Lecture 3: Bellman Equation and Value Function Iteration

Yasuyuki Sawada, Yaolang Zhong

University of Tokyo

yaolang.zhong@e.u-tokyo.ac.jp

October 22, 2025

Recap: Markov Decision Process (MDP)

- An MDP is defined by the tuple

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

  where:

  - State space $\mathcal{S}$: Possible system states.
  - Action space $\mathcal{A}(s)$: Feasible actions when in state s.
  - Transition kernel $P_t(s' \mid s, a)$: Probability of moving from s to s' given action a.
  - Reward function $r_t(s, a)$: Instantaneous payoff from taking action a in state s. (Sometimes expressed as a *cost* $-r(s, a)$.)
  - Discount factor $\beta \in (0, 1)$: Weights future rewards relative to current ones.

- Objective: Choose a policy $\pi : \mathcal{S} \to \mathcal{A}$ to maximize expected discounted rewards:

$$\mathbb{E}_\pi \left[\sum_{t=0}^{T} \beta^t r_t(s_t, a_t) + \beta^T R_T(s_T) \right].$$

Recap: Value function and Bellman equation

- Problem Specification:
  - Infinite horizon (Finite horizon problems can be solved by backward-induction)
  - Time-homogeneous: known $P(s' \mid s, a), r(s' \mid s, a)$
  - Didn't specify State space $\mathcal{S}$,Action space $\mathcal{A}(s)$ to be continuous or discrete
- Two versions of the Bellman equations:
  - Policy evaluation for a given π:

$$V^{\pi}(s) \;=\; r\big(s, \pi(s)\big) \;+\; \beta \sum_{s'} P\big(s' \mid s, \pi(s)\big)\, V^{\pi}(s'), \quad \forall s \in \mathcal{S}.$$

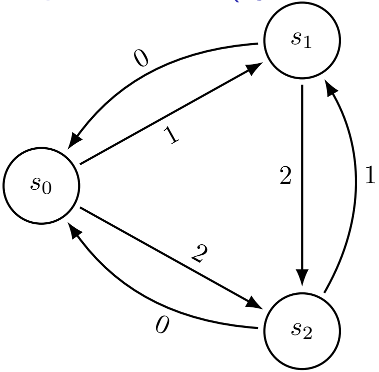
  - Optimality:

$$V^{*}(s) \;=\; \max_{a \in \mathcal{A}(s)} \Big\{ r(s,a) + \beta \sum_{s'} P(s' \mid s, a)\, V^{*}(s') \Big\}, \quad \forall s \in \mathcal{S}.$$

  where $V^{*} = V^{\pi^{*}}$ and

$$\pi^{*}(s) \in \arg\max_{a \in \mathcal{A}(s)} \Big\{ r(s,a) + \beta \sum_{s'} P(s' \mid s, a)\, V^{*}(s') \Big\}, \quad \forall s \in \mathcal{S}.$$

Example: Jacobi (synchronous) Policy Evaluation



$$\pi = \begin{array}{c|ccc} & a_0 & a_1 & a_2 \\ \hline s_0 & 0 & \frac{1}{2} & \frac{1}{2} \\ s_1 & \frac{1}{2} & 0 & \frac{1}{2} \\ s_2 & \frac{1}{2} & \frac{1}{2} & 0 \end{array}$$

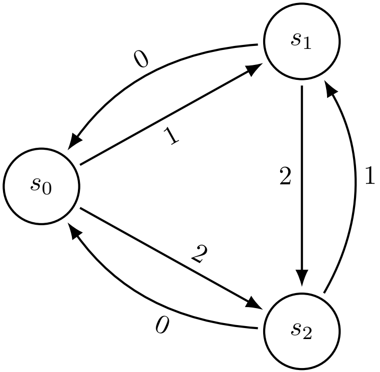
Initialization:

$$V^0(s_0) = V^0(s_1) = V^0(s_2) = 0$$

Iteration 1:

$$\begin{aligned}
V^1(s_0) &= (T^\pi V^0)(s_0) \\
&= \tfrac{1}{2}\big(1 + \beta V^0(s_1)\big) + \tfrac{1}{2}\big(2 + \beta V^0(s_2)\big) = 1.5 \\
V^1(s_1) &= (T^\pi V^0)(s_1) \\
&= \tfrac{1}{2}\big(0 + \beta V^0(s_0)\big) + \tfrac{1}{2}\big(2 + \beta V^0(s_2)\big) = 1 \\
V^1(s_2) &= (T^\pi V^0)(s_2) \\
&= \tfrac{1}{2}\big(0 + \beta V^0(s_0)\big) + \tfrac{1}{2}\big(1 + \beta V^0(s_1)\big) = 0.5
\end{aligned}$$

Example: Jacobi (synchronous) Policy Evaluation



Iteration 2:

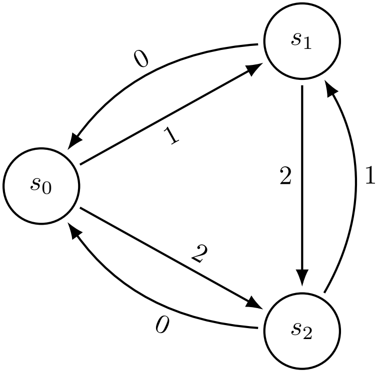
$$\begin{aligned}
V^2(s_0) &= (T^\pi V^1)(s_0) \\
&= \tfrac{1}{2}\big(1 + \beta V^1(s_1)\big) + \tfrac{1}{2}\big(2 + \beta V^1(s_2)\big) = 2.175 \\
V^2(s_1) &= (T^\pi V^1)(s_1) \\
&= \tfrac{1}{2}\big(0 + \beta V^1(s_0)\big) + \tfrac{1}{2}\big(2 + \beta V^1(s_2)\big) = 1.9 \\
V^2(s_2) &= (T^\pi V^1)(s_2) \\
&= \tfrac{1}{2}\big(0 + \beta V^1(s_0)\big) + \tfrac{1}{2}\big(1 + \beta V^1(s_1)\big) = 1.625
\end{aligned}$$

...

Iteration 89: converge to tolerance $\|\Delta V\|_\infty < 10^{-4}$

$$V^{89} \approx \big(10.344,\ 9.999,\ 9.654\big).$$

Example: Gauss–Seidel (in-place / asynchronous) Policy Evaluation



Iteration 0: $V^0(s_0) = V^0(s_1) = V^0(s_2) = 0$.
In-place sweep order $s_0 \to s_1 \to s_2$.
Iteration 1:

$$V^1(s_0) = \tfrac{1}{2}\big(1 + \beta V^0(s_1)\big) + \tfrac{1}{2}\big(2 + \beta V^0(s_2)\big) = 1.5$$

$$V^1(s_1) = \tfrac{1}{2}\big(0 + \beta V^1(s_0)\big) + \tfrac{1}{2}\big(2 + \beta V^0(s_2)\big) = 1.675$$

$$V^1(s_2) = \tfrac{1}{2}\big(0 + \beta V^1(s_0)\big) + \tfrac{1}{2}\big(1 + \beta V^1(s_1)\big) = 1.9288$$

Iteration 2:

$$V^2(s_0) = \tfrac{1}{2}\big(1 + \beta V^1(s_1)\big) + \tfrac{1}{2}\big(2 + \beta V^1(s_2)\big) = 3.1217$$

$$V^2(s_1) = \tfrac{1}{2}\big(0 + \beta V^2(s_0)\big) + \tfrac{1}{2}\big(2 + \beta V^1(s_2)\big) = 3.2727$$

$$V^2(s_2) = \tfrac{1}{2}\big(0 + \beta V^2(s_0)\big) + \tfrac{1}{2}\big(1 + \beta V^2(s_1)\big) = 3.3775$$

...
converged in 49 iterations

Policy Evaluation — Linear Algebra (building r and P)

For each state s,

$$r_s = \sum_a \pi(a|s)\, R(s,a), \qquad P_{s,s'} = \sum_a \pi(a|s)\, \mathbf{1}\{a \text{ leads to } s'\}.$$

From the diagram (outgoing rewards and next states):

$$\begin{aligned}
s_0 &: \ 1 \text{ to } s_1,\ 2 \text{ to } s_2 \quad \Rightarrow \quad r_0 = \tfrac{1+2}{2} = 1.5, \quad P_{0,1} = P_{0,2} = \tfrac{1}{2},\ P_{0,0} = 0, \\
s_1 &: \ 0 \text{ to } s_0,\ 2 \text{ to } s_2 \quad \Rightarrow \quad r_1 = \tfrac{0+2}{2} = 1, \quad P_{1,0} = P_{1,2} = \tfrac{1}{2},\ P_{1,1} = 0, \\
s_2 &: \ 0 \text{ to } s_0,\ 1 \text{ to } s_1 \quad \Rightarrow \quad r_2 = \tfrac{0+1}{2} = 0.5, \quad P_{2,0} = P_{2,1} = \tfrac{1}{2},\ P_{2,2} = 0.
\end{aligned}$$

Therefore

$$r = \begin{bmatrix} 1.5 \\ 1 \\ 0.5 \end{bmatrix}, \qquad P = \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix}.$$

Policy Evaluation — linear algebra (solve and spectral radius)

Bellman (policy evaluation):

$$V^\star = r + \beta P\, V^\star \iff (I - \beta P)\, V^\star = r.$$

With $\beta = 0.9$,

$$A \equiv I - \beta P = \begin{bmatrix} 1 & -0.45 & -0.45 \\ -0.45 & 1 & -0.45 \\ -0.45 & -0.45 & 1 \end{bmatrix}, \qquad V^\star = A^{-1} r = \begin{bmatrix} \frac{300}{29} \\ 10 \\ \frac{280}{29} \end{bmatrix} \approx \begin{bmatrix} 10.3448 \\ 10.0000 \\ 9.6552 \end{bmatrix}.$$

- If $\rho(\beta P) < 1$, then $T(V) = r + \beta P V$ is a contraction: $(I - \beta P)$ is invertible, $(I - \beta P)^{-1} = \sum_{k \geqslant 0} (\beta P)^k$, and the Bellman solution is unique; value iteration converges.
- Eigenvalues scale: $\rho(\beta P) = |\beta|\, \rho(P)$.
- P is row-stochastic $\Rightarrow \rho(P) = 1$ (here $P = \frac{1}{2}(J - I)$ with eigenvalues $1, -\frac{1}{2}, -\frac{1}{2}$).
- Hence $\rho(\beta P) = \beta < 1$.

Gauss–Seidel Policy Evaluation in linear algebra form (advanced)

Start from the linear system for policy evaluation:

$$A\,V = b, \qquad A := I - \beta P, \quad b := r.$$

Split A into diagonal, strictly lower, and strictly upper parts:

$$A = D + L + U, \qquad D := \mathrm{diag}(A), \quad L := \mathrm{tril}(A, -1), \quad U := \mathrm{triu}(A, 1).$$

Jacobi (synchronous) iteration (for comparison):

$$V^{k+1} = D^{-1}\big(b - (L+U)\,V^k\big) \quad \text{(uses only old values on the RHS; parallel-friendly).}$$

Gauss–Seidel (in-place) iteration:

$$(D+L)\,V^{k+1} = U\,V^k + b \quad \Longleftrightarrow \quad V^{k+1} = (D+L)^{-1}\big(b - U\,V^k\big).$$

- GS solves a lower-triangular system each sweep (forward substitution), so rows use newest components as soon as they're available.
- Typically fewer iterations than Jacobi; depends on ordering and sparsity.

Direct policy evaluation summary

Idea: For a fixed policy π, the Bellman equation is linear:

$$V \;=\; r \;+\; \beta P V \quad \Longleftrightarrow \quad AV = b,\; A := I - \beta P,\; b := r.$$

Intuition: This “sums the whole infinite discounted future” in one go.

When to use:

- Small/medium state spaces; A not too dense.
- You will evaluate the same P, β for many different reward vectors r
- You want tight, predictable control of the residual $\|AV - b\|$.

Trade-offs:

- Setup cost and memory for factorization; sparse A can “fill in”.
- If the policy changes (so P changes), you typically must refactor.
- Handles evaluation only; the $\max$ in control (optimality) is not a single linear solve.

Iterative policy evaluation summary

Idea: Start from a guess $V^{(0)}$ and repeatedly apply the policy-Bellman map

$$\text{Jacobi:} \quad V^{(k+1)} = r + \beta P\, V^{(k)} \quad \text{(update all states from the \underline{old} vector);}$$

Intuition: "Plug the current guess back into the right-hand side" and repeat.

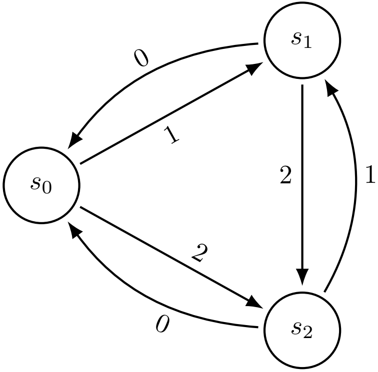
When to use:

- Large state spaces; P is sparse; you want <u>matrix-free</u> updates and low memory.
- Policies/rewards change often (easy warm starts from the last V).
- You plan to move on to controlw: the same style extends to optimality ($\max$) updates.

Trade-offs:

- Need a stopping rule ($\|V^{(k+1)} - V^{(k)}\|_\infty$ or residual); more sweeps as $\beta \to 1$.

Example: Jacobi (synchronous) Value Iteration — Optimal Policy



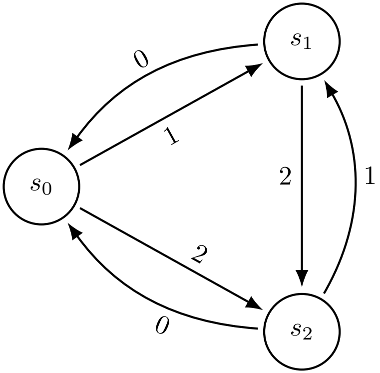
Initialization:

$$V^0(s_0) = V^0(s_1) = V^0(s_2) = 0.$$

Iteration 1:

$$\begin{aligned}
V^1(s_0) &= \max\{\, 1 + 0.9\, V^0(s_1),\ 2 + 0.9\, V^0(s_2) \,\} \\
&= \max\{1, 2\} = 2 \\
V^1(s_1) &= \max\{\, 0 + 0.9\, V^0(s_0),\ 2 + 0.9\, V^0(s_2) \,\} \\
&= \max\{0, 2\} = 2 \\
V^1(s_2) &= \max\{\, 0 + 0.9\, V^0(s_0),\ 1 + 0.9\, V^0(s_1) \,\} \\
&= \max\{0, 1\} = 1.
\end{aligned}$$

Example: Jacobi (synchronous) Value Iteration — Optimal Policy



Iteration 2:

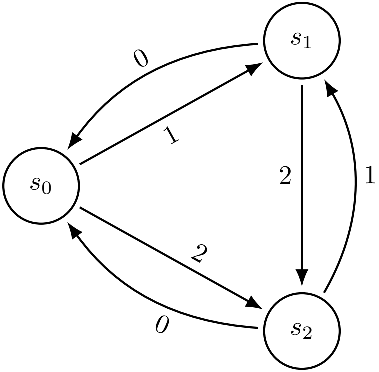
$$
\begin{aligned}
V^2(s_0) &= \max\{\,1 + \beta\,V^1(s_1),\ 2 + \beta\,V^1(s_2)\,\} \\
&= \max\{\,1 + 0.9 \cdot 2,\ 2 + 0.9 \cdot 1\,\} = 2.9 \\
V^2(s_1) &= \max\{\,0 + \beta\,V^1(s_0),\ 2 + \beta\,V^1(s_2)\,\} \\
&= \max\{\,0.9 \cdot 2,\ 2 + 0.9 \cdot 1\,\} = 2.9 \\
V^2(s_2) &= \max\{\,0 + \beta\,V^1(s_0),\ 1 + \beta\,V^1(s_1)\,\} \\
&= \max\{\,0.9 \cdot 2,\ 1 + 0.9 \cdot 2\,\} = 2.8
\end{aligned}
$$

...

Iteration 95: converge to tolerance $\|\Delta V\|_\infty < 10^{-4}$

$$V^{95} \approx \big(15.263,\ 15.263,\ 14.737\big).$$

Example: Gauss–Seidel (in-place) Value Iteration — Optimal Policy



Iteration 0: $V^0(s_0) = V^0(s_1) = V^0(s_2) = 0$.

Iteration 1:

$$V^1(s_0) = \max\{\, 1 + \beta V^0(s_1),\ 2 + \beta V^0(s_2) \,\} = 2$$
$$V^1(s_1) = \max\{\, 0 + \beta V^1(s_0),\ 2 + \beta V^0(s_2) \,\} = 2$$
$$V^1(s_2) = \max\{\, 0 + \beta V^1(s_0),\ 1 + \beta V^1(s_1) \,\} = 2.8$$

Iteration 2:

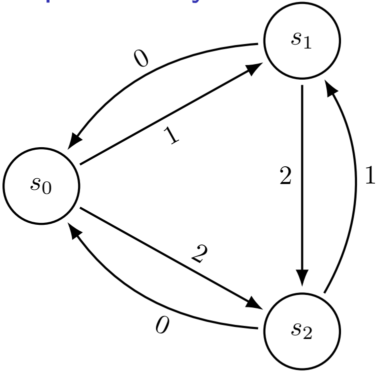
$$V^2(s_0) = \max\{\, 1 + \beta V^1(s_1),\ 2 + \beta V^1(s_2) \,\} = 4.52$$
$$V^2(s_1) = \max\{\, 0 + \beta V^2(s_0),\ 2 + \beta V^1(s_2) \,\} = 4.52$$
$$V^2(s_2) = \max\{\, 0 + \beta V^2(s_0),\ 1 + \beta V^2(s_1) \,\} = 5.068$$

...

Iteration 51: converge to tolerance $\|\Delta V\|_\infty < 10^{-4}$

$$V^{51} \approx \big(15.263,\ 15.263,\ 14.737\big).$$

Example: Policy Function Iteration (PFI)



Iteration 0: initialize policy

$$\pi^0 = \begin{array}{c|ccc} & a_0 & a_1 & a_2 \\ \hline s_0 & 0 & \frac{1}{2} & \frac{1}{2} \\ s_1 & \frac{1}{2} & 0 & \frac{1}{2} \\ s_2 & \frac{1}{2} & \frac{1}{2} & 0 \end{array}$$

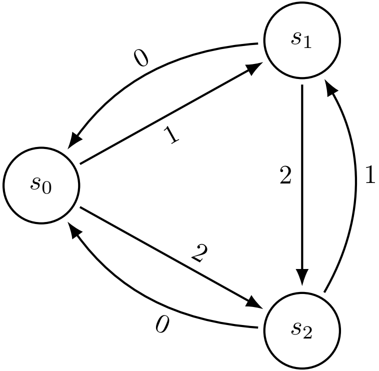
Iteration 1:
Step 1: Policy evaluation: Converged in 49 iterations to

$$V^{\pi^0} = (10.3448,\ 10.0000,\ 9.6552)$$

Step 2: Policy improvement (greedy w.r.t. V^{π^0})

$$\pi^1(s_0) = \arg\max\{\,1 + \beta V_1^{\pi^0},\ 2 + \beta V_2^{\pi^0}\,\} = a_2$$
$$\pi^1(s_1) = \arg\max\{\,0 + \beta V_0^{\pi^0},\ 2 + \beta V_2^{\pi^0}\,\} = a_2$$
$$\pi^1(s_2) = \arg\max\{\,0 + \beta V_0^{\pi^0},\ 1 + \beta V_1^{\pi^0}\,\} = a_1$$

Example: Policy Function Iteration (PFI)



Iteration 2: policy $\pi^1 = (a_2, a_2, a_1)$

Step 1: Policy evaluation: Start from V^{π^0} as initial guess. Converged in 46 iterations to

$$V^{\pi^1} = \big(15.2632,\ 15.2632,\ 14.7368\big).$$

Step 2: Policy improvement (greedy w.r.t. V^{π^1})

$$\pi^2(s_0) = \arg\max\{\, 1 + \beta V_1^{\pi^1},\ 2 + \beta V_2^{\pi^1} \,\} = a_2$$
$$\pi^2(s_1) = \arg\max\{\, 0 + \beta V_0^{\pi^1},\ 2 + \beta V_2^{\pi^1} \,\} = a_2$$
$$\pi^2(s_2) = \arg\max\{\, 0 + \beta V_0^{\pi^1},\ 1 + \beta V_1^{\pi^1} \,\} = a_1$$

$\Rightarrow$ $\boxed{\pi^2 = \pi^1}$ (policy converged) $\Rightarrow$ Stop.

Value Function Iteration (time-homogeneous, infinite horizon)

- Step 0 (inputs): state space $\mathcal{S}$; actions $\mathcal{A}(s)$; reward $r(s,a)$; transition $P(\cdot \mid s,a)$; discount $\beta \in (0,1)$; tolerance $\varepsilon > 0$.
- Step 1 (Bellman operator): for any $V : \mathcal{S} \to \mathbb{R}$,

$$(TV)(s) \;=\; \max_{a \in \mathcal{A}(s)} \Big\{ r(s,a) + \beta\, \mathbb{E}\big[V(S') \mid s,a\big] \Big\}.$$

- Step 2 (initialize value): choose $V^{(0)}$ (e.g., $V^{(0)} \equiv 0$); set $k \leftarrow 0$.
- Step 3 (value update): for each state s,

$$Q_{V^{(k)}}(s,a) \;=\; r(s,a) + \beta\, \mathbb{E}\Big[V^{(k)}(S') \mid s,a\Big] \quad \text{for all } a \in \mathcal{A}(s),$$

$$V^{(k+1)}(s) \;\leftarrow\; \max_{a \in \mathcal{A}(s)} Q_{V^{(k)}}(s,a).$$

- Step 4 (convergence check): compute $\Delta_{k+1} = \|V^{(k+1)} - V^{(k)}\|_\infty$. If $\Delta_{k+1} \leqslant \varepsilon$, go to Step 5; else set $k \leftarrow k+1$ and return to Step 3.
- Step 5 (policy extraction): greedy w.r.t. $Q_{V^{(k+1)}}$,

$$\pi^*(s) \;\in\; \arg\max_{a \in \mathcal{A}(s)} \Big\{ r(s,a) + \beta\, \mathbb{E}\Big[V^{(k+1)}(S') \mid s,a\Big] \Big\}.$$

Policy Function Iteration (time-homogeneous, infinite horizon)

- Step 0 (inputs): state space $\mathcal{S}$; actions $\mathcal{A}(s)$; reward $r(s,a)$; transition $P(\cdot \mid s,a)$; discount $\beta \in (0,1)$; tolerances $\varepsilon_{\text{eval}}, \varepsilon_{\text{imp}} > 0$.
- Step 1 (policy Bellman operator): for a stationary policy π and any $V : \mathcal{S} \to \mathbb{R}$,

$$(T^\pi V)(s) \;=\; r\big(s,\pi(s)\big) \;+\; \beta\,\mathbb{E}\big[V(S') \mid s,\pi(s)\big].$$

- Step 2 (initialize policy): choose any feasible stationary policy $\pi^{(0)}$; set $k \leftarrow 0$.
- Step 3 (policy evaluation): compute $V^{\pi^{(k)}}$ as the fixed point of $T^{\pi^{(k)}}$,

$$V^{\pi^{(k)}} \;=\; T^{\pi^{(k)}} V^{\pi^{(k)}},$$

  either <u>exactly</u> (linear solve in finite MDPs) or <u>iteratively</u> until $\|T^{\pi^{(k)}}V - V\|_\infty \leqslant \varepsilon_{\text{eval}}$.
- Step 4 (policy improvement): for each $s \in \mathcal{S}$,

$$\pi^{(k+1)}(s) \;\in\; \arg\max_{a\in\mathcal{A}(s)} \Big\{ r(s,a) + \beta\,\mathbb{E}\Big[V^{\pi^{(k)}}(S') \mid s,a\Big] \Big\}.$$

- Step 5 (convergence check): if $\pi^{(k+1)} = \pi^{(k)}$ (or the improvement gain $\leqslant \varepsilon_{\text{imp}}$), go to Step 6; else set $k \leftarrow k+1$ and return to Step 3.
- Step 6 (outputs): optimal policy $\pi^* = \pi^{(k)}$ and value $V^* = V^{\pi^{(k)}}$.

Decomposing value iteration into evaluation + improvement

Value iteration (one step):

$$V^{(k+1)} \;=\; (TV^{(k)})(s) \;=\; \max_{a\in\mathcal{A}(s)} \left\{ r(s,a) + \beta\,\mathbb{E}[\,V^{(k)}(S') \mid s,a] \right\}.$$

Equivalent two-step decomposition:

1. improvement (greedy w.r.t. $V^{(k)}$):

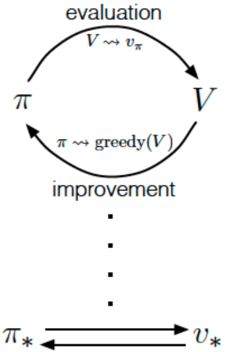
$$\pi^{(k)}(s) \in \arg\max_{a\in\mathcal{A}(s)} \left\{ r(s,a) + \beta\,\mathbb{E}[\,V^{(k)}(S') \mid s,a] \right\};$$

2. one-step policy evaluation:

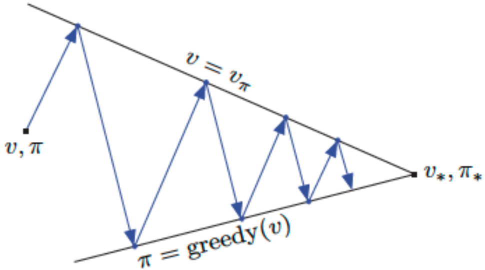
$$V^{(k+1)}(s) \;=\; (T^{\pi^{(k)}} V^{(k)})(s) \;=\; r\big(s, \pi^{(k)}(s)\big) + \beta\,\mathbb{E}\Big[V^{(k)}(S') \mid s, \pi^{(k)}(s)\Big].$$

Notes: Full evaluation to $V^{\pi^{(k)}}$ yields classical policy iteration; a single backup yields value iteration; in-between gives modified policy iteration. All are instances of GPI.

Generalized Policy Iteration: two views



(a) View 1



(b) View 2

Figure 1: Generalized Policy Iteration (two views), adapted from Sutton and Barto (2018), Chapter 4.

Howard Policy Iteration (PFI/VFI hybrid)

- Step 0 (inputs): as in PFI, plus an integer $m \geqslant 1$ (number of evaluation sweeps per improvement) and tolerance $\varepsilon > 0$.
- Step 1 (initialize): choose $\pi^{(0)}$ and $V^{(0)}$; set $k \leftarrow 0$.
- Step 2 (partial policy evaluation): set $U^{(0)} \leftarrow V^{(k)}$; for $i = 0, 1, \ldots, m-1$,

$$U^{(i+1)} \;\leftarrow\; T^{\pi^{(k)}} U^{(i)}.$$

  Set $V^{(k+1)} \leftarrow U^{(m)}$.
- Step 3 (policy improvement): for each s,

$$\pi^{(k+1)}(s) \;\in\; \arg\max_{a \in \mathcal{A}(s)} \Big\{ r(s,a) + \beta\, \mathbb{E}\Big[V^{(k+1)}(S') \mid s, a \Big] \Big\}.$$

- Step 4 (convergence check): if $\|V^{(k+1)} - V^{(k)}\|_\infty \leqslant \varepsilon$ <u>and</u> $\pi^{(k+1)} = \pi^{(k)}$, stop; else set $k \leftarrow k+1$ and return to Step 2.
- Notes: $m = 1$ recovers value iteration (VFI-style single backup before improvement); $m = \infty$ recovers policy function iteration with full evaluation (PFI/Howard).

Approximation in Dynamic Programming

approximate $\max$
Discretization / Selective Minimization / Gradient

at s_t

$$\max_{a_t \in \mathcal{A}_t(s_t)} \mathbb{E}\Big[r_t(s_t, a_t, W_t) \; + \; \beta \, \tilde{V}_{t+1}(S_{t+1}) \,\Big|\, s_t, a_t \Big]$$

approximate $\mathbb{E}$
Monte Carlo / Quasi-MC / Quadrature / Sparse sampling

approximate $\tilde{V}_{t+1}$
Parametric approximation / Neural networks / Problem approximation / Rollout

References & Further Reading

- Dimitri P. Bertsekas, **Reinforcement learning and optimal control (2025 Spring course at ASU)**, Lecture 3: https://web.mit.edu/dimitrib/www/RLbook.html
- Sutton and Barto, **Reinforcement Learning: An Introduction**, Chapter 4 and 5, https://web.stanford.edu/class/psych209/Readings/SuttonBartoIPRLBook2ndEd.pdf

Sutton, Richard S. and Andrew G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed., MIT Press, 2018.