

Lecture Schedule

Dynamical programming

- ① The finite-horizon decision problem
7 February
- ② Dynamical Programming
14 February
- ③ DP reformulations and introduction to Control
21 February

Control

- ④ Discretization and PID control
28 February
- ⑤ Direct methods and control by optimization
7 March
- ⑥ Linear-quadratic problems in control
14 March
- ⑦ Linearization and iterative LQR
21 March

Syllabus: <https://02465material.pages.compute.dtu.dk/02465public>
Help improve lecture by giving feedback on DTU learn

Reinforcement learning

- ⑧ Exploration and Bandits
28 March
- ⑨ **Bellmans equations and exact planning**
4 April
- ⑩ Monte-carlo methods and TD learning
11 April
- ⑪ Model-Free Control with tabular and linear methods
25 April
- ⑫ Eligibility traces
2 May
- ⑬ Deep-Q learning
9 May

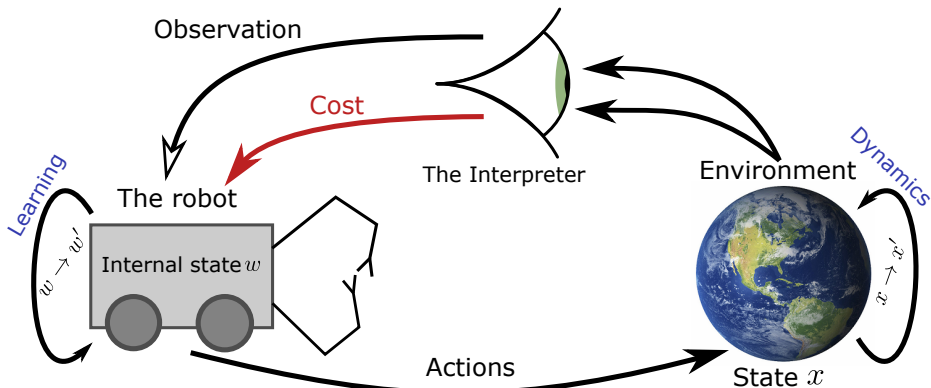
Reading material:

- [SB18, Chapter 3; 4]

Learning Objectives

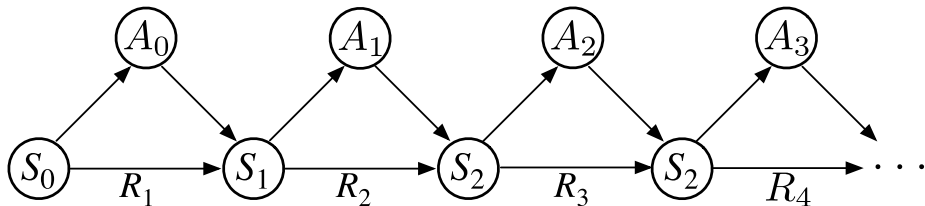
- Markov decision process
- Value/action value function and other tools
- Dynamical programming for policy evaluation and control

- Feedback on project 2 in about 2 weeks
- Project 3 is online
- Due to a combination of illness+baby I might have opened but not answered some emails. Please contact me again if I do not respond timely.



- Last time: Exploration and exploitation (+No effects)
- This time: Value functions and recursions (+Known dynamics)
- Next time: The full reinforcement-learning problem

Markov decision process



- Agent/system interacts at times $t = 0, 1, 2, \dots$
 - Agent observes state $S_t \in \mathcal{S}$
 - Agent takes action $A_t \in \mathcal{A}(S_t)$
 - Agent obtains a reward $R_{t+1} \in \mathbb{R}$; time increments to $t + 1$
- Dynamics described using conditional probabilities

$$\begin{aligned}
 p(s', r | s, a) &= \Pr \{S_{t+1} = s', R_{t+1} = r \mid S_t = s, A_t = a\} \\
 &= \Pr \{w \mid \text{s.t. } s' = f_t(s, a, w) \text{ and } r = -g_t(s, a, w)\}
 \end{aligned}$$

- If the environments stops we call it **episodic**

 `unf_gridworld.py`

Assumptions in a Markov Decision Process

- $\mathcal{S}, \mathcal{A}(s)$ are finite
- Markov property

$$\Pr \{S_{t+1}, R_{t+1} \mid S_t, A_t\} = \Pr \{S_{t+1}, R_{t+1} \mid S_0, A_0, \dots, S_t, A_t\}$$

- The **transition probabilities** are **stationary** (time-independent)

$$p(s_{t+1}, r_{t+1} \mid s_t, a_t) = p(s_{t'+1}, r_{t'+1} \mid s_{t'}, a_{t'})$$

Markov Decision Process - practically speaking

- A function that says which actions are available in a given state $\mathcal{A}(s)$
- The transition probability $p(s', r | s, a)$
- The initial state s_0
- A function which determines
 - if a state is **non-terminal**, $s_t \in \mathcal{S}$
 - or **terminal**, $s_T \notin \mathcal{S}$
- $\mathcal{S}, \mathcal{A}(s)$ are finite

An episode is $s_0, A_0, R_1, s_1, A_1, R_2, \dots, s_{T-1}, A_{T-1}, R_T, s_T$

Policy

A **policy** is a distribution over actions

$$\pi(a|s) = \Pr \{A_t = a \mid S_t = s\}$$

- Policy is time-independent
- Now a **Distribution** rather than **function** $a = \pi(s)$ because we want to **explore**

Return

For $0 \leq \gamma \leq 1$ and any t we define the accumulated γ -discounted return

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

- Equivalent to:

$$\lim_{N \rightarrow \infty} \left[\gamma^N g_N(x_N) + \sum_{k=0}^N \gamma^k g_k(s_k, a_k, w_k) \right]$$

- **Fancy rationale for $\gamma < 1$:**

- Don't worry about the far and uncertain future

- **Actual rationale for $\gamma < 1$:**

- Avoids infinities when $\gamma = 1$; simpler convergence theory

- **tl;dr:** Use $\gamma > 0.9$ unless you have good reasons not to.

Value and action-value function

The **state-value function** $v_{\pi}(s)$ is the expected return starting in s and assuming actions are selected using π :

$$v_{\pi}(s) = \mathbb{E}_{\pi} [G_t | S_t = s], \quad A_t \sim \pi(\cdot | S_t)$$

The **action-value function** $q_{\pi}(s, a)$ is the expected return starting in s , taking action a , and then follow π :

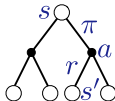
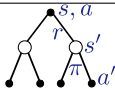
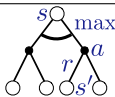
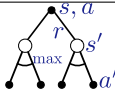
$$q_{\pi}(s, a) = \mathbb{E}_{\pi} [G_t | S_t = s, A_t = a]$$

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots$$

Note that $J_{\pi}(s) = -v_{\pi}(s)$

The reinforcement-learning problem

Where we want to end up

Bellman equation	Learning algorithm	
Bellman expectation equation for v_π $v_\pi(s) = \mathbb{E}_\pi [R + \gamma v_\pi(S') s]$	Iterative policy evaluation to learn v_π $V(s) \leftarrow \mathbb{E}_\pi [R + \gamma V(S') s]$	
Bellman expectation equation for q_π $q_\pi(s, a) = \mathbb{E}_\pi [R + \gamma q_\pi(S', A') s, a]$	Iterative policy evaluation to learn q_π $Q(s, a) \leftarrow \mathbb{E}_\pi [R + \gamma Q(S', A') s, a]$	
Policy iteration: Use policy evaluation to estimate v_π or q_π Improve by acting greedily: $\pi'(s) \leftarrow \arg \max_a q_\pi(s, a)$		
Bellman optimality equation for v_* $v_*(s) = \max_a \mathbb{E} [R + \gamma v_*(S') s, a]$	Value iteration $V(s) \leftarrow \max_a \mathbb{E} [R + \gamma V(S') s, a]$	
Bellman optimality equation for q_* $q_*(s, a) = \mathbb{E} [R + \gamma \max_{a'} q_*(S', a') s, a]$	Q-value iteration $Q(s, a) \leftarrow \mathbb{E} [R + \gamma \max_{a'} Q(S', a') s, a]$	

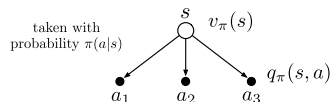
Fundamental properties of value/action-value functions

- Fundamental recursion

$$G_t = R_{t+1} + \gamma G_{t+1}$$

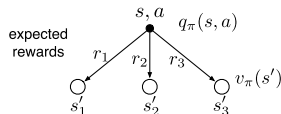
- Action-value to value function

$$v_{\pi}(s) = \mathbb{E}_{a \sim \pi(s)} [q_{\pi}(s, a)]$$



- value-function to action-value

$$q_{\pi}(s, a) = \mathbb{E} [R_{t+1} + \gamma v_{\pi}(S_{t+1}) | S_t = s, A_t = a] \quad (1)$$



The reinforcement-learning problem

Two first two Bellman equations

$$v_{\pi}(s_t) = \mathbb{E}[R_{t+1} + \gamma G_{t+1} | s] = \mathbb{E} \left[R_{t+1} + \gamma \underbrace{\mathbb{E}[G_{t+1} | s_{t+1}]}_{=v_{\pi}(s_{t+1})} | s \right]$$



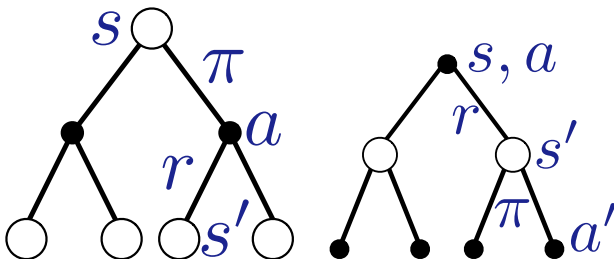
Bellman equations

- Recursive decomposition of value function. $V : \mathcal{S} \mapsto \mathbb{R}$ **initialized randomly**

$$v_{\pi}(s) \leftarrow \mathbb{E}[R_{t+1} + \gamma v_{\pi}(S_{t+1}) | S_t = s]$$

- Recursive decomposition of action-value function (**Q initialized randomly**)

$$q_{\pi}(s, a) \leftarrow \mathbb{E}[R_{t+1} + \gamma q_{\pi}(S_{t+1}, A_{t+1}) | S_t = s, A_t = a]$$



Task 1: Evaluate a policy

Iterative policy evaluation

- Given a policy π , initialize V randomly. For all s perform updates:

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

until terminal condition is met. $V(s)$ will converge to $v_\pi(s)$

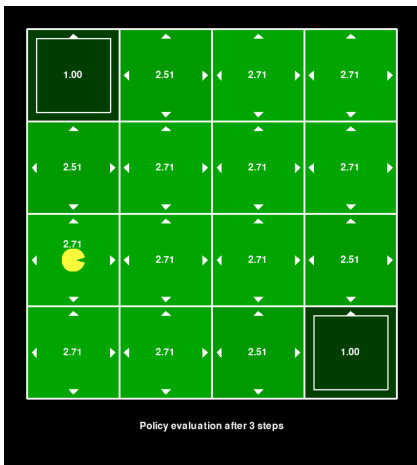
- Initialize Q randomly. For all s, a perform updates:

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

until terminal condition is met. Q will converge to q_π

 `unf_policy_improvement_gridworld.py`

Quiz: Policy evaluation



The value function v_π for the policy $\pi(a|s) = \frac{1}{4}$ is estimated using Policy Evaluation with $\gamma = 0.9$.

What is the value function in the state indicated by Pacman in the next step?

- a. 3.41
- b. 3.39
- c. 3.31
- d. 3.28
- e. Don't know.

The environment has a living reward of $R = 1$ and if it moves into the wall it stays in the current state.

Optimal value function

The optimal state-value function v_* is the maximum value function over all policies

$$v_*(s) = \max_{\pi} v_{\pi}(s)$$

The optimal action-value function q_* is the maximum action-value function over all policies

$$q_*(s, a) = \max_{\pi} q_{\pi}(s, a)$$

We define a partial ordering over policies as

$$\pi \geq \pi' \text{ if for all } s: v_{\pi}(s) \geq v_{\pi'}(s)$$

- Given any function $q : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ we can define the **greedy policy** π' wrt. q

$$\pi'(s) = \arg \max_a q(s, a)$$

- Given any function $v : \mathcal{S} \mapsto \mathbb{R}$ we can define **greedy policy** π' wrt. v

$$\pi'(s) = \arg \max_a \mathbb{E}_{s', r} [r + \gamma v(s') | s, a]$$

Policy improvement theorem

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

$$q_{\pi}(s, \pi'(s)) \geq v_{\pi}(s) \quad (2)$$

Then $\pi' \geq \pi$ meaning for all $s \in \mathcal{S}$

$$v_{\pi'}(s) \geq v_{\pi}(s)$$

Inequality is strict if any inequality in eq. (2) is strict.

Given v_π , define new policy π' to be greedy with respect to v_π . Then:

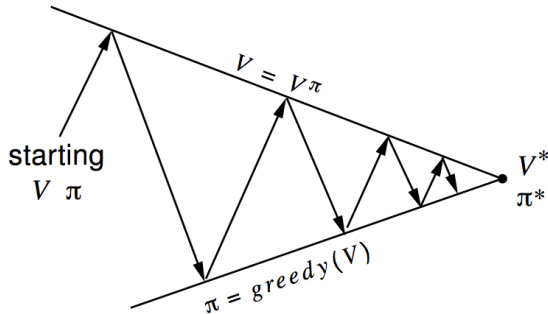
$$\begin{aligned} v_\pi(s) &= \mathbb{E}_{a \sim \pi(s)} [q_\pi(s, a)] \\ &\leq \max_a q_\pi(s, a), \quad \text{True by simple properties of expectations} \\ &= q_\pi(s, a^*), \quad a^* = \arg \max_a q_\pi(s, a) \\ &= q_\pi(s, \pi'(s)), \quad \pi' \text{ greedy policy wrt. } v_\pi \end{aligned}$$

Observations:

- Being greedy wrt. π means $\pi' \geq \pi$ by the policy-improvement theorem

Let v_* , q_* be the optimal value and action-value functions of an MDP, let π be any policy and finally let v_π and q_π be the value/action-value function associated with π . Which one of the following statements are true in general?

- a. $\max_s q_*(s, a) = v_*(a)$
- b. There is a policy π , a state s and an action a so that $q_*(s, a) < q_\pi(s, a)$
- c. For all π and a it is true that $q_*(s, a) > q_\pi(s, a)$
- d. There is a policy π and state s so that $\max_a q_*(s, a) = v_\pi(s)$
- e. Don't know.



- Given initial policy π
- Compute v_π using policy evaluation
- Let π' be greedy policy wrt. v_π
- Repeat until $v_\pi = v_{\pi'}$

 `lecture_09_policy_improvement.py`

Policy Iteration (using iterative policy evaluation) for estimating $\pi \approx \pi_*$

1. Initialization

 $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation

Loop:

 $\Delta \leftarrow 0$ Loop 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

policy-stable $\leftarrow$ trueFor each $s \in \mathcal{S}$:*old-action* $\leftarrow \pi(s)$ $\pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s',r|s,a) [r + \gamma V(s')]$ If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ falseIf *policy-stable*, then stop andreturn $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

- In each step, the PI theorem guarantees that $\pi \leq \pi'$
- There is a limited number of policies so improvement cannot continue
- If $\pi = \pi'$, then the policy is in fact optimal
 - (it satisfy the Bellman optimality equation as we will see in a moment)

Suppose π_* is the policy corresponding to the optimal value function $v_*(s)$

$$\begin{aligned}v_*(s) &= \max_a q_{\pi_*}(s, a) \\ &= \max_a \mathbb{E} [R + v_{\pi_*}(S') | s, a]\end{aligned}$$

Bellmans optimality equations

- Recursion of optimal value function v_* : **Given any V**

$$v_*(s) = V(s) \leftarrow \max_a \mathbb{E} [R_{t+1} + \gamma v_*(S_{t+1}) V(S_{t+1}) | S_t = s, A_t = a] \quad (3)$$

- Recursion of optimal action-value function q_* :

$$q_*(s, a) = \mathbb{E} \left[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a') | S_t = s, A_t = a \right] \quad (4)$$

- **Theorem:** v_* (or q_*) satisfies the above recursions if (and only if) they corresponds to the optimal value function

Bellmans optimality equations Value Iteration

- Recursion of optimal value function v_* : **Given any V**

$$v_*(s) = V(s) \leftarrow \max_a \mathbb{E} [R_{t+1} + \gamma v_*(S_{t+1}) V(S_{t+1}) | S_t = s, A_t = a] \quad (5)$$

- Recursion of optimal action-value function q_* : **Given any Q**

$$q_*(s, a) = Q(s, a) \leftarrow \mathbb{E} \left[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, A'_{t+1}) Q(S_{t+1}, A_{t+1}) | S_t = s, A_t = a \right] \quad (6)$$

- **Theorem:** VI converge to optimal v_* (or q_*)

Value Iteration, for estimating $\pi \approx \pi_*$

Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
 Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(\text{terminal}) = 0$

Loop:

```

    |  $\Delta \leftarrow 0$ 
    | Loop for each  $s \in \mathcal{S}$ :
    |    $v \leftarrow V(s)$ 
    |    $V(s) \leftarrow \max_a \sum_{s', r} p(s', r | s, a) [r + \gamma V(s')]$ 
    |    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$ 
    until  $\Delta < \theta$ 
    
```

Output a deterministic policy, $\pi \approx \pi_*$, such that
 $\pi(s) = \arg \max_a \sum_{s', r} p(s', r | s, a) [r + \gamma V(s')]$

-  Dimitri P Bertsekas and Huizhen Yu.
Distributed asynchronous policy iteration in dynamic programming.
In *2010 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1368–1375. IEEE, 2010.
-  Richard S. Sutton and Andrew G. Barto.
Reinforcement Learning: An Introduction.
The MIT Press, second edition, 2018.
(Freely available online).

$$J_k(x_k) = \min_{u_k} \mathbb{E} [J_{k+1} (f_k(x_k, u_k, w_k)) + g_k(x_k, u_k, w_k)]$$

Assume the problem is independent of k :

$$J_k(x) = \min_u \mathbb{E} [J_{k+1} (f(x, u, w)) + g(x, u, w)]$$

- It will be true that $J_0 \approx J_1 \approx J_2$ etc.
- Policies will be the same initially $\pi_0 \approx \pi_1$ etc.

In fact just iterate to convergence:

$$J(x) \leftarrow \min_u \mathbb{E} [J(f(x, u, w)) + g(x, u, w)]$$

This is in fact value iteration

$$J_k(x_k) = \min_{u_k} \mathbb{E}[J_{k+1}(f_k(x_k, u_k, w_k)) + g_k(x_k, u_k, w_k)]$$

We want to remove the green part

$$J_k(x_k) = \min_{u_k} Q(x_k, u_k)$$

$$Q(x_k, u_k) = \mathbb{E}[\underbrace{J_{k+1}(f_k(x_k, u_k, w_k))}_{=\min_{u_{k+1}} Q(x_{k+1}, u_{k+1})} + g_k(x_k, u_k, w_k)]$$

Substituting, the entire equation becomes red:

$$Q(x_k, u_k) = \mathbb{E} \left[\min_{u_{k+1}} Q(f_k(x_k, u_k, w_k), u_{k+1}) + g_k(x_k, u_k, w_k) \right]$$

- Simply VI for Q -functions!

- In **synchronous updates**, we do

- For each $s \in \mathcal{S}$ compute:

$$v'_\pi(s) \leftarrow \mathbb{E}_\pi[R + \gamma v_\pi(S')|s]$$

- When done, set $v_\pi \leftarrow v'_\pi$

- In **asynchronous updates**, we re-use the updated values within one sweep

- For each $s \in \mathcal{S}$ compute:

$$v_\pi(s) \leftarrow \mathbb{E}_\pi[R + \gamma v_\pi(S')|s]$$

Both converge: You implement the **asynchronous version**, but most analysis is done in the **synchronous version**. It is also possible to structure sweeps for efficiency (see [BY10])

Convergence results

We will focus on the value function as the action-value results are very similar. First we define the operators $\mathcal{T}$ and $\mathcal{T}_\pi$:

$$(\mathcal{T}_\pi v)(s) = \mathbb{E}_\pi [R + \gamma v(S') | s] \quad (7)$$

$$(\mathcal{T}v)(s) = \max_a \mathbb{E} [R + \gamma v(S') | s, a] \quad (8)$$

If the state space is discrete $\mathcal{S} = \{s_1, \dots, s_N\}$ we can define the vector

$$v_i = v(s_i)$$

then the operators act on these vectors $\mathcal{T} : \mathbb{R}^N \rightarrow \mathbb{R}^N$

Fixed-point theorem

Let $T : A \mapsto A$ be a function and $A \subset \mathbb{R}^n$ a compact subset of $\mathbb{R}^n$. Then if for all $\mathbf{x}, \mathbf{z} \in A$

$$\|T(\mathbf{x}) - T(\mathbf{z})\| \leq \gamma \|\mathbf{x} - \mathbf{z}\|, \quad 0 \leq \gamma < 1$$

then repeatedly applying T to any $\mathbf{x}$ will converge to a single, unique fixed point $\mathbf{x}^* = T(\mathbf{x}^*)$

- In synchronous updates, we iterate for all $s \in \mathcal{S}$:

$$v'_\pi(s) \leftarrow \mathbb{E}_\pi[R + \gamma v_\pi(S')|s]$$

then $v_\pi \leftarrow v'_\pi$

- In asynchronous updates, we re-use the updated values within one sweep

$$v_\pi(s) \leftarrow \mathbb{E}_\pi[R + \gamma v_\pi(S')|s]$$

Both converge. It is also possible to structure sweeps for efficiency (see [BY10])

- Both the operators $\mathcal{T}$ and $\mathcal{T}_\pi$ are contractions in the max-norm $\|\mathbf{x}\|_\infty = \max_i |x_i|$. Example:

$$\|\mathcal{T}_\pi \mathbf{v} - \mathcal{T}_\pi \mathbf{w}\|_\infty = \max_i |\mathbb{E}_\pi [R + \gamma v(S') | s_i] - \mathbb{E}_\pi [R + \gamma w(S') | s_i]| \quad (9)$$

$$= \max_i \left| \sum_{s'} p(s' | s_i, a) (\gamma v(s') - \gamma w(s')) \right| \quad (10)$$

$$\leq \gamma \max_i \sum_{s'} p(s' | s_i, a) |v(s') - w(s')| \quad (11)$$

$$\leq \gamma \max_i \sum_{s'} p(s' | s_i, a) \|\mathbf{v} - \mathbf{w}\|_\infty = \gamma \|\mathbf{v} - \mathbf{w}\|_\infty \quad (12)$$

- Consequence: Repeatedly applying Bellmans operators will lead to a single, fixed point policy $\mathcal{T} \mathbf{v}_* = \mathbf{v}_*$ and $\mathcal{T}_\pi \mathbf{v}_\pi = \mathbf{v}_\pi$
- Therefore, PE/PI converge to v_π . VI also converges, but does it converge to the maximum?

- We know: Value iteration converge to a unique fixed point

$$\mathbf{v}_* = (\mathcal{T}\mathcal{T} \cdots \mathcal{T})(\mathbf{v})$$

- Maximum value function is defined as

$$\tilde{v}(s) = \max_{\pi} v_{\pi}(s)$$

- It could be the case that $\tilde{v}(s) = v_{\pi}(s)$, $\tilde{v}(s') = v_{\pi'}(s')$, and neither was equal to $v_*(s), v_*(s')$

Show that $v_*(s) \leq \tilde{v}(s)$

- Value iteration gives us v_* as a fixed point
- From v_* we can construct the action-values

$$q_*(s, a) = \mathbb{E}[R + \gamma v_*(S') | s, a]$$

- From these we can define the greedy policy π_*

$$\pi_*(s) = \arg \max_a q_*(s, a)$$

- By definition now $v_*(s) = (Tv_*)(s) = (\mathcal{T}_{\pi_*}v)(s)$
- Therefore v_* is the value function of the policy π_* , and so $v_*(s) \leq \tilde{v}(s)$ for all s

Show that $v_*(s) \geq \tilde{v}(s)$

- Assume $v_*(s) < \tilde{v}_\pi(s)$ for a specific s , π
- Let π_1 be the greedy policy according to $\tilde{v}_\pi$. We know that

$$\tilde{v}_\pi \leq v_{\pi_1}$$

by the policy improvement theorem

- Therefore, $v_*(s) < \tilde{v}_\pi(s) \leq v_{\pi_1}(s)$
- Repeat again to obtain π_2 and notice we are doing policy iteration
- Since we are doing policy iteration eventually $\pi_k \rightarrow \pi_\infty$
- It must be the case v_{π_∞} is a fixed-point of $\mathcal{T}$, otherwise by the policy improvement theorem we could select a better (greedy) policy
- Since the fixed point is unique, $v_{\pi_\infty} = v_*$, which is a contradiction