Theory and Methods for 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-618 (Spring 2020)

lions@epfl HASLERSTIFTUNG SDSC FNSNF Google AI ZEISS Fonds National Suisse Schweizerischer Nationalfonds Fondo Nazionale Svizzero Swiss National Science Foundation erc EPFL

License Information for Reinforcement Learning Slides

- This work is released under a Creative Commons License with the following terms:
- **Attribution**
  - The licensor permits others to copy, distribute, display, and perform the work. In return, licensees must give the original authors credit.
- **Non-Commercial**
  - The licensor permits others to copy, distribute, display, and perform the work. In return, licensees may not use the work for commercial purposes – unless they get the licensor's permission.
- **Share Alike**
  - The licensor permits others to distribute derivative works only under a license identical to the one that governs the licensor's work.
- Full Text of the License

Outline

- This class:
  1. Dymanic programming
- Next class:
  1. Monte Carlo Methods

Recommended reading

- Chapter 4 in S. Sutton, and G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, 2018.

Motivation

Motivation

How to optimally act in the environment (MDP)? In this lecture, we study the iterative approaches to solve the planning problem.

Prediction and Control

Prediction

For a given policy π, estimate

- state value function $v_\pi : \mathcal{S} \to \mathbb{R}$
- state-action value function $q_\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$

Control

Estimate

- optimal state value function $v_* : \mathcal{S} \to \mathbb{R}$
- optimal state-action value function $q_* : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$

Policies and Value Functions

State-value function for policy π

- value of a state s under a policy π:

$$v_\pi(s) \ := \ \mathbb{E}_\pi[G_t \mid S_t = s] \ = \ \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \middle| S_t = s\right]$$

- how good for an agent to be in a particular state, for a given policy.
- the expected return when starting in s and following π thereafter.
- the value of the terminal state, if any, is always zero.

Policies and Value Functions

Action-value function for policy π

- value of taking action a in state s under a policy π:

$$q_\pi(s,a) \;:=\; \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a] \;=\; \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \middle| S_t = s, A_t = a\right]$$

- how good for an agent to execute a particular action in a particular state, for a given policy.
- expected return starting from s, taking the action a, and thereafter following policy π

Richard E. Bellman

Figure: https://en.wikipedia.org

- Major contributions:
  - Bellman equation
  - Hamilton-Jacobi-Bellman equation
  - Bellman-Ford algorithm

Bellman Equation

Bellman equation for v_π

- relationship between the value of a state and the values of its successor states:

$$\begin{aligned} v_\pi(s) &:= \mathbb{E}_\pi[G_t \mid S_t = s] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} \mid S_t = s] \\ &= \sum_a \pi(a \mid s) \sum_{s'} \sum_r p(s', r \mid s, a) \Big[r + \gamma \mathbb{E}_\pi[G_{t+1} \mid S_{t+1} = s']\Big] \\ &= \sum_a \pi(a \mid s) \sum_{s', r} p(s', r \mid s, a) \Big[r + \gamma v_\pi(s')\Big], \quad \forall s \in \mathcal{S} \end{aligned}$$

- this is a set of (linear) equations, one for each state
- the value function v_π is the unique solution to its Bellman equation
- state value function $=$ immediate reward $+$ discounted value of successor state

Bellman Equation

Bellman equation for q_π

The action value function can similarly be decomposed:

$$\begin{aligned}
q_\pi(s,a) &:= \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a] \\
&= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} \mid S_t = s, A_t = a] \\
&= \sum_{s',r} p(s',r \mid s,a) \sum_{a'} \pi(a' \mid s') \Big[r + \gamma \mathbb{E}_\pi[G_{t+1} \mid S_{t+1} = s', A_{t+1} = a']\Big] \\
&= \sum_{s',r} p(s',r \mid s,a) \sum_{a'} \pi(a' \mid s') \Big[r + \gamma q_\pi(s',a')\Big], \quad \forall s \in \mathcal{S}, a \in \mathcal{A}
\end{aligned}$$

Backup Diagram

- Bellman equation for v_π:

$$v_\pi(s) \;=\; \mathbb{E}_{a\sim\pi(\cdot|s)}\mathbb{E}_{s',r\sim P(\cdot|s,a)}\left[r+\gamma v_\pi(s')\right]$$

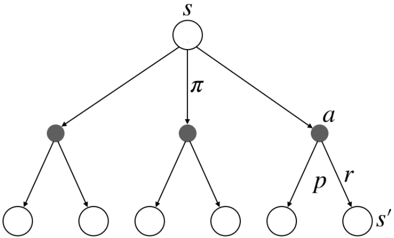


Figure: Backup diagram for v_π

Backup Diagram

- Bellman equation for q_π:

$$q_\pi(s,a) \;=\; \mathbb{E}_{s',r\sim P(\cdot|s,a)}\mathbb{E}_{a'\sim\pi(\cdot|s')}\left[r + \gamma q_\pi(s',a')\right]$$

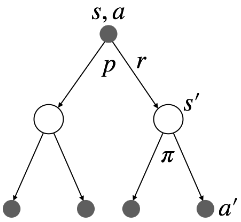


Figure: Backup diagram for q_π

Example: Gridworld

- Recall the gridworld example:
  - random policy: $\pi(a \mid s) = 0.25, \forall a \in \mathcal{A}, s \in \mathcal{S}$
  - discount rate: $\gamma = 0.9$
  - $v_\pi(s) \;=\; \mathbb{E}_{a\sim\pi(\cdot|s)}\mathbb{E}_{s',r\sim P(\cdot|s,a)}\left[r + \gamma v_\pi(s')\right]$
  - solve the linear system $V^\pi = R^\pi + \gamma P^\pi V^\pi$, where $V^\pi_s = v_\pi(s)$, $R^\pi_s = \mathbb{E}_{a\sim\pi(\cdot|s)}\mathbb{E}_{s',r\sim P(\cdot|s,a)}\left[\sum_{s'} r(s,a,s')\right]$, and $P^\pi_{s,s'} = \sum_a \pi(a \mid s)P(s' \mid s,a)$.

	1	2	3	4	5
1	3.31	8.79	4.43	5.32	1.49
2	1.52	2.99	2.25	1.91	0.55
3	0.05	0.74	0.67	0.36	-0.4
4	-0.97	-0.44	-0.35	-0.59	-1.18
5	-1.86	-1.35	-1.23	-1.42	-1.98

Optimal Value Fuctions

• Optimal value function measures the best possible goodness of state or state-action pair under all possible policies.

Definition

- *optimal state-value function*:

$$v_*(s) \;:=\; \max_\pi v_\pi(s), \quad \forall s \in \mathcal{S}.$$

- *optimal action-value function*:

$$q_*(s,a) \;:=\; \max_\pi q_\pi(s,a), \quad \forall s \in \mathcal{S}, a \in \mathcal{A}.$$

• Relationship between q_* and v_*:

$$q_*(s,a) \;=\; \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1}) \mid S_t = s, A_t = a].$$

Optimal Policy

- Partial order:

$$\pi \geq \pi' \text{ iff } v_\pi(s) \geq v_{\pi'}(s), \forall s \in \mathcal{S}$$

Theorem ([?])

For any Markov Decision Process

- *there exists an optimal policy π_* that is better than or equal to all other policies,* $\pi_* \geq \pi, \forall \pi$
- *all optimal policies achieve the same optimal value function,* $v_{\pi_*}(s) = v_*(s)$
- *all optimal policies achieve the same optimal action-value function,* $q_{\pi_*}(s, a) = q_*(s, a)$

- Optimal policy for a MDP in continuing task is:
  - deterministic
  - stationary (does not depend on time step)
  - not necessarily unique

Finding an Optimal Policy

- An optimal policy can be found by maximising over $q_*(s,a)$,

$$\pi_*(a \mid s) \;=\; \begin{cases} 1 \text{ if } a = \arg\max_{a \in \mathcal{A}} q_*(s,a) \\ 0 \text{ otherwise} \end{cases}$$

  - there is always a deterministic optimal policy for any MDP
  - if we know $q_*(s,a)$, we immediately have the optimal policy

- Assumptions:
  - we accurately know the dynamics of the environment
  - we have enough computational resources to complete the computation of the solution (polynomial in number of states)
  - the Markov property

Bellman Optimality Equation

Bellman optimality equation for v_*

- relationship between the optimal value of a state and the optimal values of its successor states:

$$
\begin{aligned}
v_*(s) &= \max_a q_{\pi_*}(s,a) \\
&= \max_a \mathbb{E}_{\pi_*}[G_t \mid S_t = s, A_t = a] \\
&= \max_a \mathbb{E}_{\pi_*}[R_{t+1} + \gamma G_{t+1} \mid S_t = s, A_t = a] \\
&= \max_a \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1}) \mid S_t = s, A_t = a] \\
&= \max_a \sum_{s',r} p(s',r \mid s,a)[r + \gamma v_*(s')]
\end{aligned}
$$

- value of a state under an optimal policy must equal the expected return for the best action from that state
- a system of (non-linear) equations, one for each state
- v_* is the unique solution

Bellman Optimality Equation

Bellman optimality equation for q_*

The optimal action value function can similarly be decomposed:

$$
\begin{aligned}
q_*(s,a) &= \mathbb{E}[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a') \mid S_t = s, A_t = a] \\
&= \sum_{s',r} p(s', r \mid s, a)[r + \gamma \max_{a'} q_*(s', a')]
\end{aligned}
$$

Backup Diagram

- Bellman optimality equation for v_*:

$$v_*(s) \;=\; \max_a \mathbb{E}_{s',r\sim P(\cdot|s,a)}\left[r + \gamma v_*(s')\right]$$

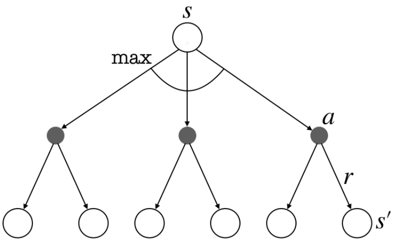


Figure: Backup diagram for v_*

Backup Diagram

- Bellman optimality equation for q_*:

$$q_*(s,a) \;=\; \mathbb{E}_{s',r\sim P(\cdot|s,a)}\left[r + \gamma \max_{a'} q_*(s',a')\right]$$

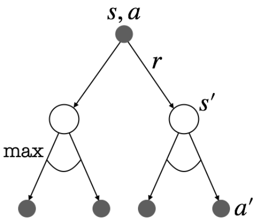


Figure: Backup diagram for q_*

Planning Problem

Assumptions

- we consider finite MDPs $(\mathcal{S}^+, \mathcal{A}, \mathcal{R}, P, P_0, \gamma)$ with finite $\mathcal{S}^+$, $\mathcal{A}$ and $\mathcal{R}$, *i.e.*,

$$\left|\mathcal{S}^+\right| < \infty, \; |\mathcal{A}| < \infty, \; \text{and } \; |\mathcal{R}| < \infty.$$

- we consider a problem with known dynamics given by a set of probabilities, *i.e.*,

$$p(s', r \mid s, a) \text{ is known } \forall s \in \mathcal{S}, s' \in \mathcal{S}^+, a \in \mathcal{A}, r \in \mathcal{R}$$

Policy Evaluation (Prediction)

Prediction

Compute the state-value function v_π for an arbitrary policy π:

$$v_\pi(s) \; = \; \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a) \left[r + \gamma v_\pi(s') \right]$$

- By Banach fixed-point theorem, existence and uniqueness of v_π are guaranteed if:
  - discounting rate $\gamma < 1$ or,
  - eventual termination is guaranteed from all states under the policy π.
- System of $|\mathcal{S}|$ simultaneous linear equations in $|\mathcal{S}|$ unknowns.

Policy Evaluation (Prediction)

Iterative solution

Construct a sequence of approximations $\{v_k\}_{k\in\mathbb{N}}$ of v_π where $v_k : \mathcal{S}^+ \to \mathbb{R}$.

- initialization: Pick v_0 arbitrarily (except that $v_0(\text{terminal state}) = 0$).
- update: $\forall s \in \mathcal{S}$

$$\begin{aligned} v_{k+1}(s) &= \mathbb{E}_\pi[R_{t+1} + \gamma v_k(S_{t+1}) \mid S_t = s] \\ &= \sum_a \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a) \left[r + \gamma v_k(s')\right] \end{aligned}$$

- $v_k = v_\pi$ is a fixed point for this update rule.
- the sequence $\{v_k\}$, in general, converges to v_π as $k \to \infty$ under the same conditions that guarantee the existence of v_π.

Iterative Policy Evaluation (IPE)

Stopping condition

$$\max_{s \in \mathcal{S}} |v_k(s) - v_{k+1}(s)| \;\leq\; \theta,$$

where θ is a small threshold determining the accuracy of the estimation.

Iterative policy evaluation

Input: the policy π to be evaluated

Initialize: an array $V(s) = 0$, for all $s \in \mathcal{S}^+$, a small number $\theta > 0$

Repeat:

$\quad \Delta \leftarrow 0$

$\quad$ For each $s \in \mathcal{S}$:

$\quad\quad v \leftarrow V(s)$

$\quad\quad V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a) \left[r + \gamma V(s')\right]$

$\quad\quad \Delta \leftarrow \max(\Delta, |v - V(s)|)$

until $\Delta < \theta$

Output: $V \approx v_\pi$

Example: 4×4 Gridworld

- Environment setup:
  - actions $\mathcal{A} = \{\texttt{up}, \texttt{down}, \texttt{right}, \texttt{left}\}$.
  - states $\mathcal{S} = \{1, 2, 3, \ldots, 14\}$.
  - rewards $r = -1$ until the terminal sate is reached
  - terminal states are denoted by T which is (0,0) and (4,4), $v_k(T) = 0$ for all k
  - actions that would take the agent off the grid leave the state unchanged
  - suppose the agent follows the equiprobable random policy (all actions equally likely)
  - $\gamma = 1$, $\pi(a|s) = 1/4$ for all state s and action a.
  - initial $v_0(s) = 0$ for all state s.

T	1	2	3
4	5	6	7
8	9	10	11
12	13	14	T

IPE: first iteration $k = 0$

- We have, for all $s = 1, \ldots, 14$:

$$
\begin{aligned}
v_1(s) &= \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)\Big(r + \gamma v_0(s')\Big) \\
&= \frac{1}{4} \sum_a \sum_{s'} p(s', -1|s, a)\Big(-1 + 1 \times v_0(s')\Big) \\
&= -\frac{1}{4} \sum_a \sum_{s'} p(s', -1|s, a) \\
&= -1
\end{aligned}
$$

- Hence

$v_1 =$

0	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	0

IPE: second iteration $k = 1$

- We have, for all $s = 1, \ldots, 14$:

$$
\begin{aligned}
v_1(s) &= \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)\Big(r + \gamma v_0(s')\Big) \\
&= \frac{1}{4} \sum_a \sum_{s'} p(s', -1|s, a)\Big(-1 + 1 \times v_0(s')\Big) \\
&= \frac{1}{4} \sum_{s'=0}^{15} p(s', -1|s, \texttt{up})\Big(-1 + v_0(s')\Big) \\
&\quad + \frac{1}{4} \sum_{s'=0}^{15} p(s', -1|s, \texttt{down})\Big(-1 + v_0(s')\Big) \\
&\quad + \frac{1}{4} \sum_{s'=0}^{15} p(s', -1|s, \texttt{left})\Big(-1 + v_0(s')\Big) \\
&\quad + \frac{1}{4} \sum_{s'=0}^{15} p(s', -1|s, \texttt{right})\Big(-1 + v_0(s')\Big)
\end{aligned}
$$

IPE: second iteration $k = 1$

- If $s = 1$ then

1. $p(s', -1|1, \texttt{up}) = 0$ for all $s' \neq 1$ and hence
$\frac{1}{4}\sum_{s'=0}^{15} p(s', -1|1, \texttt{up})\Big(-1 + v_0(s')\Big) = \frac{1}{4}p(1, -1|1, \texttt{up})(-1-1) = -1/2$

2. $p(s', -1|1, \texttt{down}) = 0$ for all $s' \neq 5$ and hence
$\frac{1}{4}\sum_{s'=0}^{15} p(s', -1|s, \texttt{down})\Bigg(-1 + v_0(s')\Bigg) = \frac{1}{4}p(5, -1|1, \texttt{down})(-1-1) = -1/2$

3. $p(s', -1|1, \texttt{left}) = 0$ for all $s' \neq 0$ and hence
$\frac{1}{4}\sum_{s'=0}^{15} p(s', -1|s, \texttt{left})\Bigg(-1 + v_0(s')\Bigg) = \frac{1}{4}p(0, -1|1, \texttt{left})(-1-0) = -1/4$

4. $p(s', -1|1, \texttt{right}) = 0$ for all $s' \neq 3$ and hence
$\frac{1}{4}\sum_{s'=0}^{15} p(s', -1|s, \texttt{right})\Bigg(-1 + v_0(s')\Bigg) = \frac{1}{4}p(3, -1|1, \texttt{right})(-1-1) = -1/2$

- Therefore, $v_2(1) = -1.75$.

IPE: second iteration $k = 1$

- Similarly,
  - $v_2(4) = v(14) = v(11) = 1.75$
  - $v_2(T) = 0$
  - $v_2(s) = -2 \quad \forall s \in \{3, 5, 6, 7, 8, 9, 10, 12, 13\}$
- Overall

$v_2 =$

0	-1.75	-2	-2
-1.75	-2	-2	-2
-2	-2	-2	-1.75
-2	-2	-1.75	0

IPE: convergence

Theorem

The IPE iterates given by

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

converges to the following fixed point equation (as $k \to \infty$):

$$v_{\pi}(s) \;=\; \sum_{a} \pi(a \mid s) \sum_{s',r} p(s', r \mid s, a) \left[r + \gamma v_{\pi}(s')\right]$$

IPE: convergence

Proof.

Let $\bar{s}$ be such that $|v_{k+1}(\bar{s}) - v_\pi(\bar{s})| = \|v_{k+1} - v_\pi\|_\infty$, then

$$
\begin{aligned}
\|v_{k+1} - v_\pi\|_\infty &= \gamma \Big| \sum_a \pi(a|\bar{s}) \sum_{s',r} p(s', r|\bar{s}, a)[v_k(s') - v_\pi(s')] \Big| \\
&\leq \gamma \sum_a \pi(a|\bar{s}) \sum_{s',r} p(s', r|\bar{s}, a) \max_{s'} |v_k(s') - v_\pi(s')| \\
&= \gamma \sum_a \pi(a|\bar{s}) \sum_{s',r} p(s', r|\bar{s}, a) \|v_k - v_\pi\|_\infty \\
&= \gamma \|v_k - v_\pi\|_\infty \\
&\leq \gamma^n \|v_0 - v_\pi\|_\infty \\
&\to 0 \text{ since } \gamma < 1.
\end{aligned}
$$

□

Policy Improvement

Example

Suppose that we know $\{v_\pi(s)\}_{s\in\mathcal{S}}$ for an arbitrary deterministic policy π.

- for some state s, should we change the policy to deterministically choose an action $a \neq \pi(s)$?
- would changing to this new policy yield an improvement?
- the answer is given by the *policy improvement theorem*.

Policy Improvement Theorem

Theorem (Policy improvement theorem)

Let π and π' be any pair of deterministic policies such that, for all $s \in \mathcal{S}$

$$q_\pi\left(s, \pi'(s)\right) \geq v_\pi(s). \tag{1}$$

Then the policy π' must be as good as, or better than, π. That is, it must obtain greater or equal expected return from all states $s \in \mathcal{S}$:

$$v_{\pi'}(s) \geq v_\pi(s). \tag{2}$$

If (1) is strict inequality at state s', then (2) is also strict inequality at state s'.

- Considering the previous example, with $\pi'(s) = a \neq \pi(s)$,
  - if $q_\pi(s, a) > v_\pi(s)$, then the changed policy π' is indeed better than π.

Policy Improvement Theorem

Proof.

From equation (1), we have:

$$
\begin{aligned}
v_\pi(s) &\leq q_\pi(s, \pi'(s)) && (3)\\
&= \mathbb{E}[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s, A_t = \pi'(s)] \\
&= \mathbb{E}_{\pi'}[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s] \\
&\leq \mathbb{E}_{\pi'}[R_{t+1} + \gamma q_\pi(S_{t+1}, \pi'(S_{t+1})) | S_t = s] && \text{(Applying (1) again)}\\
&= \mathbb{E}_{\pi'}[R_{t+1} + \gamma \mathbb{E}[R_{t+2} + \gamma v_\pi(S_{t+2}) | S_{t+1}, A_{t+1} = \pi'(S_{t+1})] | S_t = s] \\
&= \mathbb{E}_{\pi'}[R_{t+1} + \gamma \mathbb{E}_{\pi'}[R_{t+2} + \gamma v_\pi(S_{t+2}) | S_{t+1}] | S_t = s] \\
&= \mathbb{E}_{\pi'}[R_{t+1} + \gamma R_{t+2} + \gamma^2 v_\pi(S_{t+2}) | S_t = s]. && (4)
\end{aligned}
$$

Apply (4) recursively, we obtain

$$
\begin{aligned}
v_\pi(s) &\leq \mathbb{E}_{\pi'}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \ldots | S_t = s] \\
&= \mathbb{E}_{\pi'}[G_t | S_t = s] = v_{\pi'}(s). && (5)
\end{aligned}
$$

If (3) is a strict inequality, then (5) is also strict. □

Greedy Policy

Definition (Greedy Policy)

A *greedy policy* π' is defined as

$$\begin{aligned} \pi'(s) &:= \arg\max_a q_\pi(s,a) \\ &= \arg\max_a \sum_{s',r} p(s',r|s,a)[r+\gamma v_\pi(s')]. \end{aligned} \tag{6}$$

Remarks:

1. π' satisfies the condition of policy improvement theorem by construction.
2. if π' is as good as, but not better than π, then $v_{\pi'} = v_\pi$ and by (6),

   $$v_{\pi'}(s) = \max_a \sum_{s',r} p(s',r|s,a)[r+\gamma v_{\pi'}(s')],$$

   i.e., the fixed point equation is satisfied and consequently, both π and π' are optimal policies.
3. the process of making a new policy π' that improves on an original policy π, by making it greedy with respect to q_π is called *policy improvement*.
4. these concepts can easily be extended to stochastic policies.

Example: Gridworld

v_k for the random policy

Greedy policy w.r.t. v_k

$k = 0$

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

T	⌖	⌖	⌖
⌖	⌖	⌖	⌖
⌖	⌖	⌖	⌖
⌖	⌖	⌖	T

$\leftarrow$ Random policy

$k = 1$

0	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	0

T	$\leftarrow$	⌖	⌖
$\uparrow$	⌖	⌖	⌖
⌖	⌖	⌖	$\downarrow$
⌖	⌖	$\rightarrow$	T

Example: Gridworld

v_k for the random policy | Greedy policy w.r.t. v_k

$k = 2$

0	-1.7	-2	-2
-1.7	-2	-2	-2
-2	-2	-2	-1.7
-2	-2	-1.7	0

T	←	←	↑↓←→
↑	↑←	↑↓←→	↓
↑	↑↓←→	↓→	↓
↑↓←→	→	→	T

$k = 3$

0	-2.4	-2.9	-3
-2.4	-2.9	-3	-2.9
-2.9	-3	-2.9	-2.4
-3	-2.9	-2.4	0

T	←	←	↓←
↑	↑←	↓←	↓
↑	↑→	↓→	↓
↑→	→	→	T

← Optimal

⋮

$k = \infty$

0	-14	-20	-22
-14	-18	-20	-20
-20	-20	-18	-14
-22	-20	-14	0

T	←	←	↓←
↑	↑←	↓←	↓
↑	↑→	↓→	↓
↑→	→	→	T

← Optimal

Policy Iteration

Policy iteration

- start from policy π, compute v_π and use it to yield a better policy π'.
- compute $v_{\pi'}$ and improve π' to yield an even better π''.
- repeat this process until the optimal policy is reached.

$$\pi_0 \xrightarrow{E} v_{\pi_0} \xrightarrow{I} \pi_1 \xrightarrow{E} v_{\pi_1} \xrightarrow{I} \pi_2 \xrightarrow{E} \dots \xrightarrow{I} \pi_* \xrightarrow{E} v_*.$$

Remarks:

1. $\xrightarrow{E}$ denotes a step of policy evaluation.
2. $\xrightarrow{I}$ denotes a step of policy improvement.
3. Each policy is guaranteed to be a strict improvement over the previous one. If not, the optimal policy has been reached.
4. A finite MDP means a finite number of policies. Consequently, this process must converge to the optimal an optimal policy and optimal value function in a finite number of steps.

Policy Iteration Algorithm

Policy Iteration

1. **Initialization**: $\{V(s)\}_{s\in\mathcal{S}}$; a small number $\theta > 0$; $\{\pi(s)\}_{s\in\mathcal{S}}$.
2. **Policy evaluation**
   Repeat:
   - $\Delta \leftarrow 0$
   - For each $s \in \mathcal{S}$:
     - $v \leftarrow V(s)$
     - $V(s) \leftarrow \sum_{s',r} p(s', r|s, \pi(s))[r + \gamma V(s')]$
     - $\Delta \leftarrow \max(\Delta, |v - V(s)|)$

   Until $\Delta < \theta$.
3. **Policy improvement**
   $\text{policyStable} \leftarrow \text{true}$
   For each $s \in \mathcal{S}$:
   - $\text{oldAction} \leftarrow \pi(s)$
   - $\pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s', r|s, a)[r + \gamma V(s')]$
   - If $\text{oldAction} \neq \pi(s)$, then $\text{policyStable} \leftarrow \text{false}$

   If policyStable then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; Else go to 2

Value Iteration

- Drawback of policy iteration:
  - each iteration involves policy evaluation – requiring multiple sweeps through the state set.
  - convergence to v_π happens exactly only in the limit

- Can we truncate the policy evaluation step without losing the convergence guarantees of policy iteration algorithm?
  - yes, use the *value iteration* algorithm.
  - policy evaluation is stopped after only one sweep.
  - update rule, for all $s \in \mathcal{S}$:

$$
\begin{aligned}
v_{k+1}(s) &:= \max_a \mathbb{E}\left[R_{t+1} + \gamma v_k(S_{t+1}) \mid S_t = s, A_t = a\right] \\
&= \max_a \sum_{s',r} p(s', r|s, a) \left[r + \gamma v_k(s')\right]
\end{aligned}
$$

  - *interpretation:* value iteration is obtained by turning the Bellman optimality equation into an update rule.

Value Iteration Algorithm

Value Iteration

Initialize: $\{V(s)\}_{s\in\mathcal{S}}$ arbitrarily ; a small number $\theta > 0$ (determines termination).
Repeat:
$\quad \Delta \leftarrow 0$
$\quad$ For each $s \in \mathcal{S}$:
$\quad\quad v \leftarrow V(s)$
$\quad\quad V(s) \leftarrow \max_a \sum_{s',r} p(s', r|s, a) [r + \gamma V(s')]$
$\quad\quad \Delta \leftarrow \max(\Delta, |v - V(s)|)$
Until $\Delta < \theta$.
Output: a deterministic policy $\pi \approx \pi_*$ s.t.

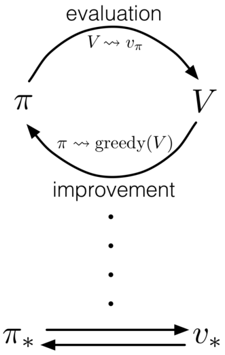
$$\pi(s) = \arg\max_a \sum_{s',r} p(s', r|s, a) \left[r + \gamma V(s')\right]$$

Asynchronous Dynamic Programming

- DP involves operations over the entire state set of the MDP.
  - if the state is very large, a single sweep of the state set can be prohibitively expensive.
  - example: Backgammon has over 10^{20} states.
- Asynchronous DP: in-place iterative DP algorithms.
  - not organized in systematic sweeps of state set.
  - uses available values of other states.
  - does not necessarily mean less computation overall.
  - the algorithm does not need to get locked in hopelessly long sweeps before making progress to improve a policy.
  - this allows to intermix computation with real-time interaction – we can run an iterative DP algorithm at the same time that the agent is experiencing the MDP.

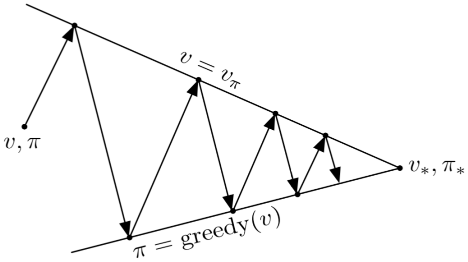
Generalized Policy Iteration

- Two processes:
  - policy evaluation – making the value function consistent with the current policy.
  - policy improvement – making the policy greedy with respect to the current value function.
- Interaction between these processes:
  - policy Iteration – these two processes alternate, each completing before the other begins.
  - value Iteration – only a single iteration of policy evaluation is performed in between each policy improvement.
  - asynchronous DP – these processes are interleaved at an even finer grain.
- Generalized Policy Iteration (GPI):
  - the general idea of letting these processes interact.
  - most RL methods are described as GPI.



Generalized Policy Iteration

- Convergence of GPI:
  - the value function stabilizes only when it is consistent with the current policy.
  - the policy stabilizes only when it is greedy with respect to the current value function.
  - thus, both processes stabilize only when a policy has been found that is greedy with respect to its own evaluation function.
  - this implies that the Bellman optimality equation holds, and thus that the policy and the value function are optimal.



Efficiency of Dynamic Programming

- Dynamic Programming
  - worst case complexity: polynomial time in $|\mathcal{S}|$ and $|\mathcal{A}|$.
  - curse of dimensionality (inherent difficulty of the problem).
  - exponentially better than direct search in policy space ($|\mathcal{A}|^{|\mathcal{S}|}$ complexity).
  - for the largest problems, DP better than linear programming methods.
  - asynchronous DP requires less memory and computation than synchronous DP methods.

Summary

- Policy iteration and value iteration algorithms:
  - built upon policy evaluation and policy improvement.
  - operate in sweeps through state space.
  - no more updating means convergence to values that satisfy the corresponding Bellman equation.
- Asynchronous DP methods:
  - in-place iterative methods.
  - update states in arbitrary order.

References

[1] Martin L Puterman.
Markov Decision Processes: Discrete Stochastic Dynamic Programming.
John Wiley & Sons, Inc., 1994.