate

Let $V_w$ approximate the current policy's value and form

$$\delta_t = R_{t+1} + \gamma V_w(S_{t+1}) - V_w(S_t).$$

If $V_w = V^\pi$, the Bellman equation gives

$$\begin{aligned}\mathbb{E}[\delta_t \mid S_t = s, A_t = a] &= r(s,a) + \gamma \sum_{s'} P(s' \mid s,a) V^\pi(s') - V^\pi(s) \\ &= A^\pi(s,a).\end{aligned}$$

The critic replaces a complete return by a bootstrapped estimate. With an approximate critic, this conditional mean can differ from the true advantage, so the actor update can be biased.

Sutton and Barto, Sec. 13.5.

# An actor–critic update

Use separate parameters: actor $\pi_\theta$ and critic $V_w$. For an on-policy transition at time $t$ in an episode, compute

$$\delta_t = R_{t+1} + \gamma V_w(S_{t+1}) - V_w(S_t).$$

Using the current parameters on both right-hand sides,

$$w \leftarrow w + \alpha\, \delta_t \nabla_w V_w(S_t),$$
$$\theta \leftarrow \theta + \eta\, \gamma^t \delta_t \nabla_\theta \log \pi_\theta(A_t \mid S_t).$$

The critic uses a TD semi-gradient. The actor treats $\delta_t$ as a target. At a terminal next state, set its continuation value to zero.

The $\gamma^t$ weight matches $J_\mu$. With changing parameters and an approximate critic, this is an algorithmic update rather than an exact-gradient guarantee.

# Policy optimization in the larger RL picture

| Method | Object learned | Policy update |
|---|---|---|
| Value-based control | Action values $Q$ | Greedy or exploratory action selection |
| REINFORCE | Policy $\pi_\theta$ | Scores weighted by complete returns |
| Actor–critic | Policy and value estimates | Scores weighted by estimated advantages |

Policy gradients connect sequential decisions to stochastic optimization. Sampling noise, representation error, and changing data distributions still matter.

A differentiable policy objective generally remains nonconcave. The gradient formula alone does not guarantee global optimality.

# Summary

# The route through reinforcement learning

| Stage | Main question | Tools |
|---|---|---|
| Model | What is a decision problem? | MDP formulation |
| Values | How good is a policy? Which is best? | Bellman equations |
| Planning | How can we compute a policy? | Value and policy iteration |
| Prediction | How can we learn values from data? | MC, TD, function approximation |
| Control | How can data improve decisions? | SARSA, Q-learning, importance sampling |
| Policy optimization | How can we update a policy directly? | Policy gradients, actor–critic |

# References for the mathematical development

**Models and planning**

Puterman, *Markov Decision Processes* (1994), Chapters 2, 4–6.

**Learning and approximation**

Sutton and Barto, *Reinforcement Learning: An Introduction*, second edition (2018), Chapters 3–6, 9–11, and 13.

**Course materials**

Nan Jiang, CS 443 and CS 542: MDPs, value prediction, function approximation, control, importance sampling, policy gradient.

**Convergence and policy gradients**

Watkins and Dayan (1992). Tsitsiklis (1994). Sutton, McAllester, Singh, and Mansour (1999). Individual slides link the primary sources for applications and specific results.