Introduction to Reinforcement Learning

Bruno Scherrer

INRIA (Institut National de Recherche en Informatique et ses Applications)
IECL (Institut Elie Cartan de Lorraine)

Toulouse - September 2015

Credits for this lecture

Based on some material (slides, code, etc...) from:

- Alessandro Lazaric, “Introduction to Reinforcement learning”, Toulouse, 2015
- Dimitri Bertsekas, “A series of lectures given at Tsinghua University”, Jue 2014, `http://web.mit.edu/dimitrib/www/publ.html`

Based on the book:

- “Neuro-Dynamic Programming,” by D. P. Bertsekas and J. N. Tsitsiklis, Athena Scientific, 1996

Topic: “Reinforcement Learning”

- Research area initiated in the 1950s (Bellman), known under various names (in various communities)
  - Reinforcement learning (Artificial Intelligence, Machine Learning)
  - Stochastic optimal control (Control theory)
  - Stochastic shortest path (Operations research)
  - Sequential decision making under uncertainty (Economics)

  ⇒ Markov decision processes, dynamic programming
- Control of dynamical systems (under uncertainty)
- A rich variety of (accessible & elegant) theory/math, algorithms, and applications/illustrations

Topic: "Reinforcement Learning"

- Research area initiated in the 1950s (Bellman), known under various names (in various communities)
  - Reinforcement learning (Artificial Intelligence, Machine Learning)
  - Stochastic optimal control (Control theory)
  - Stochastic shortest path (Operations research)
  - Sequential decision making under uncertainty (Economics)

  $\Rightarrow$ Markov decision processes, dynamic programming
- Control of dynamical systems (under uncertainty)
- A rich variety of (accessible & elegant) theory/math, algorithms, and applications/illustrations

Brief Outline

- Part 1: "Small" problems
  - Optimal control problem definitions
  - Dynamic Programming (DP) principles, standard algorithms
  - Learning (solving from samples)
- Part 2: "Large" problems
  - Approximate DP Algorithms
  - Theoretical guarantees

Outline for Part 1

- Finite-Horizon Optimal Control
  - Problem definition
  - Policy evaluation: Value Iteration[1]
  - Policy optimization: Value Iteration[2]

- Stationary Infinite-Horizon Optimal Control
  - Bellman operators
  - Contraction Mappings
  - Stationary policies
  - Policy evaluation
  - Policy optimization: Value Iteration[3], Policy Iteration, Modified/Optimistic Policy Iteration
  - Asynchronous Algorithms
  - Learning from samples: Real-Time Dynamic Programming, Q-Learning, TD-Learning, SARSA

The Finite-Horizon Optimal Control Problem

- Discrete-time dynamical system

$$x_{t+1} = f_t(x_t, a_t, w_t), \qquad t = 0, 1, \ldots, H-1$$

  - t: Discrete time
  - x_t: State: summarizes past information for predicting future optimization
  - a_t: Control/Action: decision to be selected at time t from a given set $A(x_t)$
  - w_t: Random parameter: disturbance/noise
  - H: Horizon: number of times control is applied
- Reward (or Cost) function that is additive over time

$$\mathbb{E}\left\{\sum_{t=0}^{H-1} r_t(x_t, a_t, w_t) + R(x_H)\right\}$$

- Goal: optimize over policies (feedback control law):

$$a_t = \pi_t(x_t), \qquad t = 0, 1, \ldots, H-1$$

Important assumptions

- The distribution of the noise w_t does not depend on past values $w_{t-1}, \ldots, w_0$ but may depend on x_t and a_t. Equivalently:

$$\mathbb{P}(x_{t+1} = x' | x_t = x, a_t = a) = \mathbb{P}(x_t = x' | \mathcal{F}_t) \quad \text{(Markov)}$$

- Optimization over policies $\pi_0, \ldots, \pi_{H-1}$, i.e. functions/rules

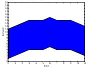
$$a_t = \pi_t(x_t)$$

  that map states to controls. This (closed-loop control) is DIFFERENT FROM optimizing over sequences of actions $a_0, \ldots, a_{H-1}$ (open-loop)!
- Optimization is in expectation (no risk measure)

The model is called: Markov Decision Process (MDP)

Example: the Retail Store Management Problem

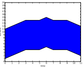
Each month t, a store contains x_t items (maximum capacity M) of a specific goods and the demand for that goods is w_t. At the end of each month the manager of the store can order a_t more items from his supplier. The cost of maintaining an inventory of x is $h(x)$. The cost to order a items is $C(a)$. The income for selling q items is $f(q)$. If the demand w is bigger than the available inventory x, customers that cannot be served leave. The value of the remaining inventory at the end of the year is $g(x)$.

$M = 20$, $f(x) = x$, $g(x) = 0.25x$, $h(x) = 0.25x$, $C(a) = (1 + 0.5a)\mathbb{1}_{a>0}$, $w_t \sim$ 

- $t = 0, 1, \dots, 11$, $H = 12$
- State space: $x \in X = \{0, 1, \dots, M\}$
- Action space: At state x, $a \in A(x) = \{0, 1, \dots, M - x\}$
- Dynamics: $x_{t+1} = [x_t + a_t - w_t]^+$
- Reward: $r(x_t, a_t, w_t) = -C(a_t) - h(x_t + a_t) + f([x_t + a_t - x_{t+1}]^+)$ and $R(x) = g(x)$.

Example: the Retail Store Management Problem

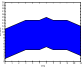
Each month t, a store contains x_t items (maximum capacity M) of a specific goods and the demand for that goods is w_t. At the end of each month the manager of the store can order a_t more items from his supplier. The cost of maintaining an inventory of x is $h(x)$. The cost to order a items is $C(a)$. The income for selling q items is $f(q)$. If the demand w is bigger than the available inventory x, customers that cannot be served leave. The value of the remaining inventory at the end of the year is $g(x)$.

$M = 20$, $f(x) = x$, $g(x) = 0.25x$, $h(x) = 0.25x$, $C(a) = (1 + 0.5a)\mathbb{1}_{a>0}$, $w_t \sim$ 

- $t = 0, 1, \ldots, 11$, $H = 12$
- State space: $x \in X = \{0, 1, \ldots, M\}$
- Action space: At state x, $a \in A(x) = \{0, 1, \ldots, M - x\}$
- Dynamics: $x_{t+1} = [x_t + a_t - w_t]^+$
- Reward: $r(x_t, a_t, w_t) = -C(a_t) - h(x_t + a_t) + f([x_t + a_t - x_{t+1}]^+)$ and $R(x) = g(x)$.

Example: the Retail Store Management Problem

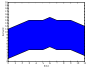
Each month t, a store contains x_t items (maximum capacity M) of a specific goods and the demand for that goods is w_t. At the end of each month the manager of the store can order a_t more items from his supplier. The cost of maintaining an inventory of x is $h(x)$. The cost to order a items is $C(a)$. The income for selling q items is $f(q)$. If the demand w is bigger than the available inventory x, customers that cannot be served leave. The value of the remaining inventory at the end of the year is $g(x)$.

$M = 20$, $f(x) = x$, $g(x) = 0.25x$, $h(x) = 0.25x$, $C(a) = (1 + 0.5a)\mathbb{1}_{a>0}$, $w_t \sim$ 

- $t = 0, 1, \ldots, 11$, $H = 12$
- State space: $x \in X = \{0, 1, \ldots, M\}$
- Action space: At state x, $a \in A(x) = \{0, 1, \ldots, M - x\}$
- Dynamics: $x_{t+1} = [x_t + a_t - w_t]^+$
- Reward: $r(x_t, a_t, w_t) = -C(a_t) - h(x_t + a_t) + f([x_t + a_t - x_{t+1}]^+)$ and $R(x) = g(x)$.

Example: the Retail Store Management Problem

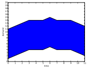
Each month t, a store contains x_t items (maximum capacity M) of a specific goods and the demand for that goods is w_t. At the end of each month the manager of the store can order a_t more items from his supplier. The cost of maintaining an inventory of x is $h(x)$. The cost to order a items is $C(a)$. The income for selling q items is $f(q)$. If the demand w is bigger than the available inventory x, customers that cannot be served leave. The value of the remaining inventory at the end of the year is $g(x)$.

$M = 20$, $f(x) = x$, $g(x) = 0.25x$, $h(x) = 0.25x$, $C(a) = (1 + 0.5a)\mathbb{1}_{a>0}$, $w_t \sim$ 

- $t = 0, 1, \ldots, 11$, $H = 12$
- State space: $x \in X = \{0, 1, \ldots, M\}$
- Action space: At state x, $a \in A(x) = \{0, 1, \ldots, M - x\}$
- Dynamics: $x_{t+1} = [x_t + a_t - w_t]^+$
- Reward: $r(x_t, a_t, w_t) = -C(a_t) - h(x_t + a_t) + f([x_t + a_t - x_{t+1}]^+)$ and $R(x) = g(x)$.

Example: the Retail Store Management Problem

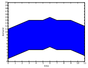
Each month t, a store contains x_t items (maximum capacity M) of a specific goods and the demand for that goods is w_t. At the end of each month the manager of the store can order a_t more items from his supplier. The cost of maintaining an inventory of x is $h(x)$. The cost to order a items is $C(a)$. The income for selling q items is $f(q)$. If the demand w is bigger than the available inventory x, customers that cannot be served leave. The value of the remaining inventory at the end of the year is $g(x)$.

$M = 20$, $f(x) = x$, $g(x) = 0.25x$, $h(x) = 0.25x$, $C(a) = (1 + 0.5a)\mathbb{1}_{a>0}$, $w_t \sim$ 

- $t = 0, 1, \ldots, 11$, $H = 12$
- State space: $x \in X = \{0, 1, \ldots, M\}$
- Action space: At state x, $a \in A(x) = \{0, 1, \ldots, M - x\}$
- Dynamics: $x_{t+1} = [x_t + a_t - w_t]^+$
- Reward: $r(x_t, a_t, w_t) = -C(a_t) - h(x_t + a_t) + f([x_t + a_t - x_{t+1}]^+)$ and $R(x) = g(x)$.

Example: the Retail Store Management Problem

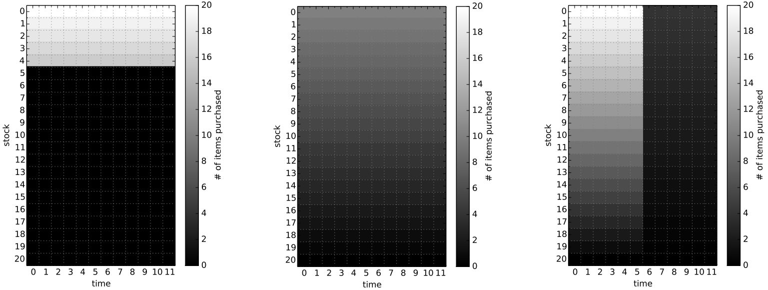
Each month t, a store contains x_t items (maximum capacity M) of a specific goods and the demand for that goods is w_t. At the end of each month the manager of the store can order a_t more items from his supplier. The cost of maintaining an inventory of x is $h(x)$. The cost to order a items is $C(a)$. The income for selling q items is $f(q)$. If the demand w is bigger than the available inventory x, customers that cannot be served leave. The value of the remaining inventory at the end of the year is $g(x)$.

$M = 20$, $f(x) = x$, $g(x) = 0.25x$, $h(x) = 0.25x$, $C(a) = (1 + 0.5a)\mathbb{1}_{a>0}$, $w_t \sim$ 

- $t = 0, 1, \ldots, 11$, $H = 12$
- State space: $x \in X = \{0, 1, \ldots, M\}$
- Action space: At state x, $a \in A(x) = \{0, 1, \ldots, M - x\}$
- Dynamics: $x_{t+1} = [x_t + a_t - w_t]^+$
- Reward: $r(x_t, a_t, w_t) = -C(a_t) - h(x_t + a_t) + f([x_t + a_t - x_{t+1}]^+)$ and $R(x) = g(x)$.

Example: the Retail Store Management Problem

2 stationary policies and 1 non-stationary policy:



$$\pi^{(2)}(x) = \max\{(M - x)/2 - x; 0\}$$

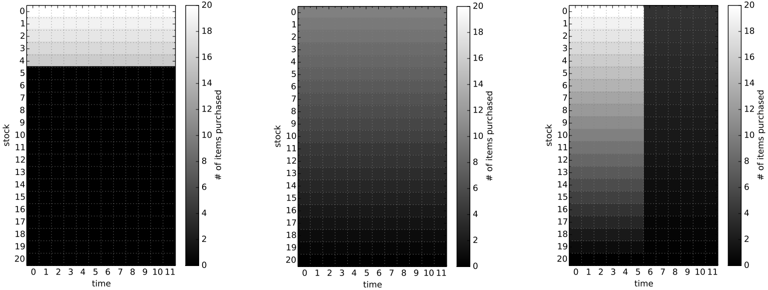
$$\pi^{(1)}(x) = \begin{cases} M - x & \text{if } x < M/4 \\ 0 & \text{otherwise} \end{cases}$$

$$\pi_t^{(3)}(x) = \begin{cases} M - x & \text{if } t < 6 \\ \lfloor (M - x)/5 \rfloor & \text{otherwise} \end{cases}$$

Remark. MDP + policy $\Rightarrow$ Markov chain on X.

Example: the Retail Store Management Problem

2 stationary policies and 1 non-stationary policy:



$$\pi^{(2)}(x) = \max\{(M - x)/2 - x; 0\}$$

$$\pi^{(1)}(x) = \begin{cases} M - x & \text{if } x < M/4 \\ 0 & \text{otherwise} \end{cases} \qquad \pi_t^{(3)}(x) = \begin{cases} M - x & \text{if } t < 6 \\ \lfloor (M - x)/5 \rfloor & \text{otherwise} \end{cases}$$

Remark. MDP + policy $\Rightarrow$ Markov chain on X.

The Finite-Horizon Optimal Control Problem

- System: $x_{t+1} = f_t(x_t, a_t, w_t), \qquad t = 0, 1, \ldots, H-1$
- Controls: $a_t \in A_t(x_t)$
- Noise: $w_t \sim \mathbb{P}(\cdot | x_t, a_t)$
- Policy $\pi = (\pi_0, \ldots, \pi_{H-1})$, such that $a_t = \pi_t(x_t) \in A_t(x_t)$.

The expected return of π starting at x at time s (the value of π in x at time s) is:

$$v_{\pi,s}(x) = \mathbb{E}_\pi \left\{ \sum_{t=s}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x \right\}$$

How can we evaluate $v_{\pi,0}(x)$ for some x ?

- Estimate by simulation and Monte-Carlo
- Develop the tree of all possible realizations ☹: time$=O(e^H)$

The Finite-Horizon Optimal Control Problem

- System: $x_{t+1} = f_t(x_t, a_t, w_t), \qquad t = 0, 1, \ldots, H-1$
- Controls: $a_t \in A_t(x_t)$
- Noise: $w_t \sim \mathbb{P}(\cdot | x_t, a_t)$
- Policy $\pi = (\pi_0, \ldots, \pi_{H-1})$, such that $a_t = \pi_t(x_t) \in A_t(x_t)$.

The expected return of π starting at x at time s (the value of π in x at time s) is:

$$v_{\pi,s}(x) = \mathbb{E}_\pi \left\{ \sum_{t=s}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x \right\}$$

How can we evaluate $v_{\pi,0}(x)$ for some x ?

- Estimate by simulation and Monte-Carlo [code]
- Develop the tree of all possible realizations ☹: time$=O(e^H)$

The Finite-Horizon Optimal Control Problem

- System: $x_{t+1} = f_t(x_t, a_t, w_t), \qquad t = 0, 1, \ldots, H-1$
- Controls: $a_t \in A_t(x_t)$
- Noise: $w_t \sim \mathbb{P}(\cdot | x_t, a_t)$
- Policy $\pi = (\pi_0, \ldots, \pi_{H-1})$, such that $a_t = \pi_t(x_t) \in A_t(x_t)$.

The expected return of π starting at x at time s (the value of π in x at time s) is:

$$v_{\pi,s}(x) = \mathbb{E}_\pi \left\{ \sum_{t=s}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x \right\}$$

How can we evaluate $v_{\pi,0}(x)$ for some x ?

- Estimate by simulation and Monte-Carlo ☹: approximate
- Develop the tree of all possible realizations ☹: time$=O(e^H)$

Policy evaluation by Value Iteration

$$
\begin{aligned}
v_{\pi,s}(x) &= \mathbb{E}_\pi\left[\sum_{t=s}^{H-1} r_t(x_t,a_t,w_t) + R(x_H) \mid x_s = x\right] \\
&= \mathbb{E}_\pi[r_s(x_s,a_s,w_s)] + \mathbb{E}_\pi\left[\sum_{t=s+1}^{H-1} r_t(x_t,a_t,w_t) + R(x_H) \mid x_s = x\right] \\
&= \mathbb{E}_\pi[r_s(x_s,a_s,w_s)] \\
&+ \sum_y \mathbb{P}(x_{s+1}=y|x_s=x, a_s=\pi(x_s))\ \mathbb{E}_\pi\left[\sum_{t=s+1}^{H-1} r_t(x_t,a_t,w_t) + R(x_H) \mid x_s=x,\ x_{s+1}=y\right] \\
&= \mathbb{E}_\pi[r_s(x_s,a_s,w_s)] + \sum_y \mathbb{P}(x_{s+1}=y|x_s=x, a_s=\pi(x_s))\ v_{\pi,s+1}(y).
\end{aligned}
$$

The computation of $v_{\pi,s}(\cdot)$ can be done from $v_{\pi,s+1}(\cdot)$, and recursively until $v_{\pi,H}(\cdot) = R(\cdot)$. ☺: time$=O(|X|^2 H)$, for all x_0!

"DP is a method for solving a complex problem by breaking it down into a collection of simpler subproblems."

[code]

Policy evaluation by Value Iteration

$$v_{\pi,s}(x) = \mathbb{E}_\pi \left[\sum_{t=s}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x \right]$$

$$= \mathbb{E}_\pi[r_s(x_s, a_s, w_s)] + \mathbb{E}_\pi \left[\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x \right]$$

$$= \mathbb{E}_\pi[r_s(x_s, a_s, w_s)]$$

$$+ \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = \pi(x_s)) \, \mathbb{E}_\pi \left[\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x, \, x_{s+1} = y \right]$$

$$= \mathbb{E}_\pi[r_s(x_s, a_s, w_s)] + \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = \pi(x_s)) \, v_{\pi,s+1}(y).$$

The computation of $v_{\pi,s}(\cdot)$ can be done from $v_{\pi,s+1}(\cdot)$, and recursively until $v_{\pi,H}(\cdot) = R(\cdot)$. ☺: time$=O(|X|^2 H)$, for all x_0!

"DP is a method for solving a complex problem by breaking it down into a collection of simpler subproblems."

[code]

Policy evaluation by Value Iteration

$$v_{\pi,s}(x) = \mathbb{E}_\pi \left[\sum_{t=s}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x \right]$$

$$= \mathbb{E}_\pi [r_s(x_s, a_s, w_s)] + \mathbb{E}_\pi \left[\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x \right]$$

$$= \mathbb{E}_\pi [r_s(x_s, a_s, w_s)]$$

$$+ \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = \pi(x_s)) \; \mathbb{E}_\pi \left[\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x, \; x_{s+1} = y \right]$$

$$= \mathbb{E}_\pi [r_s(x_s, a_s, w_s)] + \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = \pi(x_s)) \; v_{\pi,s+1}(y).$$

The computation of $v_{\pi,s}(\cdot)$ can be done from $v_{\pi,s+1}(\cdot)$, and recursively until $v_{\pi,H}(\cdot) = R(\cdot)$. ☺: time$=O(|X|^2 H)$, for all x_0!

"DP is a method for solving a complex problem by breaking it down into a collection of simpler subproblems."

[code]

Policy evaluation by Value Iteration

$$v_{\pi,s}(x) = \mathbb{E}_\pi\left[\sum_{t=s}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x\right]$$

$$= \mathbb{E}_\pi[r_s(x_s, a_s, w_s)] + \mathbb{E}_\pi\left[\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x\right]$$

$$= \mathbb{E}_\pi[r_s(x_s, a_s, w_s)]$$

$$+ \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = \pi(x_s))\ \mathbb{E}_\pi\left[\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x,\ x_{s+1} = y\right]$$

$$= \mathbb{E}_\pi[r_s(x_s, a_s, w_s)] + \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = \pi(x_s))\ v_{\pi,s+1}(y).$$

The computation of $v_{\pi,s}(\cdot)$ can be done from $v_{\pi,s+1}(\cdot)$, and recursively until $v_{\pi,H}(\cdot) = R(\cdot)$. ☺: time$=O(|X|^2 H)$, for all x_0!

"DP is a method for solving a complex problem by breaking it down into a collection of simpler subproblems."

[code]

Policy evaluation by Value Iteration

$$v_{\pi,s}(x) = \mathbb{E}_\pi\left[\sum_{t=s}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x\right]$$

$$= \mathbb{E}_\pi[r_s(x_s, a_s, w_s)] + \mathbb{E}_\pi\left[\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x\right]$$

$$= \mathbb{E}_\pi[r_s(x_s, a_s, w_s)]$$

$$+ \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = \pi(x_s))\ \mathbb{E}_\pi\left[\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x,\ x_{s+1} = y\right]$$

$$= \mathbb{E}_\pi[r_s(x_s, a_s, w_s)] + \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = \pi(x_s))\ v_{\pi,s+1}(y).$$

The computation of $v_{\pi,s}(\cdot)$ can be done from $v_{\pi,s+1}(\cdot)$, and recursively until $v_{\pi,H}(\cdot) = R(\cdot)$. ☺: time$=O(|X|^2 H)$, for all x_0!

"DP is a method for solving a complex problem by breaking it down into a collection of simpler subproblems."

[code]

Policy evaluation by Value Iteration

$$
\begin{aligned}
v_{\pi,s}(x) &= \mathbb{E}_\pi\left[\sum_{t=s}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x\right] \\
&= \mathbb{E}_\pi[r_s(x_s, a_s, w_s)] + \mathbb{E}_\pi\left[\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x\right] \\
&= \mathbb{E}_\pi[r_s(x_s, a_s, w_s)] \\
&+ \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = \pi(x_s))\ \mathbb{E}_\pi\left[\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid \cancel{x_s = x,}\ x_{s+1} = y\right] \\
&= \mathbb{E}_\pi[r_s(x_s, a_s, w_s)] + \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = \pi(x_s))\ v_{\pi,s+1}(y).
\end{aligned}
$$

The computation of $v_{\pi,s}(\cdot)$ can be done from $v_{\pi,s+1}(\cdot)$, and recursively until $v_{\pi,H}(\cdot) = R(\cdot)$. ☺: time$=O(|X|^2 H)$, for all x_0!

"DP is a method for solving a complex problem by breaking it down into a collection of simpler subproblems."

[code]

Policy evaluation by Value Iteration

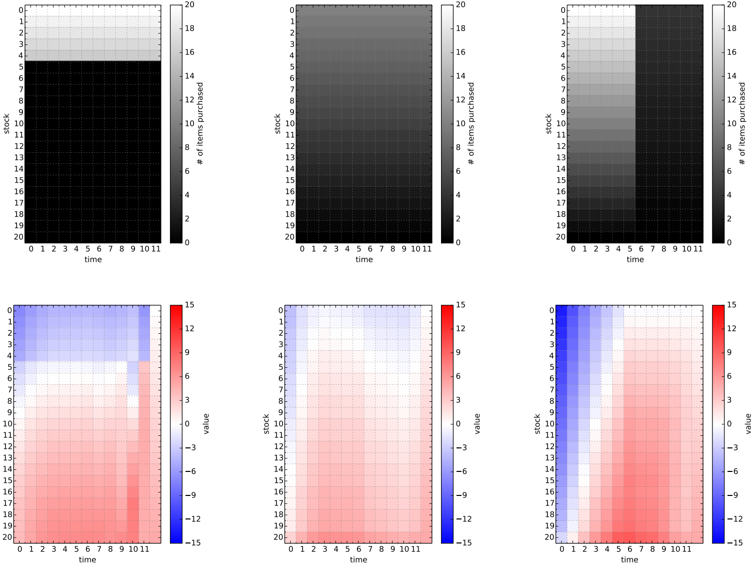
$$
\begin{aligned}
v_{\pi,s}(x) &= \mathbb{E}_\pi\left[\sum_{t=s}^{H-1} r_t(x_t,a_t,w_t) + R(x_H) \mid x_s = x\right] \\
&= \mathbb{E}_\pi[r_s(x_s,a_s,w_s)] + \mathbb{E}_\pi\left[\sum_{t=s+1}^{H-1} r_t(x_t,a_t,w_t) + R(x_H) \mid x_s = x\right] \\
&= \mathbb{E}_\pi[r_s(x_s,a_s,w_s)] \\
+ \sum_y \mathbb{P}(x_{s+1} = y \mid x_s = x, a_s = \pi(x_s))\ & \mathbb{E}_\pi\left[\sum_{t=s+1}^{H-1} r_t(x_t,a_t,w_t) + R(x_H) \mid x_s = x,\ x_{s+1} = y\right] \\
&= \mathbb{E}_\pi[r_s(x_s,a_s,w_s)] + \sum_y \mathbb{P}(x_{s+1} = y \mid x_s = x, a_s = \pi(x_s))\ v_{\pi,s+1}(y).
\end{aligned}
$$

The computation of $v_{\pi,s}(\cdot)$ can be done from $v_{\pi,s+1}(\cdot)$, and recursively until $v_{\pi,H}(\cdot) = R(\cdot)$. ☺: time$=O(|X|^2 H)$, for all x_0!

"DP is a method for solving a complex problem by breaking it down into a collection of simpler subproblems."

[code]

Example: the Retail Store Management Problem



Optimal value and policy

- System: $x_{t+1} = f_t(x_t, a_t, w_t), \qquad t = 0, 1, \ldots, H-1$
- Controls: $a_t \in A_t(x_t)$
- Noise: $w_t \sim \mathbb{P}(\cdot | x_t, a_t)$
- Policy $\pi = (\pi_0, \ldots, \pi_{H-1})$, such that $a_t = \pi_t(x_t) \in A_t(x_t)$.
- Value (expected return) of π if we start from x:

$$v_{\pi,0}(x) = \mathbb{E}_\pi \left\{ \sum_{t=0}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_0 = x \right\}$$

- Optimal value function $v_{*,0}$ and optimal policy π_*:

$$v_{*,0}(x_0) = \max_{\pi=(\pi_0,\ldots,\pi_{H-1})} v_{\pi,0}(x_0) \quad \text{and} \quad v_{\pi_*,0}(x_0) = v_{*,0}(x_0)$$

Naive optimization: time: $O(n^{mH})$ ☹

When produced by DP, $v_{*,0}$ is independent of x_0 ☺

Optimal value and policy

- System: $x_{t+1} = f_t(x_t, a_t, w_t), \qquad t = 0, 1, \ldots, H-1$
- Controls: $a_t \in A_t(x_t)$
- Noise: $w_t \sim \mathbb{P}(\cdot | x_t, a_t)$
- Policy $\pi = (\pi_0, \ldots, \pi_{H-1})$, such that $a_t = \pi_t(x_t) \in A_t(x_t)$.
- Value (expected return) of π if we start from x:

$$v_{\pi,0}(x) = \mathbb{E}_\pi \left\{ \sum_{t=0}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_0 = x \right\}$$

- Optimal value function $v_{*,0}$ and optimal policy π_*:

$$v_{*,0}(x_0) = \max_{\pi=(\pi_0,\ldots,\pi_{H-1})} v_{\pi,0}(x_0) \quad \text{and} \quad v_{\pi_*,0}(x_0) = v_{*,0}(x_0)$$

Naive optimization: time: $O(n^{mH})$ ☹

When produced by DP, $v_{*,0}$ is independent of x_0 ☺

Optimal value and policy

- System: $x_{t+1} = f_t(x_t, a_t, w_t), \qquad t = 0, 1, \ldots, H-1$
- Controls: $a_t \in A_t(x_t)$
- Noise: $w_t \sim \mathbb{P}(\cdot|x_t, a_t)$
- Policy $\pi = (\pi_0, \ldots, \pi_{H-1})$, such that $a_t = \pi_t(x_t) \in A_t(x_t)$.
- Value (expected return) of π if we start from x:

$$v_{\pi,0}(x) = \mathbb{E}_\pi \left\{ \sum_{t=0}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_0 = x \right\}$$

- Optimal value function $v_{*,0}$ and optimal policy π_*:

$$v_{*,0}(x_0) = \max_{\pi=(\pi_0,\ldots,\pi_{H-1})} v_{\pi,0}(x_0) \quad \text{and} \quad v_{\pi_*,0}(x_0) = v_{*,0}(x_0)$$

Naive optimization: time: $O(n^{mH})$ ☹
When produced by DP, $v_{*,0}$ is independent of x_0 ☺

Policy optimization by Value Iteration

$$v_{*,s}(x) = \max_{\pi_s,\dots} \mathbb{E}_{\pi_s,\dots} \left\{ \sum_{t=s}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x \right\}$$

$$= \max_{\pi_s,\pi_{s+1},\dots} \mathbb{E}_{\pi_s,\pi_{s+1},\dots} \Bigg\{ r_s(x_s, a_s, w_s)$$

$$+ \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = \pi_s(x_s)) \left(\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_{s+1} = y \right) \Bigg\}$$

$$= \max_a \Bigg\{ \mathbb{E}\left[r_s(x_s, a_s, w_s) \mid a_s = a \right]$$

$$+ \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = a) \max_{\pi_{s+1},\dots} \mathbb{E}_{\pi_{s+1},\dots} \left[\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_{s+1} = y \right] \Bigg\}$$

$$= \max_a \left\{ \mathbb{E}\left[r_s(x_s, a_s, w_s) \mid a_s = a \right] + \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = a)\, v_{*,s+1}(y) \right\}.$$

DP: The computation of $v_{*,s}(\cdot)$ can be done from $v_{*,s+1}(\cdot)$, and recursively until $v_{*,H}(\cdot) = R(\cdot)$. ☺: time$=O(|X|^2|A|H)$, for all x_0. Then, $pi_{*,s}(x)$ is the action a that minimizes the r.h.s.

[code]

Policy optimization by Value Iteration

$$v_{*,s}(x) = \max_{\pi_s,\ldots} \mathbb{E}_{\pi_s,\ldots} \left\{ \sum_{t=s}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x \right\}$$

$$= \max_{\pi_s,\pi_{s+1},\ldots} \mathbb{E}_{\pi_s,\pi_{s+1},\ldots} \left\{ r_s(x_s, a_s, w_s) \right.$$

$$\left. + \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = \pi_s(x_s)) \left(\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_{s+1} = y \right) \right\}$$

$$= \max_a \left\{ \mathbb{E}\left[r_s(x_s, a_s, w_s) \mid a_s = a \right] \right.$$

$$\left. + \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = a) \max_{\pi_{s+1},\ldots} \mathbb{E}_{\pi_{s+1},\ldots} \left[\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_{s+1} = y \right] \right\}$$

$$= \max_a \left\{ \mathbb{E}\left[r_s(x_s, a_s, w_s) \mid a_s = a \right] + \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = a)\ v_{*,s+1}(y) \right\}.$$

DP: The computation of $v_{*,s}(\cdot)$ can be done from $v_{*,s+1}(\cdot)$, and recursively until $v_{*,H}(\cdot) = R(\cdot)$. ☺: time$=O(|X|^2|A|H)$, for all x_0. Then, $pi_{*,s}(x)$ is the action a that minimizes the r.h.s.

[code]

Policy optimization by Value Iteration

$$v_{*,s}(x) = \max_{\pi_s,\dots} \mathbb{E}_{\pi_s,\dots} \left\{ \sum_{t=s}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x \right\}$$

$$= \max_{\pi_s,\pi_{s+1},\dots} \mathbb{E}_{\pi_s,\pi_{s+1},\dots} \left\{ r_s(x_s, a_s, w_s) \right.$$

$$\left. + \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = \pi_s(x_s)) \left(\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_{s+1} = y \right) \right\}$$

$$= \max_a \left\{ \mathbb{E}\left[r_s(x_s, a_s, w_s) \mid a_s = a \right] \right.$$

$$\left. + \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = a) \max_{\pi_{s+1},\dots} \mathbb{E}_{\pi_{s+1},\dots} \left[\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_{s+1} = y \right] \right\}$$

$$= \max_a \left\{ \mathbb{E}\left[r_s(x_s, a_s, w_s) \mid a_s = a \right] + \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = a)\ v_{*,s+1}(y) \right\}.$$

DP: The computation of $v_{*,s}(\cdot)$ can be done from $v_{*,s+1}(\cdot)$, and recursively until $v_{*,H}(\cdot) = R(\cdot)$. ☺: time$=O(|X|^2|A|H)$, for all x_0. Then, $pi_{*,s}(x)$ is the action a that minimizes the r.h.s.

[code]

Policy optimization by Value Iteration

$$v_{*,s}(x) = \max_{\pi_s,\dots} \mathbb{E}_{\pi_s,\dots} \left\{ \sum_{t=s}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x \right\}$$

$$= \max_{\pi_s,\pi_{s+1},\dots} \mathbb{E}_{\pi_s,\pi_{s+1},\dots} \Bigg\{ r_s(x_s, a_s, w_s)$$

$$+ \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = \pi_s(x_s)) \left(\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_{s+1} = y \right) \Bigg\}$$

$$= \max_a \Bigg\{ \mathbb{E}\left[r_s(x_s, a_s, w_s) \mid a_s = a \right]$$

$$+ \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = a) \max_{\pi_{s+1},\dots} \mathbb{E}_{\pi_{s+1},\dots} \left[\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_{s+1} = y \right] \Bigg\}$$

$$= \max_a \left\{ \mathbb{E}\left[r_s(x_s, a_s, w_s) \mid a_s = a \right] + \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = a)\ v_{*,s+1}(y) \right\}.$$

DP: The computation of $v_{*,s}(\cdot)$ can be done from $v_{*,s+1}(\cdot)$, and recursively until $v_{*,H}(\cdot) = R(\cdot)$. ☺: time$=O(|X|^2|A|H)$, for all x_0. Then, $pi_{*,s}(x)$ is the action a that minimizes the r.h.s.

[code]

Policy optimization by Value Iteration

$$v_{*,s}(x) = \max_{\pi_s,\dots} \mathbb{E}_{\pi_s,\dots} \left\{ \sum_{t=s}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x \right\}$$

$$= \max_{\pi_s,\pi_{s+1},\dots} \mathbb{E}_{\pi_s,\pi_{s+1},\dots} \Bigg\{ r_s(x_s, a_s, w_s)$$

$$+ \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = \pi_s(x_s)) \left(\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_{s+1} = y \right) \Bigg\}$$

$$= \max_a \Bigg\{ \mathbb{E}\big[r_s(x_s, a_s, w_s) \mid a_s = a \big]$$

$$+ \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = a) \max_{\pi_{s+1},\dots} \mathbb{E}_{\pi_{s+1},\dots} \left[\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_{s+1} = y \right] \Bigg\}$$

$$= \max_a \Big\{ \mathbb{E}\big[r_s(x_s, a_s, w_s) \mid a_s = a \big] + \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = a)\ v_{*,s+1}(y) \Big\}.$$

DP: The computation of $v_{*,s}(\cdot)$ can be done from $v_{*,s+1}(\cdot)$, and recursively until $v_{*,H}(\cdot) = R(\cdot)$. ☺: time$=O(|X|^2|A|H)$, for all x_0. Then, $pi_{*,s}(x)$ is the action a that minimizes the r.h.s.

[code]

Policy optimization by Value Iteration

$$v_{*,s}(x) = \max_{\pi_s,\dots} \mathbb{E}_{\pi_s,\dots} \left\{ \sum_{t=s}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_s = x \right\}$$

$$= \max_{\pi_s,\pi_{s+1},\dots} \mathbb{E}_{\pi_s,\pi_{s+1},\dots} \Bigg\{ r_s(x_s, a_s, w_s)$$

$$+ \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = \pi_s(x_s)) \left(\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_{s+1} = y \right) \Bigg\}$$

$$= \max_a \Bigg\{ \mathbb{E}\big[r_s(x_s, a_s, w_s) \mid a_s = a \big]$$

$$+ \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = a) \max_{\pi_{s+1},\dots} \mathbb{E}_{\pi_{s+1},\dots} \left[\sum_{t=s+1}^{H-1} r_t(x_t, a_t, w_t) + R(x_H) \mid x_{s+1} = y \right] \Bigg\}$$

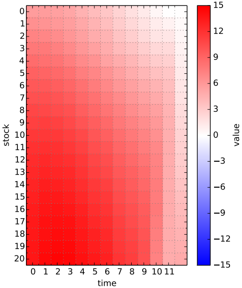
$$= \max_a \Big\{ \mathbb{E}\big[r_s(x_s, a_s, w_s) \mid a_s = a \big] + \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = a)\ v_{*,s+1}(y) \Big\}.$$

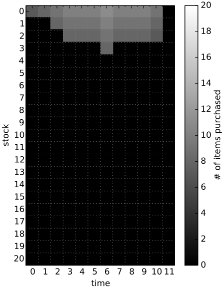
DP: The computation of $v_{*,s}(\cdot)$ can be done from $v_{*,s+1}(\cdot)$, and recursively until $v_{*,H}(\cdot) = R(\cdot)$. ☺: time$=O(|X|^2|A|H)$, for all x_0. Then, $pi_{*,s}(x)$ is the action a that minimizes the r.h.s.

[code]

Example: the Retail Store Management Problem

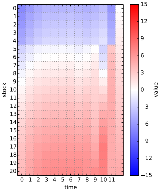
Optimal value and policy

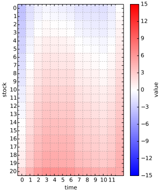


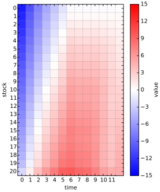


vs

values of policies $\pi^{(1)}, \pi^{(2)}, \pi^{(3)}$







Bellman's principle of optimality

- The recursive identities (recall that $v_{*,s}(\cdot) = v_{\pi_*,0}(\cdot)$)

$$v_{*,s}(x) = \max_a \Big\{ \mathbb{E}\big[r_s(x_s, a_s, w_s) \mid a_s = a\big] + \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = a)\; v_{*,s+1}(y) \Big\}$$

$$= \mathbb{E}\big[r_s(x_s, a_s, w_s) \mid a_s = \pi_{*,s}(x_s)\big] + \sum_y \mathbb{P}(x_{s+1} = y | x_s = x, a_s = \pi_{*,s}(x_s))\; v_{*,s+1}(y)$$

  are called Bellman equations.
- The tail policy is optimal for the tail subproblem (optimization of the future does not depend on what we did in the past)
- At each step, DP solves ALL the tail subroblems tail subproblems of a given time length, using the solution of the tail subproblems of shorter time length

Outline for Part 1

- Finite-Horizon Optimal Control
  - Problem definition
  - Policy evaluation: Value Iteration[1]
  - Policy optimization: Value Iteration[2]
- Stationary Infinite-Horizon Optimal Control
  - Bellman operators
  - Contraction Mappings
  - Stationary policies
  - Policy evaluation
  - Policy optimization: Value Iteration[3], Policy Iteration, Modified/Optimistic Policy Iteration
  - Asynchronous Algorithms
  - Learning from samples: Real-Time Dynamic Programming, Q-Learning, TD-Learning, SARSA

Infinite-Horizon Optimal Control Problem

- Same as finite-horizon (Markov Decision Process), but:
  - the number of stages is infinite
  - the system is stationary ($f_t = f$, $w_t \sim w$, $r_t = r$)

  $x_{t+1} = f(x_t, a_t, w_t)$ $\left[\Leftrightarrow \ \mathbb{P}(x_{t+1} = x' | x_t = x, a_t = a) = p(x, a, x') \right]$
- Find a policy $\pi_0^\infty = (\pi_0, \pi_1, \dots)$ that maximizes (for all x)

$$v_{\pi_0^\infty}(x) = \lim_{H \to \infty} \mathbb{E}\left\{ \sum_{t=0}^{H-1} \gamma^t r(x_t, a_t, w_t) \mid x_0 = x \right\}$$

- $\gamma \in (0, 1)$ is called the discount factor
  - Discounted problems ($\gamma < 1$, $|r| \leq M < \infty$, $v \leq \frac{M}{1-\gamma}$)
  - Stochastic shortest path problems ($\gamma = 1$ with a termination state reached with probability 1) (sparingly covered)
- Stationary policies $\pi = (\pi, \pi, \dots)$ play a central role

We will not cover the average reward criterion $\lim_{H\to\infty} \frac{1}{H}\mathbb{E}\left\{\sum_{t=0}^{H-1} r_t(x_t, a_t, w_t)\right\}$ nor unbounded rewards...

Infinite-Horizon Optimal Control Problem

- Same as finite-horizon (Markov Decision Process), but:
  - the number of stages is infinite
  - the system is stationary ($f_t = f$, $w_t \sim w$, $r_t = r$)

$$x_{t+1} = f(x_t, a_t, w_t) \; \left[\Leftrightarrow \; \mathbb{P}(x_{t+1} = x' | x_t = x, a_t = a) = p(x, a, x') \right]$$

- Find a policy $\pi_0^\infty = (\pi_0, \pi_1, \dots)$ that maximizes (for all x)

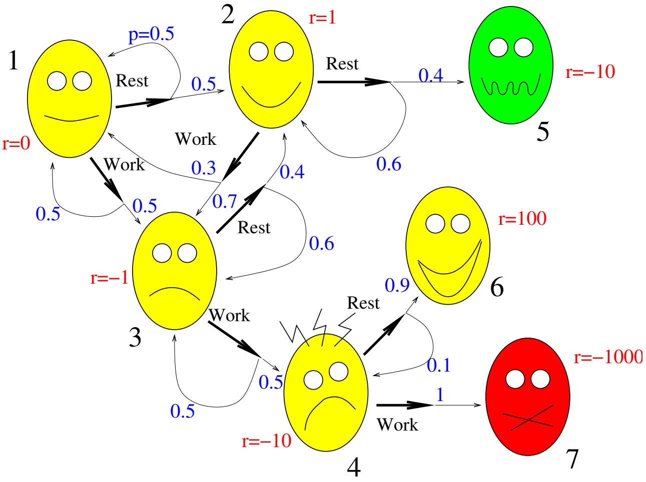
$$v_{\pi_0^\infty}(x) = \lim_{H \to \infty} \mathbb{E} \left\{ \sum_{t=0}^{H-1} \gamma^t r(x_t, a_t, w_t) \mid x_0 = x \right\}$$

- $\gamma \in (0, 1)$ is called the discount factor
  - Discounted problems ($\gamma < 1$, $|r| \leq M < \infty$, $v \leq \frac{M}{1-\gamma}$)
  - Stochastic shortest path problems ($\gamma = 1$ with a termination state reached with probability 1) (sparingly covered)
- Stationary policies $\pi = (\pi, \pi, \dots)$ play a central role

We will not cover the average reward criterion $\lim_{H \to \infty} \frac{1}{H} \mathbb{E} \left\{ \sum_{t=0}^{H-1} r_t(x_t, a_t, w_t) \right\}$ nor unbounded rewards...

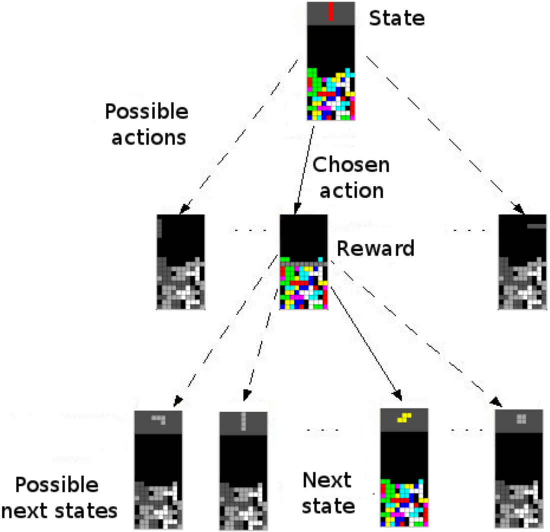
Example: Student Dilemma

Stationary MDPs naturally represented as a graph:



States x_5, x_6, x_7 are terminal. Whatever the policy, they are reached in finite time with probability 1 so we can take $\gamma = 1$.

Example: Tetris



Example: the Retail Store Management Problem

Each month t, a store contains x_t items (maximum capacity M) of a specific goods and the demand for that goods is w_t. At the end of each month the manager of the store can order a_t more items from his supplier. The cost of maintaining an inventory of x is $h(x)$. The cost to order a items is $C(a)$. The income for selling q items is $f(q)$. If the demand w is bigger than the available inventory x, customers that cannot be served leave. ~~The value of the remaining inventory at the end of the year is $g(x)$~~. $\boxed{\text{The rate of inflation is } \alpha = 3\% = 0.03.}$

$M = 20$, $f(x) = x$, $g(x) = 0.25x$, $h(x) = 0.25x$, $C(a) = (1 + 0.5a)\mathbb{1}_{a>0}$, $\boxed{w_t \sim \mathrm{U}(\{5, 6, \ldots, 15\}),\ \gamma = \frac{1}{1+\alpha}}$

- $\boxed{t = 0, 1, \ldots}$
- State space: $x \in X = \{0, 1, \ldots, M\}$
- Action space: At state x, $a \in A(x) = \{0, 1, \ldots, M - x\}$
- Dynamics: $x_{t+1} = [x_t + a_t - w_t]^+$
- Reward: $r(x_t, a_t, w_t) = -C(a_t) - h(x_t + a_t) + f([x_t + a_t - x_{t+1}]^+)$ and $R(x) = g(x)$.

Bellman operators (I)

- For any function v of x, denote,

$$\forall x, \quad (Tv)(x) = \max_a \mathbb{E}\big[r(x,a,w)\big] + \mathbb{E}\big[\gamma v(f(x,a,w))\big]$$
$$= \max_a \quad r(x,a) \qquad + \gamma \sum_y \mathbb{P}(y|x,a)v(y)$$

- Tv is the optimal value for the one-stage problem with stage reward r and terminal reward $R = \gamma v$.
- T operates on bounded functions of x to produce other bounded functions of x.
- For any stationary policy π and v, denote

$$(T_\pi v)(x) = r(x,\pi(x)) + \gamma \sum_y \mathbb{P}(y|x,\pi(x))v(y), \quad \forall x$$

- $T_\pi v$ is the value of π for the same one-stage problem
- The critical structure of the problem is captured in T and T_π and most of the theory of discounted problems can be developed using these two (Bellman) operators.

Bellman operators (I)

- For any function v of x, denote,

$$\forall x, \quad (Tv)(x) = \max_a \mathbb{E}\left[r(x,a,w)\right] + \mathbb{E}\left[\gamma v(f(x,a,w))\right]$$
$$= \max_a \quad r(x,a) \qquad + \gamma \sum_y \mathbb{P}(y|x,a)v(y)$$

- Tv is the optimal value for the one-stage problem with stage reward r and terminal reward $R = \gamma v$.
- T operates on bounded functions of x to produce other bounded functions of x.
- For any stationary policy π and v, denote

$$(T_\pi v)(x) = r(x,\pi(x)) + \gamma \sum_y \mathbb{P}(y|x,\pi(x))v(y), \quad \forall x$$

- $T_\pi v$ is the value of π for the same one-stage problem
- The critical structure of the problem is captured in T and T_π and most of the theory of discounted problems can be developed using these two (Bellman) operators.

Bellman operators (I)

- For any function v of x, denote,

$$\forall x, \quad (Tv)(x) = \max_a \mathbb{E}\big[r(x,a,w)\big] + \mathbb{E}\big[\gamma v(f(x,a,w))\big]$$
$$= \max_a \quad r(x,a) \qquad + \gamma \sum_y \mathbb{P}(y|x,a)v(y)$$

- Tv is the optimal value for the one-stage problem with stage reward r and terminal reward $R = \gamma v$.
- T operates on bounded functions of x to produce other bounded functions of x.
- For any stationary policy π and v, denote

$$(T_\pi v)(x) = r(x, \pi(x)) + \gamma \sum_y \mathbb{P}(y|x, \pi(x))v(y), \quad \forall x$$

- $T_\pi v$ is the value of π for the same one-stage problem
- The critical structure of the problem is captured in T and T_π and most of the theory of discounted problems can be developed using these two (Bellman) operators.

Bellman operators (II)

- Consider the H-stage policy $\pi_0^H = (\pi_0, \pi_1, \dots, \pi_{H-1})$ with no terminal reward $R = 0$
- For $0 \le s \le H$, consider the $(H-s)$-stage "tail" policy $\pi_s^H = (\pi_s, \pi_{s+1}, \dots, \pi_{H-1})$ with $R = 0$

$$\begin{aligned}
v_{\pi_0^H}(x) &= \mathbb{E}_{x_0=x}\left[\sum_{t=0}^{H-1} \gamma^t r(x_t, \pi_t(x_t), w_t)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma \sum_{t=1}^{H-1} \gamma^t r(x_t, \pi_t(x_t), w_t)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma \left(\sum_{t=1}^{H-1} \gamma^{t-1} r(x_t, \pi_t(x_t), w_t)\right)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma v_{\pi_1^H}(x_1)\right] \\
&= (T_{\pi_0} v_{\pi_1^H})(x)
\end{aligned}$$

- By induction ($v_{\pi_H^H} = 0$), we get for all x,

$$v_{\pi_0^H}(x) = (T_{\pi_0} T_{\pi_1} \dots T_{\pi_{H-1}} 0)(x) \xrightarrow{H \to \infty} v_{\pi_0^\infty}(x)$$

Bellman operators (II)

- Consider the H-stage policy $\pi_0^H = (\pi_0, \pi_1, \ldots, \pi_{H-1})$ with no terminal reward $R = 0$
- For $0 \leq s \leq H$, consider the $(H-s)$-stage "tail" policy $\pi_s^H = (\pi_s, \pi_{s+1}, \ldots, \pi_{H-1})$ with $R = 0$

$$
\begin{aligned}
v_{\pi_0^H}(x) &= \mathbb{E}_{x_0=x}\left[\sum_{t=0}^{H-1} \gamma^t r(x_t, \pi_t(x_t), w_t)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma \sum_{t=1}^{H-1} \gamma^t r(x_t, \pi_t(x_t), w_t)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma \left(\sum_{t=1}^{H-1} \gamma^{t-1} r(x_t, \pi_t(x_t), w_t)\right)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma v_{\pi_1^H}(x_1)\right] \\
&= (T_{\pi_0} v_{\pi_1^H})(x)
\end{aligned}
$$

- By induction ($v_{\pi_H^H} = 0$), we get for all x,

$$
v_{\pi_0^H}(x) = (T_{\pi_0} T_{\pi_1} \ldots T_{\pi_{H-1}} 0)(x) \xrightarrow{H \to \infty} v_{\pi_0^\infty}(x)
$$

Bellman operators (II)

- Consider the H-stage policy $\pi_0^H = (\pi_0, \pi_1, \dots, \pi_{H-1})$ with no terminal reward $R = 0$
- For $0 \leq s \leq H$, consider the $(H-s)$-stage "tail" policy $\pi_s^H = (\pi_s, \pi_{s+1}, \dots, \pi_{H-1})$ with $R = 0$

$$
\begin{aligned}
v_{\pi_0^H}(x) &= \mathbb{E}_{x_0=x}\left[\sum_{t=0}^{H-1} \gamma^t r(x_t, \pi_t(x_t), w_t)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma \sum_{t=1}^{H-1} \gamma^t r(x_t, \pi_t(x_t), w_t)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma \left(\sum_{t=1}^{H-1} \gamma^{t-1} r(x_t, \pi_t(x_t), w_t)\right)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma v_{\pi_1^H}(x_1)\right] \\
&= (T_{\pi_0} v_{\pi_1^H})(x)
\end{aligned}
$$

- By induction ($v_{\pi_H^H} = 0$), we get for all x,

$$
v_{\pi_0^H}(x) = (T_{\pi_0} T_{\pi_1} \dots T_{\pi_{H-1}} 0)(x) \xrightarrow{H \to \infty} v_{\pi_0^\infty}(x)
$$

Bellman operators (II)

- Consider the H-stage policy $\pi_0^H = (\pi_0, \pi_1, \dots, \pi_{H-1})$ with no terminal reward $R = 0$
- For $0 \leq s \leq H$, consider the $(H-s)$-stage "tail" policy $\pi_s^H = (\pi_s, \pi_{s+1}, \dots, \pi_{H-1})$ with $R = 0$

$$
\begin{aligned}
v_{\pi_0^H}(x) &= \mathbb{E}_{x_0=x}\left[\sum_{t=0}^{H-1} \gamma^t r(x_t, \pi_t(x_t), w_t)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma \sum_{t=1}^{H-1} \gamma^t r(x_t, \pi_t(x_t), w_t)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma \left(\sum_{t=1}^{H-1} \gamma^{t-1} r(x_t, \pi_t(x_t), w_t)\right)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma v_{\pi_1^H}(x_1)\right] \\
&= (T_{\pi_0} v_{\pi_1^H})(x)
\end{aligned}
$$

- By induction ($v_{\pi_H^H} = 0$), we get for all x,

$$
v_{\pi_0^H}(x) = (T_{\pi_0} T_{\pi_1} \dots T_{\pi_{H-1}} 0)(x) \xrightarrow{H \to \infty} v_{\pi_0^\infty}(x)
$$

Bellman operators (II)

- Consider the H-stage policy $\pi_0^H = (\pi_0, \pi_1, \ldots, \pi_{H-1})$ with no terminal reward $R = 0$
- For $0 \leq s \leq H$, consider the $(H-s)$-stage "tail" policy $\pi_s^H = (\pi_s, \pi_{s+1}, \ldots, \pi_{H-1})$ with $R = 0$

$$
\begin{aligned}
v_{\pi_0^H}(x) &= \mathbb{E}_{x_0=x}\left[\sum_{t=0}^{H-1} \gamma^t r(x_t, \pi_t(x_t), w_t)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma \sum_{t=1}^{H-1} \gamma^t r(x_t, \pi_t(x_t), w_t)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma \left(\sum_{t=1}^{H-1} \gamma^{t-1} r(x_t, \pi_t(x_t), w_t)\right)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma v_{\pi_1^H}(x_1)\right] \\
&= (T_{\pi_0} v_{\pi_1^H})(x)
\end{aligned}
$$

- By induction ($v_{\pi_H^H} = 0$), we get for all x,

$$
v_{\pi_0^H}(x) = (T_{\pi_0} T_{\pi_1} \ldots T_{\pi_{H-1}} 0)(x) \xrightarrow{H \to \infty} v_{\pi_0^\infty}(x)
$$

Bellman operators (II)

- Consider the H-stage policy $\pi_0^H = (\pi_0, \pi_1, \ldots, \pi_{H-1})$ with no terminal reward $R = 0$
- For $0 \leq s \leq H$, consider the $(H-s)$-stage "tail" policy $\pi_s^H = (\pi_s, \pi_{s+1}, \ldots, \pi_{H-1})$ with $R = 0$

$$\begin{aligned}
v_{\pi_0^H}(x) &= \mathbb{E}_{x_0=x}\left[\sum_{t=0}^{H-1} \gamma^t r(x_t, \pi_t(x_t), w_t)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma \sum_{t=1}^{H-1} \gamma^t r(x_t, \pi_t(x_t), w_t)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma \left(\sum_{t=1}^{H-1} \gamma^{t-1} r(x_t, \pi_t(x_t), w_t)\right)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma v_{\pi_1^H}(x_1)\right] \\
&= (T_{\pi_0} v_{\pi_1^H})(x)
\end{aligned}$$

- By induction ($v_{\pi_H^H} = 0$), we get for all x,

$$v_{\pi_0^H}(x) = (T_{\pi_0} T_{\pi_1} \ldots T_{\pi_{H-1}} 0)(x) \xrightarrow{H \to \infty} v_{\pi_0^\infty}(x)$$

Bellman operators (II)

- Consider the H-stage policy $\pi_0^H = (\pi_0, \pi_1, \ldots, \pi_{H-1})$ with no terminal reward $R = 0$
- For $0 \le s \le H$, consider the $(H-s)$-stage "tail" policy $\pi_s^H = (\pi_s, \pi_{s+1}, \ldots, \pi_{H-1})$ with $R = 0$

$$
\begin{aligned}
v_{\pi_0^H}(x) &= \mathbb{E}_{x_0=x}\left[\sum_{t=0}^{H-1} \gamma^t r(x_t, \pi_t(x_t), w_t)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma \sum_{t=1}^{H-1} \gamma^t r(x_t, \pi_t(x_t), w_t)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma \left(\sum_{t=1}^{H-1} \gamma^{t-1} r(x_t, \pi_t(x_t), w_t)\right)\right] \\
&= \mathbb{E}_{x_0=x}\left[r(x_0, \pi_0(x_0), w_0) + \gamma v_{\pi_1^H}(x_1)\right] \\
&= (T_{\pi_0} v_{\pi_1^H})(x)
\end{aligned}
$$

- By induction ($v_{\pi_H^H} = 0$), we get for all x,

$$
v_{\pi_0^H}(x) = (T_{\pi_0} T_{\pi_1} \ldots T_{\pi_{H-1}} 0)(x) \quad \xrightarrow{H \to \infty} \quad v_{\pi_0^\infty}(x)
$$

Bellman operators (III)

- Similarly, the optimal H-stage value function with terminal reward $R = 0$ is $T^H 0$.
- Fortunately, it can be shown that

$$v_* = \max_{\pi_0^\infty} v_{\pi_0^\infty} = \max_{\pi_0^\infty} \lim_{H\to\infty} v_{\pi_0^H} \overset{(*)}{=} \lim_{H\to\infty} \max_{\pi_0^H} v_{\pi_0^H} = \lim_{H\to\infty} T^H 0,$$

i.e, the infinite-horizon problem is the limit of the H-horizon problem when the horizon H tends to ∞

(*) For any policy $\pi_0^\infty = (\pi_0, \pi_1, \dots)$, and any initial state x_0,

$$\begin{aligned} v_{\pi_0^\infty}(x_0) &= \mathbb{E}_{x_0=x}\left[\sum_{t=0}^{\infty} \gamma^t r(x_t, \pi_t(x_t), w_t)\right] \\ &= \underbrace{\mathbb{E}_{x_0=x}\left[\sum_{t=0}^{H-1} \gamma^t r(x_t, \pi_t(x_t), w_t)\right]}_{T_{\pi_0} T_{\pi_1} \dots T_{\pi_{H-1}} 0} + \underbrace{\mathbb{E}_{x_0=x}\left[\sum_{t=H}^{\infty} \gamma^t r(x_t, \pi_t(x_t), w_t)\right]}_{\left|\cdot\right| \le \sum_{t=H}^{\infty} \gamma^t M \le \frac{\gamma^H M}{1-\gamma}} \end{aligned}$$

$$\overset{\max}{\Rightarrow} v_*(x_0) = (T^H 0)(x) + O(\gamma^H)$$

Bellman operators (III)

- Similarly, the optimal H-stage value function with terminal reward $R = 0$ is $T^H 0$.
- Fortunately, it can be shown that

$$v_* = \max_{\pi_0^\infty} v_{\pi_0^\infty} = \max_{\pi_0^\infty} \lim_{H\to\infty} v_{\pi_0^H} \overset{(*)}{=} \lim_{H\to\infty} \max_{\pi_0^H} v_{\pi_0^H} = \lim_{H\to\infty} T^H 0,$$

i.e, the infinite-horizon problem is the limit of the H-horizon problem when the horizon H tends to ∞

(*) For any policy $\pi_0^\infty = (\pi_0, \pi_1, \dots)$, and any initial state x_0,

$$\begin{aligned} v_{\pi_0^\infty}(x_0) &= \mathbb{E}_{x_0=x}\left[\sum_{t=0}^{\infty} \gamma^t r(x_t, \pi_t(x_t), w_t)\right] \\ &= \underbrace{\mathbb{E}_{x_0=x}\left[\sum_{t=0}^{H-1} \gamma^t r(x_t, \pi_t(x_t), w_t)\right]}_{T_{\pi_0} T_{\pi_1} \dots T_{\pi_{H-1}} 0} + \underbrace{\mathbb{E}_{x_0=x}\left[\sum_{t=H}^{\infty} \gamma^t r(x_t, \pi_t(x_t), w_t)\right]}_{\left|\ \cdot\ \right| \le \sum_{t=H}^{\infty} \gamma^t M \le \frac{\gamma^H M}{1-\gamma}} \end{aligned}$$

$$\overset{\max}{\Rightarrow} v_*(x_0) = (T^H 0)(x) + O(\gamma^H)$$

Bellman operators (III)

- Similarly, the optimal H-stage value function with terminal reward $R = 0$ is $T^H 0$.
- Fortunately, it can be shown that

$$v_* = \max_{\pi_0^\infty} v_{\pi_0^\infty} = \max_{\pi_0^\infty} \lim_{H\to\infty} v_{\pi_0^H} \overset{(*)}{=} \lim_{H\to\infty} \max_{\pi_0^H} v_{\pi_0^H} = \lim_{H\to\infty} T^H 0,$$

i.e, the infinite-horizon problem is the limit of the H-horizon problem when the horizon H tends to ∞

(*) For any policy $\pi_0^\infty = (\pi_0, \pi_1, \dots)$, and any initial state x_0,

$$\begin{aligned} v_{\pi_0^\infty}(x_0) &= \mathbb{E}_{x_0=x}\left[\sum_{t=0}^{\infty} \gamma^t r(x_t, \pi_t(x_t), w_t)\right] \\ &= \underbrace{\mathbb{E}_{x_0=x}\left[\sum_{t=0}^{H-1} \gamma^t r(x_t, \pi_t(x_t), w_t)\right]}_{T_{\pi_0} T_{\pi_1} \dots T_{\pi_{H-1}} 0} + \underbrace{\mathbb{E}_{x_0=x}\left[\sum_{t=H}^{\infty} \gamma^t r(x_t, \pi_t(x_t), w_t)\right]}_{\left|\ \cdot\ \right| \le \sum_{t=H}^{\infty} \gamma^t M \le \frac{\gamma^H M}{1-\gamma}} \end{aligned}$$

$$\overset{\max}{\Rightarrow} v_*(x_0) = (T^H 0)(x) + O(\gamma^H)$$

Bellman operators (III)

- Similarly, the optimal H-stage value function with terminal reward $R = 0$ is $T^H 0$.
- Fortunately, it can be shown that

$$v_* = \max_{\pi_0^\infty} v_{\pi_0^\infty} = \max_{\pi_0^\infty} \lim_{H\to\infty} v_{\pi_0^H} \overset{(*)}{=} \lim_{H\to\infty} \max_{\pi_0^H} v_{\pi_0^H} = \lim_{H\to\infty} T^H 0,$$

i.e, the infinite-horizon problem is the limit of the H-horizon problem when the horizon H tends to ∞

(*) For any policy $\pi_0^\infty = (\pi_0, \pi_1, \dots)$, and any initial state x_0,

$$\begin{aligned} v_{\pi_0^\infty}(x_0) &= \mathbb{E}_{x_0=x}\left[\sum_{t=0}^{\infty} \gamma^t r(x_t, \pi_t(x_t), w_t)\right] \\ &= \underbrace{\mathbb{E}_{x_0=x}\left[\sum_{t=0}^{H-1} \gamma^t r(x_t, \pi_t(x_t), w_t)\right]}_{T_{\pi_0} T_{\pi_1} \dots T_{\pi_{H-1}} 0} + \underbrace{\mathbb{E}_{x_0=x}\left[\sum_{t=H}^{\infty} \gamma^t r(x_t, \pi_t(x_t), w_t)\right]}_{\left|\ \cdot\ \right| \le \sum_{t=H}^{\infty} \gamma^t M \le \frac{\gamma^H M}{1-\gamma}} \end{aligned}$$

$$\overset{\max}{\Rightarrow} v_*(x_0) = (T^H 0)(x) + O(\gamma^H)$$

The contraction property

Theorem

T and T_π are γ-contraction mappings for the max norm $\|\cdot\|_\infty$.

where for all function v, $\|v\|_\infty = \max_x |v(x)|$, and an operator F is a γ-contraction mapping for that norm iff:

$$\forall v_1, v_2, \quad \|F v_1 - F v_2\|_\infty \le \gamma \|v_1 - v_2\|_\infty .$$

Proof (for T): By using $|\max_a f(a) - \max_a g(a)| \le \max_a |f(a) - g(a)|$,

$$\max_x \left| \overbrace{\max_a \left\{ r(x,a) + \gamma \sum_y \mathbb{P}(y|x,a) v_1(x) \right\}}^{(T v_1)(x)} - \overbrace{\max_a \left\{ r(x,a) + \gamma \sum_y \mathbb{P}(y|x,a) v_2(y) \right\}}^{(T v_2)(x)} \right|$$

$$\le \max_x \left| \max_a \gamma \sum_y \mathbb{P}(y|x,a)(v_1(x) - v_2(x)) \right| \le \max_x \max_a \gamma \sum_y \mathbb{P}(y|x,a) \|v_1 - v_2\|_\infty = \gamma \|v_1 - v_2\|_\infty .$$

- By Banach fixed point theorem, F has one and only one fixed point f^* to which the sequence $f_n = F f_{n-1} = F^n f_0$ converges for any f_0.
- $v_* = T v_*$, and for any stationary policy π, $v_\pi = T_\pi v_\pi$.

The contraction property

Theorem

T and T_π are γ-contraction mappings for the max norm $\|\cdot\|_\infty$.

where for all function v, $\|v\|_\infty = \max_x |v(x)|$, and an operator F is a γ-contraction mapping for that norm iff:

$$\forall v_1, v_2, \quad \|Fv_1 - Fv_2\|_\infty \le \gamma \|v_1 - v_2\|_\infty.$$

Proof (for T): By using $|\max_a f(a) - \max_a g(a)| \le \max_a |f(a) - g(a)|$,

$$\max_x \Big| \overbrace{\max_a \Big\{ r(x,a) + \gamma \sum_y \mathbb{P}(y|x,a) v_1(x) \Big\}}^{(Tv_1)(x)} - \overbrace{\max_a \Big\{ r(x,a) + \gamma \sum_y \mathbb{P}(y|x,a) v_2(y) \Big\}}^{(Tv_2)(x)} \Big|$$

$$\le \max_x \Big| \max_a \gamma \sum_y \mathbb{P}(y|x,a)(v_1(x) - v_2(x)) \Big| \le \max_x \max_a \gamma \sum_y \mathbb{P}(y|x,a) \|v_1 - v_2\|_\infty = \gamma \|v_1 - v_2\|_\infty.$$

- By Banach fixed point theorem, F has one and only one fixed point f^* to which the sequence $f_n = Ff_{n-1} = F^n f_0$ converges for any f_0.
- $v_* = Tv_*$, and for any stationary policy π, $v_\pi = T_\pi v_\pi$.

The contraction property

Theorem

T and T_π are γ-contraction mappings for the max norm $\|\cdot\|_\infty$.

where for all function v, $\|v\|_\infty = \max_x |v(x)|$, and an operator F is a γ-contraction mapping for that norm iff:

$$\forall v_1, v_2, \quad \|F v_1 - F v_2\|_\infty \leq \gamma \|v_1 - v_2\|_\infty.$$

Proof (for T): By using $|\max_a f(a) - \max_a g(a)| \leq \max_a |f(a) - g(a)|$,

$$\max_x \left| \overbrace{\max_a \left\{ r(x,a) + \gamma \sum_y \mathbb{P}(y|x,a) v_1(x) \right\}}^{(T v_1)(x)} - \overbrace{\max_a \left\{ r(x,a) + \gamma \sum_y \mathbb{P}(y|x,a) v_2(y) \right\}}^{(T v_2)(x)} \right|$$

$$\leq \max_x \left| \max_a \gamma \sum_y \mathbb{P}(y|x,a)(v_1(x) - v_2(x)) \right| \leq \max_x \max_a \gamma \sum_y \mathbb{P}(y|x,a) \|v_1 - v_2\|_\infty = \gamma \|v_1 - v_2\|_\infty.$$

- By Banach fixed point theorem, F has one and only one fixed point f^* to which the sequence $f_n = F f_{n-1} = F^n f_0$ converges for any f_0.
- $v_* = T v_*$, and for any stationary policy π, $v_\pi = T_\pi v_\pi$.

The contraction property

Theorem

T and T_π are γ-contraction mappings for the max norm $\|\cdot\|_\infty$.

where for all function v, $\|v\|_\infty = \max_x |v(x)|$, and an operator F is a γ-contraction mapping for that norm iff:

$$\forall v_1, v_2, \quad \|F v_1 - F v_2\|_\infty \le \gamma \|v_1 - v_2\|_\infty.$$

Proof (for T): By using $|\max_a f(a) - \max_a g(a)| \le \max_a |f(a) - g(a)|$,

$$\max_x \left| \overbrace{\max_a \left\{ r(x,a) + \gamma \sum_y \mathbb{P}(y|x,a) v_1(x) \right\}}^{(T v_1)(x)} - \overbrace{\max_a \left\{ r(x,a) + \gamma \sum_y \mathbb{P}(y|x,a) v_2(y) \right\}}^{(T v_2)(x)} \right|$$

$$\le \max_x \left| \max_a \gamma \sum_y \mathbb{P}(y|x,a)(v_1(x) - v_2(x)) \right| \le \max_x \max_a \gamma \sum_y \mathbb{P}(y|x,a) \|v_1 - v_2\|_\infty = \gamma \|v_1 - v_2\|_\infty.$$

- By Banach fixed point theorem, F has one and only one fixed point f^* to which the sequence $f_n = F f_{n-1} = F^n f_0$ converges for any f_0.
- $v_* = T v_*$, and for any stationary policy π, $v_\pi = T_\pi v_\pi$.

The contraction property

Theorem

T and T_π are γ-contraction mappings for the max norm $\|\cdot\|_\infty$.

where for all function v, $\|v\|_\infty = \max_x |v(x)|$, and an operator F is a γ-contraction mapping for that norm iff:

$$\forall v_1, v_2, \quad \|F v_1 - F v_2\|_\infty \leq \gamma \|v_1 - v_2\|_\infty.$$

Proof (for T): By using $|\max_a f(a) - \max_a g(a)| \leq \max_a |f(a) - g(a)|$,

$$\max_x \Bigg| \overbrace{\max_a \Big\{ r(x,a) + \gamma \sum_y \mathbb{P}(y|x,a) v_1(x) \Big\}}^{(T v_1)(x)} - \overbrace{\max_a \Big\{ r(x,a) + \gamma \sum_y \mathbb{P}(y|x,a) v_2(y) \Big\}}^{(T v_2)(x)} \Bigg|$$

$$\leq \max_x \Big| \max_a \gamma \sum_y \mathbb{P}(y|x,a)(v_1(x) - v_2(x)) \Big| \leq \max_x \max_a \gamma \sum_y \mathbb{P}(y|x,a) \|v_1 - v_2\|_\infty = \gamma \|v_1 - v_2\|_\infty.$$

- By Banach fixed point theorem, F has one and only one fixed point f^* to which the sequence $f_n = F f_{n-1} = F^n f_0$ converges for any f_0.
- $v_* = T v_*$, and for any stationary policy π, $v_\pi = T_\pi v_\pi$.

There exists an optimal stationary policy

Theorem

A stationary policy π is optimal if and only if for all x, $\pi(x)$ attains the maximum in Bellman's optimality equation $v_* = Tv_*$, i.e.

$$\forall x, \quad \pi(x) \in \arg\max_a \left\{ r(x,a) + \sum_y \mathbb{P}(y|x,a) v_*(y) \right\}$$

or equivalently $T_\pi v_* = T v_*$

In the sequel, for any function v (not necessarily v_*!), we shall say that π is greedy with respect to v when $T_\pi v = T v$, and write $\pi = \mathcal{G} v$.
$\Rightarrow$ A policy π_* is optimal iff $\pi_* = \mathcal{G} v_*$.

Proof: (1) Let π be such that $T_\pi v_* = T v_*$. Since $v_* = T v_*$, we have $v_* = T_\pi v_*$, and by the uniqueness of the fixed point of T_π (which is v_π), then $v_\pi = v_*$.
(2) Let π be optimal. This means $v_\pi = v_*$. Since $v_\pi = T_\pi v_\pi$, we have $v_* = T_\pi v_*$ and the result follows from $v_* = T v_*$.

There exists an optimal stationary policy

Theorem

A stationary policy π is optimal if and only if for all x, $\pi(x)$ attains the maximum in Bellman's optimality equation $v_* = T v_*$, i.e.

$$\forall x, \quad \pi(x) \in \arg\max_a \left\{ r(x,a) + \sum_y \mathbb{P}(y|x,a) v_*(y) \right\}$$

or equivalently $T_\pi v_* = T v_*$

In the sequel, for any function v (not necessarily v_*!), we shall say that π is greedy with respect to v when $T_\pi v = T v$, and write $\pi = \mathcal{G} v$.
$\Rightarrow$ A policy π_* is optimal iff $\pi_* = \mathcal{G} v_*$.

Proof: (1) Let π be such that $T_\pi v_* = T v_*$. Since $v_* = T v_*$, we have $v_* = T_\pi v_*$, and by the uniqueness of the fixed point of T_π (which is v_π), then $v_\pi = v_*$.

(2) Let π be optimal. This means $v_\pi = v_*$. Since $v_\pi = T_\pi v_\pi$, we have $v_* = T_\pi v_*$ and the result follows from $v_* = T v_*$.

There exists an optimal stationary policy

Theorem

A stationary policy π is optimal if and only if for all x, $\pi(x)$ attains the maximum in Bellman's optimality equation $v_* = T v_*$, i.e.

$$\forall x, \quad \pi(x) \in \arg\max_a \left\{ r(x,a) + \sum_y \mathbb{P}(y|x,a) v_*(y) \right\}$$

or equivalently $T_\pi v_* = T v_*$

In the sequel, for any function v (not necessarily v_*!), we shall say that π is greedy with respect to v when $T_\pi v = T v$, and write $\pi = \mathcal{G} v$.
$\Rightarrow$ A policy π_* is optimal iff $\pi_* = \mathcal{G} v_*$.

Proof: (1) Let π be such that $T_\pi v_* = T v_*$. Since $v_* = T v_*$, we have $v_* = T_\pi v_*$, and by the uniqueness of the fixed point of T_π (which is v_π), then $v_\pi = v_*$.
(2) Let π be optimal. This means $v_\pi = v_*$. Since $v_\pi = T_\pi v_\pi$, we have $v_* = T_\pi v_*$ and the result follows from $v_* = T v_*$.

A few comments

- The space of stationary policies is much smaller than the space of non-stationary policies. If the state and action spaces are finite, then it is finite ($|A|^{|X|}$).
- Solving an infinite-horizon problem essentially amounts to find the optimal value function v_*, i.e. to solve the fixed point equation $v_* = T v_*$ (then take any policy $\pi \in \mathcal{G} v_*$)
- We already have an algorithm: for any v_0,

$$v_{k+1} \leftarrow T v_k \qquad \text{(Value Iteration)}$$

  converges asymptotically to the optimal value v_*
- Convergence rate is at least linear:

$$\|v_* - v_{k+1}\|_\infty = \|T v_* - T v_k\|_\infty \le \gamma \|v_* - v_k\|_\infty.$$

A few comments

- The space of stationary policies is much smaller than the space of non-stationary policies. If the state and action spaces are finite, then it is finite ($|A|^{|X|}$).
- Solving an infinite-horizon problem essentially amounts to find the optimal value function v_*, i.e. to solve the fixed point equation $v_* = T v_*$ (then take any policy $\pi \in \mathcal{G} v_*$)
- We already have an algorithm: for any v_0,

$$v_{k+1} \leftarrow T v_k \qquad \text{(Value Iteration)}$$

  converges asymptotically to the optimal value v_*
- Convergence rate is at least linear:

$$\|v_* - v_{k+1}\|_\infty = \|T v_* - T v_k\|_\infty \leq \gamma \|v_* - v_k\|_\infty.$$

A few comments

- The space of stationary policies is much smaller than the space of non-stationary policies. If the state and action spaces are finite, then it is finite ($|A|^{|X|}$).
- Solving an infinite-horizon problem essentially amounts to find the optimal value function v_*, i.e. to solve the fixed point equation $v_* = T v_*$ (then take any policy $\pi \in \mathcal{G} v_*$)
- We already have an algorithm: for any v_0,

$$v_{k+1} \leftarrow T v_k \qquad \text{(Value Iteration)}$$

converges asymptotically to the optimal value v_*

- Convergence rate is at least linear:

$$\|v_* - v_{k+1}\|_\infty = \|T v_* - T v_k\|_\infty \leq \gamma \|v_* - v_k\|_\infty.$$

A few comments

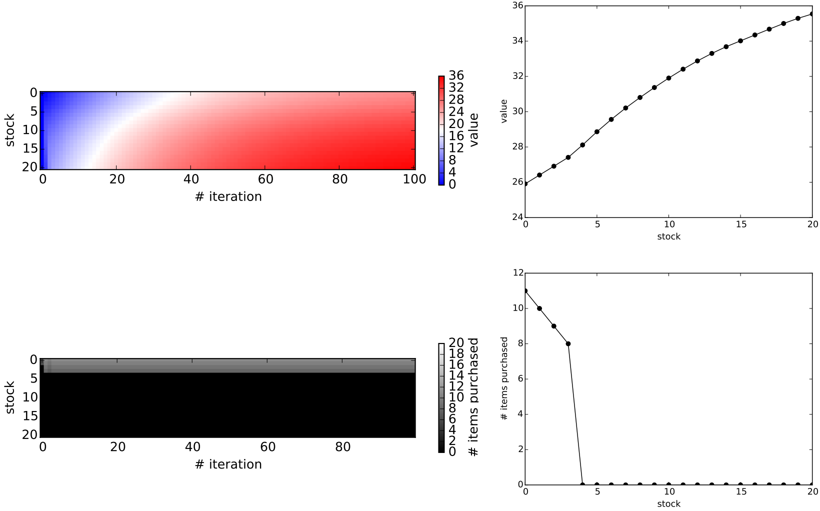
- The space of stationary policies is much smaller than the space of non-stationary policies. If the state and action spaces are finite, then it is finite ($|A|^{|X|}$).
- Solving an infinite-horizon problem essentially amounts to find the optimal value function v_*, i.e. to solve the fixed point equation $v_* = Tv_*$ (then take any policy $\pi \in \mathcal{G}v_*$)
- We already have an algorithm: for any v_0,

$$v_{k+1} \leftarrow Tv_k \qquad \text{(Value Iteration)}$$

  converges asymptotically to the optimal value v_*
- Convergence rate is at least linear:

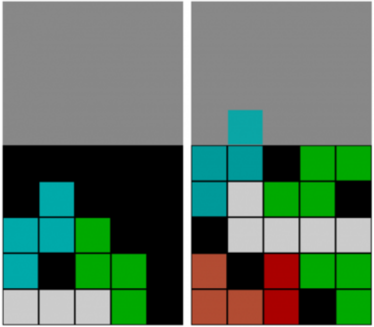
$$\|v_* - v_{k+1}\|_\infty = \|Tv_* - Tv_k\|_\infty \leq \gamma \|v_* - v_k\|_\infty.$$

Example: the Retail Store Management Problem



Mini-Tetris

Assume we play on a small 5×5 board.

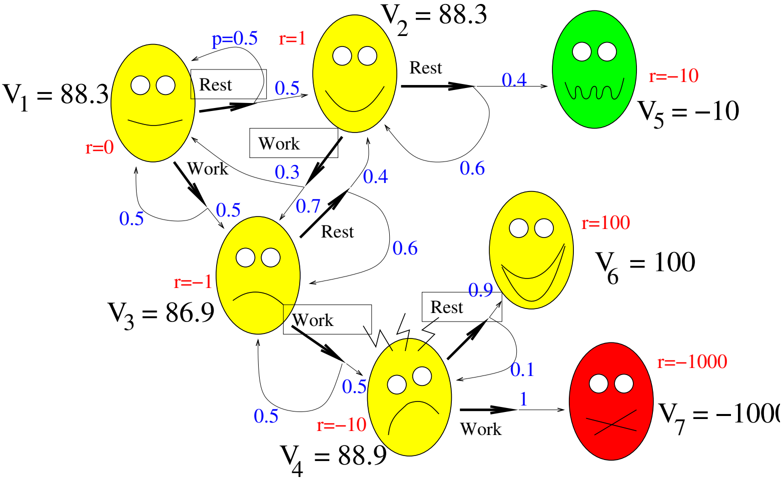


We can enumerate the $2^{25} \simeq 3.10^6$ possible boards and run Value Iteration. The optimal value from the start of the game is $\simeq 13,7$ lines on average per game.

[simulation]

Example: the student dilemma

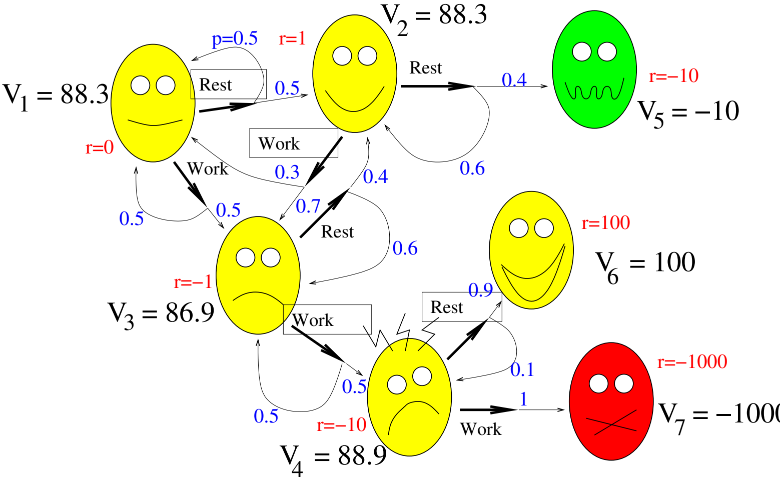
Evaluation of v_π with $\pi = \{\text{rest, work, work, rest}\}$



This can be done by Value Iteration: $v_{k+1} \leftarrow T_\pi v_k \dots$

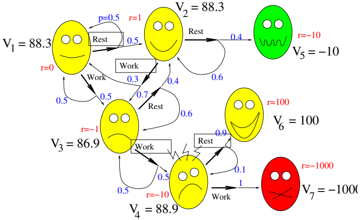
Example: the student dilemma

Evaluation of v_π with $\pi = \{\text{rest, work, work, rest}\}$



This can be done by Value Iteration: $v_{k+1} \leftarrow T_\pi v_k \ldots$

Example: the student dilemma



$$v_\pi = T_\pi v_\pi$$
$$\Updownarrow$$
$$v_\pi(x) = r(x, \pi(x)) + \gamma \sum_y p(y|x, \pi(x)) v_\pi(y)$$

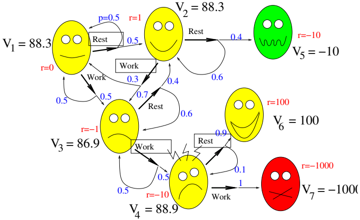
Linear system of equations with unknowns $V_i = v_\pi(x_i)$

$$\begin{cases} V_1 &= 0 + 0.5 V_1 + 0.5 V_2 \\ V_2 &= 1 + 0.3 V_1 + 0.7 V_3 \\ V_3 &= -1 + 0.5 V_4 + 0.5 V_3 \\ V_4 &= -10 + 0.9 V_6 + 0.1 V_4 \\ V_5 &= -10 \\ V_6 &= 100 \\ V_7 &= -1000 \end{cases} \Rightarrow$$

$$(v_\pi, r_\pi \in \mathbb{R}^7,\ P_\pi \in \mathbb{R}^{7\times 7})$$
$$v_\pi = r_\pi + \gamma P_\pi v_\pi$$
$$\Downarrow$$
$$v_\pi = (I - \gamma P_\pi)^{-1} r_\pi$$

$(I - \gamma P_\pi)^{-1} = I + \gamma P_\pi + (\gamma P_\pi)^2 + \ldots$ (always invertible)

Example: the student dilemma



$$v_\pi = T_\pi v_\pi$$
$$\Updownarrow$$
$$v_\pi(x) = r(x,\pi(x)) + \gamma \sum_y p(y|x,\pi(x)) v_\pi(y)$$

Linear system of equations with unknowns $V_i = v_\pi(x_i)$

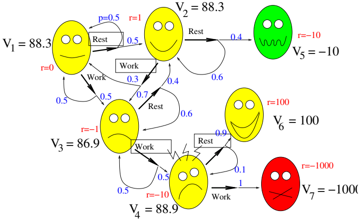
$$\begin{cases} V_1 &= 0 + 0.5V_1 + 0.5V_2 \\ V_2 &= 1 + 0.3V_1 + 0.7V_3 \\ V_3 &= -1 + 0.5V_4 + 0.5V_3 \\ V_4 &= -10 + 0.9V_6 + 0.1V_4 \\ V_5 &= -10 \\ V_6 &= 100 \\ V_7 &= -1000 \end{cases}$$

$$\Rightarrow$$

$$(v_\pi, r_\pi \in \mathbb{R}^7,\ P_\pi \in \mathbb{R}^{7\times 7})$$
$$v_\pi = r_\pi + \gamma P_\pi v_\pi$$
$$\Downarrow$$
$$v_\pi = (I - \gamma P_\pi)^{-1} r_\pi$$

$(I - \gamma P_\pi)^{-1} = I + \gamma P_\pi + (\gamma P_\pi)^2 + \dots$ (always invertible)

Example: the student dilemma



$$v_\pi = T_\pi v_\pi$$

$$\Updownarrow$$

$$v_\pi(x) = r(x, \pi(x)) + \gamma \sum_y p(y|x, \pi(x)) v_\pi(y)$$

Linear system of equations with unknowns $V_i = v_\pi(x_i)$

$$\begin{cases} V_1 &= 0 + 0.5V_1 + 0.5V_2 \\ V_2 &= 1 + 0.3V_1 + 0.7V_3 \\ V_3 &= -1 + 0.5V_4 + 0.5V_3 \\ V_4 &= -10 + 0.9V_6 + 0.1V_4 \\ V_5 &= -10 \\ V_6 &= 100 \\ V_7 &= -1000 \end{cases} \quad \Rightarrow$$

$$(v_\pi, r_\pi \in \mathbb{R}^7,\ P_\pi \in \mathbb{R}^{7\times 7})$$

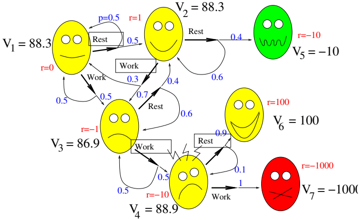
$$v_\pi = r_\pi + \gamma P_\pi v_\pi$$

$$\Downarrow$$

$$v_\pi = (I - \gamma P_\pi)^{-1} r_\pi$$

$$(I - \gamma P_\pi)^{-1} = I + \gamma P_\pi + (\gamma P_\pi)^2 + \ldots \text{ (always invertible)}$$

Example: the student dilemma



$$v_\pi = T_\pi v_\pi$$

$$\Updownarrow$$

$$v_\pi(x) = r(x, \pi(x)) + \gamma \sum_y p(y|x, \pi(x)) v_\pi(y)$$

Linear system of equations with unknowns $V_i = v_\pi(x_i)$

$$\begin{cases} V_1 & = 0 + 0.5 V_1 + 0.5 V_2 \\ V_2 & = 1 + 0.3 V_1 + 0.7 V_3 \\ V_3 & = -1 + 0.5 V_4 + 0.5 V_3 \\ V_4 & = -10 + 0.9 V_6 + 0.1 V_4 \\ V_5 & = -10 \\ V_6 & = 100 \\ V_7 & = -1000 \end{cases} \quad \Rightarrow \quad \begin{array}{c} (v_\pi, r_\pi \in \mathbb{R}^7,\ P_\pi \in \mathbb{R}^{7\times 7}) \\ v_\pi = r_\pi + \gamma P_\pi v_\pi \\ \Downarrow \\ v_\pi = (I - \gamma P_\pi)^{-1} r_\pi \end{array}$$

$(I - \gamma P_\pi)^{-1} = I + \gamma P_\pi + (\gamma P_\pi)^2 + \ldots$ (always invertible)

Policy Iteration

- For any initial stationary policy π_0, for $k = 0, 1, \ldots$
  - Policy evaluation: compute the value v_{π_k} of π_k:

$$v_{\pi_k} = T_\pi v_{\pi_k} \quad \Leftrightarrow \quad v_{\pi_k} = (I - \gamma P_{\pi_k})^{-1} r_{\pi_k}$$

  - Policy improvement: pick π_{k+1} greedy wrt to v_{π_k} ($\pi_{k+1} = \mathcal{G} v_{\pi_k}$):

$$T_{\pi_{k+1}} v_{\pi_k} = T v_{\pi_k} \quad \Leftrightarrow \quad \forall x, \ \ \pi_{k+1}(x) \in \arg\max_a \left\{ r(x, a) + \sum_y \mathbb{P}(y|x, a) v_{\pi_{k+1}}(y) \right\}$$

- Stop when $v_{\pi_{k+1}} = v_{\pi_k}$.

Theorem

Policy Iteration generates a sequence of policies with non-decreasing values ($v_{\pi_{k+1}} \geq v_{\pi_k}$). When the MDP is finite, convergence occurs in a finite number of iterations.

Policy Iteration

- For any initial stationary policy π_0, for $k = 0, 1, \ldots$
  - Policy evaluation: compute the value v_{π_k} of π_k:

$$v_{\pi_k} = T_\pi v_{\pi_k} \quad \Leftrightarrow \quad v_{\pi_k} = (I - \gamma P_{\pi_k})^{-1} r_{\pi_k}$$

  - Policy improvement: pick π_{k+1} greedy wrt to v_{π_k} ($\pi_{k+1} = \mathcal{G} v_{\pi_k}$):

$$T_{\pi_{k+1}} v_{\pi_k} = T v_{\pi_k} \quad \Leftrightarrow \quad \forall x, \;\; \pi_{k+1}(x) \in \arg\max_a \left\{ r(x, a) + \sum_y \mathbb{P}(y|x, a) v_{\pi_{k+1}}(y) \right\}$$

- Stop when $v_{\pi_{k+1}} = v_{\pi_k}$.

Theorem

Policy Iteration generates a sequence of policies with non-decreasing values ($v_{\pi_{k+1}} \geq v_{\pi_k}$). When the MDP is finite, convergence occurs in a finite number of iterations.

Policy Iteration

Proof: (1) Monotonicity:

$$
\begin{aligned}
v_{\pi_{k+1}} - v_{\pi_k} &= (I - \gamma P_{\pi_{k+1}})^{-1} r_{\pi_{k+1}} - v_{\pi_k} \\
&= (I - \gamma P_{\pi_{k+1}})^{-1} \left(r_{\pi_{k+1}} + \gamma P_{\pi_{k+1}} v_{\pi_k} - v_{\pi_k} \right) \\
&= (I - \gamma P_{\pi_{k+1}})^{-1} \left(T_{\pi_{k+1}} v_{\pi_k} - v_{\pi_k} \right) \\
&= \underbrace{(I - \gamma P_{\pi_{k+1}})^{-1}}_{\geq 0} \underbrace{\left(T v_{\pi_k} - T_{\pi_k} v_{\pi_k} \right)}_{\geq 0}
\end{aligned}
$$

where we used $(I - \gamma P_{\pi_{k+1}})^{-1} = I + \gamma P_{\pi_{k+1}} + (\gamma P_{\pi_{k+1}})^2 + \cdots \geq 0$

(2) Optimality: Assume $v_{\pi_{k+1}} = v_{\pi_k}$. Then $v_{\pi_k} = T_{\pi_{k+1}} v_{\pi_{k+1}} = T_{\pi_{k+1}} v_{\pi_k} = T v_{\pi_k}$, and thus $v_{\pi_k} = v_*$ (by the uniqueness of the fixed point of T).

Proof: (1) Monotonicity:

$$
\begin{aligned}
v_{\pi_{k+1}} - v_{\pi_k} &= (I - \gamma P_{\pi_{k+1}})^{-1} r_{\pi_{k+1}} - v_{\pi_k} \\
&= (I - \gamma P_{\pi_{k+1}})^{-1} \left(r_{\pi_{k+1}} + \gamma P_{\pi_{k+1}} v_{\pi_k} - v_{\pi_k} \right) \\
&= (I - \gamma P_{\pi_{k+1}})^{-1} \left(T_{\pi_{k+1}} v_{\pi_k} - v_{\pi_k} \right) \\
&= \underbrace{(I - \gamma P_{\pi_{k+1}})^{-1}}_{\geq 0} \underbrace{(T v_{\pi_k} - T_{\pi_k} v_{\pi_k})}_{\geq 0}
\end{aligned}
$$

where we used $(I - \gamma P_{\pi_{k+1}})^{-1} = I + \gamma P_{\pi_{k+1}} + (\gamma P_{\pi_{k+1}})^2 + \cdots \geq 0$

(2) Optimality: Assume $v_{\pi_{k+1}} = v_{\pi_k}$. Then $v_{\pi_k} = T_{\pi_{k+1}} v_{\pi_{k+1}} = T_{\pi_{k+1}} v_{\pi_k} = T v_{\pi_k}$, and thus $v_{\pi_k} = v_*$ (by the uniqueness of the fixed point of T).

Proof: (1) Monotonicity:

$$\begin{aligned}
v_{\pi_{k+1}} - v_{\pi_k} &= (I - \gamma P_{\pi_{k+1}})^{-1} r_{\pi_{k+1}} - v_{\pi_k} \\
&= (I - \gamma P_{\pi_{k+1}})^{-1} \left(r_{\pi_{k+1}} + \gamma P_{\pi_{k+1}} v_{\pi_k} - v_{\pi_k} \right) \\
&= (I - \gamma P_{\pi_{k+1}})^{-1} \left(T_{\pi_{k+1}} v_{\pi_k} - v_{\pi_k} \right) \\
&= \underbrace{(I - \gamma P_{\pi_{k+1}})^{-1}}_{\geq 0} \underbrace{(T v_{\pi_k} - T_{\pi_k} v_{\pi_k})}_{\geq 0}
\end{aligned}$$

where we used $(I - \gamma P_{\pi_{k+1}})^{-1} = I + \gamma P_{\pi_{k+1}} + (\gamma P_{\pi_{k+1}})^2 + \cdots \geq 0$

(2) Optimality: Assume $v_{\pi_{k+1}} = v_{\pi_k}$. Then $v_{\pi_k} = T_{\pi_{k+1}} v_{\pi_{k+1}} = T_{\pi_{k+1}} v_{\pi_k} = T v_{\pi_k}$, and thus $v_{\pi_k} = v_*$ (by the uniqueness of the fixed point of T).

Proof: (1) Monotonicity:

$$\begin{aligned} v_{\pi_{k+1}} - v_{\pi_k} &= (I - \gamma P_{\pi_{k+1}})^{-1} r_{\pi_{k+1}} - v_{\pi_k} \\ &= (I - \gamma P_{\pi_{k+1}})^{-1} \left(r_{\pi_{k+1}} + \gamma P_{\pi_{k+1}} v_{\pi_k} - v_{\pi_k} \right) \\ &= (I - \gamma P_{\pi_{k+1}})^{-1} \left(T_{\pi_{k+1}} v_{\pi_k} - v_{\pi_k} \right) \\ &= \underbrace{(I - \gamma P_{\pi_{k+1}})^{-1}}_{\geq 0} \underbrace{(T v_{\pi_k} - T_{\pi_k} v_{\pi_k})}_{\geq 0} \end{aligned}$$

where we used $(I - \gamma P_{\pi_{k+1}})^{-1} = I + \gamma P_{\pi_{k+1}} + (\gamma P_{\pi_{k+1}})^2 + \cdots \geq 0$

(2) Optimality: Assume $v_{\pi_{k+1}} = v_{\pi_k}$. Then $v_{\pi_k} = T_{\pi_{k+1}} v_{\pi_{k+1}} = T_{\pi_{k+1}} v_{\pi_k} = T v_{\pi_k}$, and thus $v_{\pi_k} = v_*$ (by the uniqueness of the fixed point of T).

Proof: (1) Monotonicity:

$$
\begin{aligned}
v_{\pi_{k+1}} - v_{\pi_k} &= (I - \gamma P_{\pi_{k+1}})^{-1} r_{\pi_{k+1}} - v_{\pi_k} \\
&= (I - \gamma P_{\pi_{k+1}})^{-1} \left(r_{\pi_{k+1}} + \gamma P_{\pi_{k+1}} v_{\pi_k} - v_{\pi_k} \right) \\
&= (I - \gamma P_{\pi_{k+1}})^{-1} \left(T_{\pi_{k+1}} v_{\pi_k} - v_{\pi_k} \right) \\
&= \underbrace{(I - \gamma P_{\pi_{k+1}})^{-1}}_{\geq 0} \underbrace{(T v_{\pi_k} - T_{\pi_k} v_{\pi_k})}_{\geq 0}
\end{aligned}
$$

where we used $(I - \gamma P_{\pi_{k+1}})^{-1} = I + \gamma P_{\pi_{k+1}} + (\gamma P_{\pi_{k+1}})^2 + \cdots \geq 0$

(2) Optimality: Assume $v_{\pi_{k+1}} = v_{\pi_k}$. Then $v_{\pi_k} = T_{\pi_{k+1}} v_{\pi_{k+1}} = T_{\pi_{k+1}} v_{\pi_k} = T v_{\pi_k}$, and thus $v_{\pi_k} = v_*$ (by the uniqueness of the fixed point of T).

Value Iteration vs Policy Iteration

- Policy Iteration (PI)
  - Convergence in finite time (in practice very fast)$^{(*)}$
  - Each iteration has complexity $O(|X|^2|A|) + O(|X|^3)$ ($\mathcal{G}$ + inv.)
- Value Iteration (VI)
  - Asymptotic convergence (in practice may be long for π to converge)
  - Each iteration has complexity $O(|X|^2|A|)$ (T)

(*) Theorem (Ye, 2010, Hansen 2011, Scherrer 2013)

Policy Iteration converges in at most $O(\frac{|X||A|}{1-\gamma}\log\frac{1}{1-\gamma})$ iterations.

Proof of the complexity of PI

Lemma

For all pairs of policies π and π', $\quad v_{\pi'} - v_\pi = (I - \gamma P_{\pi'})^{-1}(T_{\pi'} v_\pi - v_\pi).$

For some state s_0, (the "worst" state of π_0)

$$
\begin{aligned}
v_*(s_0) - T_{\pi_k} v_*(s_0) &\leq \|v_* - T_{\pi_k} v_*\|_\infty \\
&\leq \|v_* - v_{\pi_k}\|_\infty && \{\text{Lemma}\} \\
&\leq \gamma^k \|v_{\pi_*} - v_{\pi_0}\|_\infty && \{\gamma\text{-contraction}\} \\
&= \gamma^k \|(I - \gamma P_{\pi_0})^{-1}(v_* - T_{\pi_0} v_*)\|_\infty && \{\text{Lemma}\} \\
&\leq \frac{\gamma^k}{1-\gamma} \|v_* - T_{\pi_0} v_*\|_\infty . && \{\|(I - \gamma P_{\pi_0})^{-1}\|_\infty = \frac{1}{1-\gamma}\} \\
&= \frac{\gamma^k}{1-\gamma} (v_*(s_0) - T_{\pi_0} v_*(s_0)).
\end{aligned}
$$

Elimination of a non-optimal action:
For all "sufficiently big" k, $\pi_k(s_0)$ must differ from $\pi_0(s_0)$.
"sufficiently big": $\frac{\gamma^k}{1-\gamma} < 1 \quad \Leftrightarrow \quad k \geq \left\lceil \frac{\log \frac{1}{1-\gamma}}{1-\gamma} \right\rceil > \left\lceil \frac{\log \frac{1}{1-\gamma}}{\log \frac{1}{\gamma}} \right\rceil$.
There are at most $n(m-1)$ non-optimal actions to eliminate.

Proof of the complexity of PI

Lemma

For all pairs of policies π and π', $\quad v_{\pi'} - v_\pi = (I - \gamma P_{\pi'})^{-1}(T_{\pi'} v_\pi - v_\pi).$

For some state s_0, (the "worst" state of π_0)

$$
\begin{aligned}
v_*(s_0) - T_{\pi_k} v_*(s_0) &\leq \|v_* - T_{\pi_k} v_*\|_\infty \\
&\leq \|v_* - v_{\pi_k}\|_\infty && \{\text{Lemma}\} \\
&\leq \gamma^k \|v_{\pi_*} - v_{\pi_0}\|_\infty && \{\gamma\text{-contraction}\} \\
&= \gamma^k \|(I - \gamma P_{\pi_0})^{-1}(v_* - T_{\pi_0} v_*)\|_\infty && \{\text{Lemma}\} \\
&\leq \frac{\gamma^k}{1-\gamma} \|v_* - T_{\pi_0} v_*\|_\infty. && \left\{\|(I - \gamma P_{\pi_0})^{-1}\|_\infty = \frac{1}{1-\gamma}\right\} \\
&= \frac{\gamma^k}{1-\gamma} (v_*(s_0) - T_{\pi_0} v_*(s_0)).
\end{aligned}
$$

Elimination of a non-optimal action:
For all "sufficiently big" k, $\pi_k(s_0)$ must differ from $\pi_0(s_0)$.
"sufficiently big": $\frac{\gamma^k}{1-\gamma} < 1 \quad \Leftrightarrow \quad k \geq \left\lceil \frac{\log \frac{1}{1-\gamma}}{1-\gamma} \right\rceil > \left\lceil \frac{\log \frac{1}{1-\gamma}}{\log \frac{1}{\gamma}} \right\rceil.$
There are at most $n(m-1)$ non-optimal actions to eliminate.

Proof of the complexity of PI

Lemma

For all pairs of policies π and π', $\quad v_{\pi'} - v_\pi = (I - \gamma P_{\pi'})^{-1}(T_{\pi'} v_\pi - v_\pi).$

For some state s_0, (the "worst" state of π_0)

$$
\begin{aligned}
v_*(s_0) - T_{\pi_k} v_*(s_0) &\leq \|v_* - T_{\pi_k} v_*\|_\infty \\
&\leq \|v_* - v_{\pi_k}\|_\infty && \{\text{Lemma}\} \\
&\leq \gamma^k \|v_{\pi_*} - v_{\pi_0}\|_\infty && \{\gamma\text{-contraction}\} \\
&= \gamma^k \|(I - \gamma P_{\pi_0})^{-1}(v_* - T_{\pi_0} v_*)\|_\infty && \{\text{Lemma}\} \\
&\leq \frac{\gamma^k}{1-\gamma} \|v_* - T_{\pi_0} v_*\|_\infty. && \left\{\|(I - \gamma P_{\pi_0})^{-1}\|_\infty = \frac{1}{1-\gamma}\right\} \\
&= \frac{\gamma^k}{1-\gamma}(v_*(s_0) - T_{\pi_0} v_*(s_0)).
\end{aligned}
$$

Elimination of a non-optimal action:
For all "sufficiently big" k, $\pi_k(s_0)$ must differ from $\pi_0(s_0)$.
"sufficiently big": $\frac{\gamma^k}{1-\gamma} < 1 \quad \Leftrightarrow \quad k \geq \left\lceil \frac{\log \frac{1}{1-\gamma}}{1-\gamma} \right\rceil > \left\lceil \frac{\log \frac{1}{1-\gamma}}{\log \frac{1}{\gamma}} \right\rceil$.
There are at most $n(m-1)$ non-optimal actions to eliminate.

Proof of the complexity of PI

Lemma

For all pairs of policies π and π', $\quad v_{\pi'} - v_\pi = (I - \gamma P_{\pi'})^{-1}(T_{\pi'} v_\pi - v_\pi).$

For some state s_0, (the "worst" state of π_0)

$$
\begin{aligned}
v_*(s_0) - T_{\pi_k} v_*(s_0) &\leq \|v_* - T_{\pi_k} v_*\|_\infty \\
&\leq \|v_* - v_{\pi_k}\|_\infty && \{\text{Lemma}\} \\
&\leq \gamma^k \|v_{\pi_*} - v_{\pi_0}\|_\infty && \{\gamma\text{-contraction}\} \\
&= \gamma^k \|(I - \gamma P_{\pi_0})^{-1}(v_* - T_{\pi_0} v_*)\|_\infty && \{\text{Lemma}\} \\
&\leq \frac{\gamma^k}{1-\gamma} \|v_* - T_{\pi_0} v_*\|_\infty . && \left\{\|(I - \gamma P_{\pi_0})^{-1}\|_\infty = \frac{1}{1-\gamma}\right\} \\
&= \frac{\gamma^k}{1-\gamma} (v_*(s_0) - T_{\pi_0} v_*(s_0)).
\end{aligned}
$$

Elimination of a non-optimal action:

For all "sufficiently big" k, $\pi_k(s_0)$ must differ from $\pi_0(s_0)$.

"sufficiently big": $\frac{\gamma^k}{1-\gamma} < 1 \quad \Leftrightarrow \quad k \geq \left\lceil \frac{\log \frac{1}{1-\gamma}}{1-\gamma} \right\rceil > \left\lceil \frac{\log \frac{1}{1-\gamma}}{\log \frac{1}{\gamma}} \right\rceil.$

There are at most $n(m-1)$ non-optimal actions to eliminate.

Proof of the complexity of PI

Lemma

For all pairs of policies π and π', $\quad v_{\pi'} - v_\pi = (I - \gamma P_{\pi'})^{-1}(T_{\pi'} v_\pi - v_\pi).$

For some state s_0, (the "worst" state of π_0)

$$
\begin{aligned}
v_*(s_0) - T_{\pi_k} v_*(s_0) &\leq \|v_* - T_{\pi_k} v_*\|_\infty \\
&\leq \|v_* - v_{\pi_k}\|_\infty && \{\text{Lemma}\} \\
&\leq \gamma^k \|v_{\pi_*} - v_{\pi_0}\|_\infty && \{\gamma\text{-contraction}\} \\
&= \gamma^k \|(I - \gamma P_{\pi_0})^{-1}(v_* - T_{\pi_0} v_*)\|_\infty && \{\text{Lemma}\} \\
&\leq \frac{\gamma^k}{1-\gamma} \|v_* - T_{\pi_0} v_*\|_\infty . && \{\|(I - \gamma P_{\pi_0})^{-1}\|_\infty = \frac{1}{1-\gamma}\} \\
&= \frac{\gamma^k}{1-\gamma} (v_*(s_0) - T_{\pi_0} v_*(s_0)).
\end{aligned}
$$

Elimination of a non-optimal action:

For all "sufficiently big" k, $\pi_k(s_0)$ must differ from $\pi_0(s_0)$.

"sufficiently big": $\frac{\gamma^k}{1-\gamma} < 1 \quad \Leftrightarrow \quad k \geq \left\lceil \frac{\log \frac{1}{1-\gamma}}{1-\gamma} \right\rceil > \left\lceil \frac{\log \frac{1}{1-\gamma}}{\log \frac{1}{\gamma}} \right\rceil$.

There are at most $n(m-1)$ non-optimal actions to eliminate.

Proof of the complexity of PI

Lemma

For all pairs of policies π and π', $\quad v_{\pi'} - v_\pi = (I - \gamma P_{\pi'})^{-1}(T_{\pi'} v_\pi - v_\pi).$

For some state s_0, (the "worst" state of π_0)

$$
\begin{aligned}
v_*(s_0) - T_{\pi_k} v_*(s_0) &\leq \|v_* - T_{\pi_k} v_*\|_\infty \\
&\leq \|v_* - v_{\pi_k}\|_\infty && \{\text{Lemma}\} \\
&\leq \gamma^k \|v_{\pi_*} - v_{\pi_0}\|_\infty && \{\gamma\text{-contraction}\} \\
&= \gamma^k \|(I - \gamma P_{\pi_0})^{-1}(v_* - T_{\pi_0} v_*)\|_\infty && \{\text{Lemma}\} \\
&\leq \frac{\gamma^k}{1-\gamma} \|v_* - T_{\pi_0} v_*\|_\infty. && \{\|(I - \gamma P_{\pi_0})^{-1}\|_\infty = \frac{1}{1-\gamma}\} \\
&= \frac{\gamma^k}{1-\gamma} (v_*(s_0) - T_{\pi_0} v_*(s_0)).
\end{aligned}
$$

Elimination of a non-optimal action:
For all "sufficiently big" k, $\pi_k(s_0)$ must differ from $\pi_0(s_0)$.
"sufficiently big": $\frac{\gamma^k}{1-\gamma} < 1 \quad \Leftrightarrow \quad k \geq \left\lceil \frac{\log \frac{1}{1-\gamma}}{1-\gamma} \right\rceil > \left\lceil \frac{\log \frac{1}{1-\gamma}}{\log \frac{1}{\gamma}} \right\rceil$.
There are at most $n(m-1)$ non-optimal actions to eliminate.

Proof of the complexity of PI

Lemma

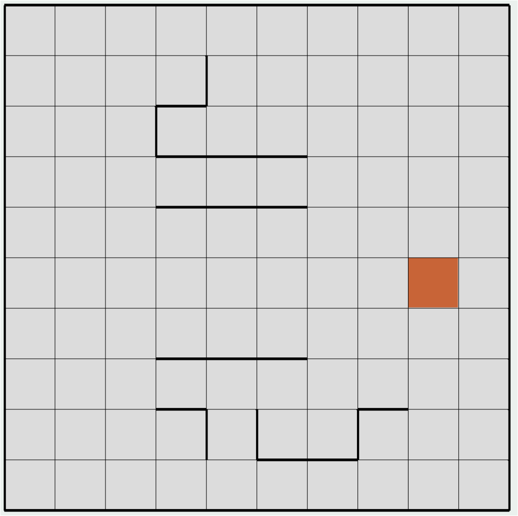
For all pairs of policies π and π', $\quad v_{\pi'} - v_\pi = (I - \gamma P_{\pi'})^{-1}(T_{\pi'} v_\pi - v_\pi)$.

For some state s_0, (the "worst" state of π_0)

$$
\begin{aligned}
v_*(s_0) - T_{\pi_k} v_*(s_0) &\le \|v_* - T_{\pi_k} v_*\|_\infty \\
&\le \|v_* - v_{\pi_k}\|_\infty && \{\text{Lemma}\} \\
&\le \gamma^k \|v_{\pi_*} - v_{\pi_0}\|_\infty && \{\gamma\text{-contraction}\} \\
&= \gamma^k \|(I - \gamma P_{\pi_0})^{-1}(v_* - T_{\pi_0} v_*)\|_\infty && \{\text{Lemma}\} \\
&\le \frac{\gamma^k}{1-\gamma} \|v_* - T_{\pi_0} v_*\|_\infty. && \left\{\|(I - \gamma P_{\pi_0})^{-1}\|_\infty = \frac{1}{1-\gamma}\right\} \\
&= \frac{\gamma^k}{1-\gamma} (v_*(s_0) - T_{\pi_0} v_*(s_0)).
\end{aligned}
$$

Elimination of a non-optimal action:
For all "sufficiently big" k, $\pi_k(s_0)$ must differ from $\pi_0(s_0)$.
"sufficiently big": $\frac{\gamma^k}{1-\gamma} < 1 \quad \Leftrightarrow \quad k \ge \left\lceil \frac{\log \frac{1}{1-\gamma}}{1-\gamma} \right\rceil > \left\lceil \frac{\log \frac{1}{1-\gamma}}{\log \frac{1}{\gamma}} \right\rceil$.
There are at most $n(m-1)$ non-optimal actions to eliminate.

Example: Grid-World



[simulation]

Modified/Optimistic Policy Iteration (I)

Value Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow T v_k = T_{\pi_{k+1}} v_k$$

Policy Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow v_{\pi_{k+1}} = (T_{\pi_{k+1}})^\infty v_k$$

Modified Policy Iteration (Puterman and Shin, 1978)

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow (T_{\pi_{k+1}})^m v_k \qquad m \in \mathbb{N}$$

In practice, moderate values of m allow to find optimal policies faster than VI while being lighter than PI.

λ-Policy Iteration (Ioffe and Bertsekas, 1996)

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow (1-\lambda) \sum_{i=0}^\infty \lambda^i (T_{\pi_{k+1}})^{i+1} v_k \qquad \lambda \in [0,1]$$

Optimistic Policy Iteration (Thiéry and Scherrer, 2009)

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow \sum_{i=0}^\infty \lambda_i (T_{\pi_{k+1}})^{i+1} v_k \qquad \lambda_i \geq 0, \qquad \sum_{i=0}^\infty \lambda_i = 1$$

Modified/Optimistic Policy Iteration (I)

Value Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow T v_k = T_{\pi_{k+1}} v_k$$

Policy Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow v_{\pi_{k+1}} = (T_{\pi_{k+1}})^\infty v_k$$

Modified Policy Iteration (Puterman and Shin, 1978)

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow (T_{\pi_{k+1}})^m v_k \qquad m \in \mathbb{N}$$

In practice, moderate values of m allow to find optimal policies faster than VI while being lighter than PI.

λ-Policy Iteration (Ioffe and Bertsekas, 1996)

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow (1-\lambda) \sum_{i=0}^{\infty} \lambda^i (T_{\pi_{k+1}})^{i+1} v_k \qquad \lambda \in [0,1]$$

Optimistic Policy Iteration (Thiéry and Scherrer, 2009)

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow \sum_{i=0}^{\infty} \lambda_i (T_{\pi_{k+1}})^{i+1} v_k \qquad \lambda_i \geq 0, \qquad \sum_{i=0}^{\infty} \lambda_i = 1$$

Modified/Optimistic Policy Iteration (I)

Value Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow T v_k = T_{\pi_{k+1}} v_k$$

Policy Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow v_{\pi_{k+1}} = (T_{\pi_{k+1}})^\infty v_k$$

Modified Policy Iteration (Puterman and Shin, 1978)

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow (T_{\pi_{k+1}})^m v_k \qquad m \in \mathbb{N}$$

In practice, moderate values of m allow to find optimal policies faster than VI while being lighter than PI.

λ-Policy Iteration (Ioffe and Bertsekas, 1996)

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow (1-\lambda) \sum_{i=0}^{\infty} \lambda^i (T_{\pi_{k+1}})^{i+1} v_k \qquad \lambda \in [0,1]$$

Optimistic Policy Iteration (Thiéry and Scherrer, 2009)

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow \sum_{i=0}^{\infty} \lambda_i (T_{\pi_{k+1}})^{i+1} v_k \qquad \lambda_i \geq 0, \qquad \sum_{i=0}^{\infty} \lambda_i = 1$$

Modified/Optimistic Policy Iteration (I)

Value Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow T v_k = T_{\pi_{k+1}} v_k$$

Policy Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow v_{\pi_{k+1}} = (T_{\pi_{k+1}})^{\infty} v_k$$

Modified Policy Iteration (Puterman and Shin, 1978)

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow (T_{\pi_{k+1}})^m v_k \qquad m \in \mathbb{N}$$

In practice, moderate values of m allow to find optimal policies faster than VI while being lighter than PI.

λ-Policy Iteration (Ioffe and Bertsekas, 1996)

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow (1-\lambda) \sum_{i=0}^{\infty} \lambda^i (T_{\pi_{k+1}})^{i+1} v_k \qquad \lambda \in [0,1]$$

Optimistic Policy Iteration (Thiéry and Scherrer, 2009)

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow \sum_{i=0}^{\infty} \lambda_i (T_{\pi_{k+1}})^{i+1} v_k \qquad \lambda_i \geq 0, \qquad \sum_{i=0}^{\infty} \lambda_i = 1$$

Modified/Optimistic Policy Iteration (I)

Value Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow T v_k = T_{\pi_{k+1}} v_k$$

Policy Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow v_{\pi_{k+1}} = (T_{\pi_{k+1}})^\infty v_k$$

Modified Policy Iteration (Puterman and Shin, 1978)

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow (T_{\pi_{k+1}})^m v_k \qquad m \in \mathbb{N}$$

In practice, moderate values of m allow to find optimal policies faster than VI while being lighter than PI.

λ-Policy Iteration (Ioffe and Bertsekas, 1996)

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow (1-\lambda) \sum_{i=0}^{\infty} \lambda^i (T_{\pi_{k+1}})^{i+1} v_k \qquad \lambda \in [0,1]$$

Optimistic Policy Iteration (Thiéry and Scherrer, 2009)

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow \sum_{i=0}^{\infty} \lambda_i (T_{\pi_{k+1}})^{i+1} v_k \qquad \lambda_i \geq 0, \qquad \sum_{i=0}^{\infty} \lambda_i = 1$$

Modified/Optimistic Policy Iteration (II)

Theorem (Puterman and Shin, 1978)

For any m, Modified Policy Iteration converges asymptotically to an optimal value-policy pair v_*, π_*.

Theorem (Ioffe and Bertsekas, 1996)

For any λ, λ-Policy Iteration converges asymptotically to an optimal value-policy pair v_*, π_*.

Theorem (Thiéry and Scherrer, 2009)

For any set of weights λ_i, Optimistic Policy Iteration converges asymptotically to an optimal value-policy pair v_*, π_*.

Modified/Optimistic Policy Iteration (II)

Theorem (Puterman and Shin, 1978)

For any m, Modified Policy Iteration converges asymptotically to an optimal value-policy pair v_*, π_*.

Theorem (Ioffe and Bertsekas, 1996)

For any λ, λ-Policy Iteration converges asymptotically to an optimal value-policy pair v_*, π_*.

Theorem (Thiéry and Scherrer, 2009)

For any set of weights λ_i, Optimistic Policy Iteration converges asymptotically to an optimal value-policy pair v_*, π_*.

Modified/Optimistic Policy Iteration (II)

Theorem (Puterman and Shin, 1978)

For any m, Modified Policy Iteration converges asymptotically to an optimal value-policy pair v_*, π_*.

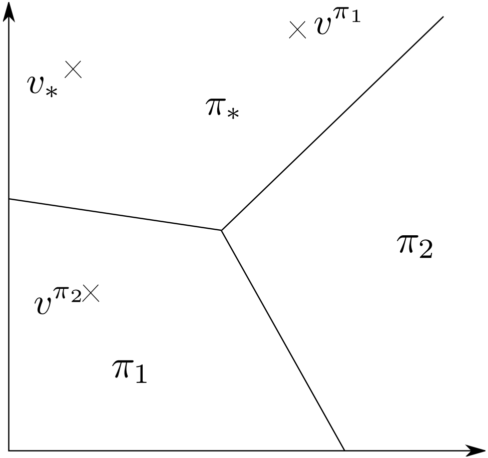
Theorem (Ioffe and Bertsekas, 1996)

For any λ, λ-Policy Iteration converges asymptotically to an optimal value-policy pair v_*, π_*.

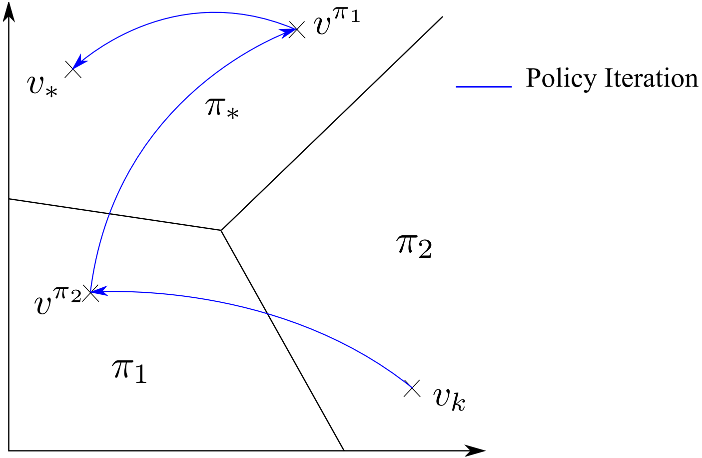
Theorem (Thiéry and Scherrer, 2009)

For any set of weights λ_i, Optimistic Policy Iteration converges asymptotically to an optimal value-policy pair v_*, π_*.

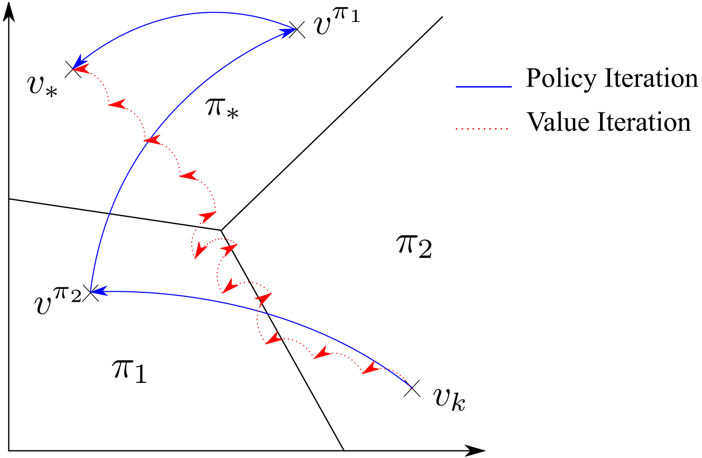
Optimism in the greedy partition



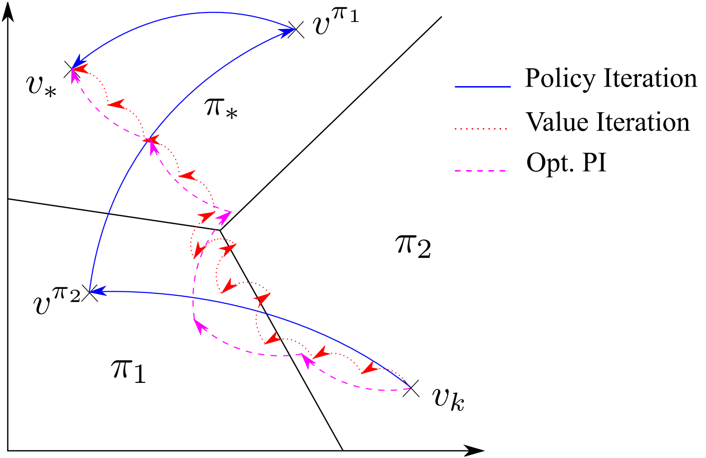
Optimism in the greedy partition



Optimism in the greedy partition



Optimism in the greedy partition



The "q-value" variation (I)

- The **q-value** of policy π at (x, a) is the value if one first takes action a and then follows policy π:

$$q_\pi(x,a) = E\left[\sum_{t=0}^{\infty} \gamma^t r(x_t,a_t) \,\middle|\, x_0 = x, a_0 = a, \{\forall t \geq 1,\ a_t = \pi(x_t)\}\right]$$

- q_π and q_* satisfy the following Bellman equations

$$\forall x,\ q_\pi(x,a) = r(x,a) + \gamma \sum_y p(y|x,a) q_\pi(y,\pi(y)) \quad \Leftrightarrow \quad q_\pi = T_\pi q_\pi$$

$$\forall x,\ q_*(x,a) = r(x,a) + \gamma \sum_y p(y|x,a) \max_{a'} q_*(y,a') \quad \Leftrightarrow \quad q_* = T q_*$$

$$\forall x,\ \pi(x) \in \arg\max_a q(x,a) \quad \Leftrightarrow \quad \pi = \mathcal{G} q$$

- The following relations hold:

$$v_\pi(x) = q_\pi(x,\pi(x)), \qquad q_\pi(x,a) = r(x,a) + \gamma \sum_y p(y|x,a) v_\pi(y)$$

$$v_*(x) = \max_a q_*(x,a), \qquad q_*(x,a) = r(x,a) + \gamma \sum_y p(y|x,a) v_*(y)$$

The "q-value" variation (I)

- The **q-value** of policy π at (x, a) is the value if one <u>first</u> takes action a and <u>then</u> follows policy π:

$$q_\pi(x,a) = E\left[\sum_{t=0}^{\infty} \gamma^t r(x_t, a_t) \middle| x_0 = x, a_0 = a, \{\forall t \geq 1,\ a_t = \pi(x_t)\} \right]$$

- q_π and q_* satisfy the following Bellman equations

$$\forall x,\ q_\pi(x,a) = r(x,a) + \gamma \sum_y p(y|x,a) q_\pi(y, \pi(y)) \quad \Leftrightarrow \quad q_\pi = T_\pi q_\pi$$

$$\forall x,\ q_*(x,a) = r(x,a) + \gamma \sum_y p(y|x,a) \max_{a'} q_*(y, a') \quad \Leftrightarrow \quad q_* = T q_*$$

$$\forall x,\ \pi(x) \in \arg\max_a q(x,a) \quad \Leftrightarrow \quad \pi = \mathcal{G} q$$

- The following relations hold:

$$v_\pi(x) = q_\pi(x, \pi(x)), \qquad q_\pi(x,a) = r(x,a) + \gamma \sum_y p(y|x,a) v_\pi(y)$$

$$v_*(x) = \max_a q_*(x,a), \qquad q_*(x,a) = r(x,a) + \gamma \sum_y p(y|x,a) v_*(y)$$

The "q-value" variation (I)

- The **q-value** of policy π at (x, a) is the value if one <u>first</u> takes action a and <u>then</u> follows policy π:

$$q_\pi(x,a) = E\left[\sum_{t=0}^{\infty} \gamma^t r(x_t, a_t) \middle| x_0 = x, a_0 = a, \{\forall t \geq 1,\ a_t = \pi(x_t)\}\right]$$

- q_π and q_* satisfy the following Bellman equations

$$\forall x,\ q_\pi(x,a) = r(x,a) + \gamma \sum_y p(y|x,a) q_\pi(y, \pi(y)) \quad \Leftrightarrow \quad q_\pi = T_\pi q_\pi$$

$$\forall x,\ q_*(x,a) = r(x,a) + \gamma \sum_y p(y|x,a) \max_{a'} q_*(y, a') \quad \Leftrightarrow \quad q_* = T q_*$$

$$\forall x,\ \pi(x) \in \arg\max_a q(x,a) \quad \Leftrightarrow \quad \pi = \mathcal{G} q$$

- The following relations hold:

$$v_\pi(x) = q_\pi(x, \pi(x)), \qquad q_\pi(x,a) = r(x,a) + \gamma \sum_y p(y|x,a) v_\pi(y)$$

$$v_*(x) = \max_a q_*(x, a), \qquad q_*(x,a) = r(x,a) + \gamma \sum_y p(y|x,a) v_*(y)$$

The “q-value” variation (II)

- “q-values” are values in an “augmented problem” where states are $X \times A$:

$$\boxed{(x_t, a_t)} \xrightarrow{\text{uncontrolled/stochastic}} (x_{t+1}) \xrightarrow{\text{controlled/deterministic}} \boxed{(x_{t+1}, a_{t+1})}$$

- VI, PI and MPI with $q-values$ are mathematically equivalent to their v-counterparts
- Requires more memory ($O(|X||A|)$ instead of $O(|X|)$)
- The computation of $\mathcal{G}q$ is lighter ($O(|A|)$ instead of $O(|X|^2|A|)$) and model-free:

$$\forall x,\ \pi(x) \in \arg\max_a q(x,a) \quad \Leftrightarrow \quad \pi = \mathcal{G}q$$

$$\forall x,\ \pi_*(x) \in \arg\max_a q_*(x,a)$$

The “q-value” variation (II)

- “q-values” are values in an “augmented problem” where states are $X \times A$:

$$\boxed{(x_t, a_t)} \xrightarrow{\text{uncontrolled/stochastic}} (x_{t+1}) \xrightarrow{\text{controlled/deterministic}} \boxed{(x_{t+1}, a_{t+1})}$$

- VI, PI and MPI with $q-values$ are mathematically equivalent to their v-counterparts
- Requires more memory ($O(|X||A|)$ instead of $O(|X|)$)
- The computation of $\mathcal{G}q$ is lighter ($O(|A|)$ instead of $O(|X|^2|A|)$) and model-free:

$$\forall x,\ \pi(x) \in \arg\max_a q(x, a) \quad \Leftrightarrow \quad \pi = \mathcal{G}q$$

$$\forall x,\ \pi_*(x) \in \arg\max_a q_*(x, a)$$

Asynchronous algorithms

- Motivations:
  - Faster convergence
  - Parallel and distributed computations
  - Simulation-based implementations
- General framework: Partition X into disjoint non-empty subsets $X_1, \dots, X_n$, and use separate processor ℓ for updating $v(x)$ for $x \in X_\ell$. Let v be partitioned as $v = (v_1, \dots, v_n)$ where v_ℓ is the restriction of v on X_ℓ.
- (Synchronous) VI does:

$$v_\ell^{t+1}(x) = T(v_1^t, \dots, v_n^t)(x), \quad x \in X_\ell, \quad \ell = 1, \dots, n$$

- Asynchronous VI does, for some subsets of times $\mathcal{W}_\ell$:

$$v_\ell^{t+1}(x) = \begin{cases} T(v_1^{\tau_{\ell_1}(t)}, \dots, v_n^{\tau_{\ell_1}(t)})(x) & \text{if } t \in \mathcal{W}_\ell \\ v_\ell^t(x) & \text{if } t \notin \mathcal{W}_\ell \end{cases}$$

where $t - \tau_{\ell_j}(t)$ are communications "delays".

Asynchronous algorithms

- Motivations:
  - Faster convergence
  - Parallel and distributed computations
  - Simulation-based implementations
- General framework: Partition X into disjoint non-empty subsets $X_1, \dots, X_n$, and use separate processor ℓ for updating $v(x)$ for $x \in X_\ell$. Let v be partitioned as $v = (v_1, \dots, v_n)$ where v_ℓ is the restriction of v on X_ℓ.
- (Synchronous) VI does:

$$v_\ell^{t+1}(x) = T(v_1^t, \dots, v_n^t)(x), \quad x \in X_\ell, \quad \ell = 1, \dots, n$$

- Asynchronous VI does, for some subsets of times $\mathcal{W}_\ell$:

$$v_\ell^{t+1}(x) = \begin{cases} T(v_1^{\tau_{\ell_1}(t)}, \dots, v_n^{\tau_{\ell_1}(t)})(x) & \text{if } t \in \mathcal{W}_\ell \\ v_\ell^t(x) & \text{if } t \notin \mathcal{W}_\ell \end{cases}$$

where $t - \tau_{\ell_j}(t)$ are communications "delays".

One-state-at-a-time iterations

- An important special case: Assume n states, a separate processor for each state, and no delays
- Generate a sequence or states $\{x_0, x_1, \ldots\}$
- Asynchronous VI:

$$v_{t+1}(x) = \begin{cases} T v_t(x) & \text{if } x = x_t \\ v_t(x) & \text{if } x \neq x_t \end{cases}$$

- $\{x_0, x_1, \ldots\}$ may be generated by simulations

[simulation]

- The special case where

$$\{x_0, x_1, \ldots\} = \{1, \ldots, n, 1, \ldots, n, 1, \ldots\}$$

is the Gauss-Seidel method.

One-state-at-a-time iterations

- An important special case: Assume n states, a separate processor for each state, and no delays
- Generate a sequence or states $\{x_0, x_1, \ldots\}$
- Asynchronous VI:

$$v_{t+1}(x) = \begin{cases} T v_t(x) & \text{if } x = x_t \\ v_t(x) & \text{if } x \neq x_t \end{cases}$$

- $\{x_0, x_1, \ldots\}$ may be generated by simulations

  [simulation]

- The special case where

$$\{x_0, x_1, \ldots\} = \{1, \ldots, n, 1, \ldots, n, 1, \ldots\}$$

is the Gauss-Seidel method.

One-state-at-a-time iterations

- An important special case: Assume n states, a separate processor for each state, and no delays
- Generate a sequence or states $\{x_0, x_1, \ldots\}$
- Asynchronous VI:

$$v_{t+1}(x) = \begin{cases} T v_t(x) & \text{if } x = x_t \\ v_t(x) & \text{if } x \neq x_t \end{cases}$$

- $\{x_0, x_1, \ldots\}$ may be generated by simulations

  [simulation]

- The special case where

$$\{x_0, x_1, \ldots\} = \{1, \ldots, n, 1, \ldots, n, 1, \ldots\}$$

is the Gauss-Seidel method.

Asynchronous Convergence Therorem

$$f_\ell^{t+1}(x) = \begin{cases} F(f_1^{\tau_{\ell_1}(t)}, \dots, f_n^{\tau_{\ell_1}(t)})(x) & \text{if } t \in \mathcal{W}_\ell \\ f_\ell^t(x) & \text{if } t \notin \mathcal{W}_\ell \end{cases}$$

Theorem

Assume that for all ℓ, $j = 1, \dots, n$, $\mathcal{W}_\ell$ is infinite and $\lim_{t\to\infty} \tau_{\ell_j}(t) = \infty$. Assume that F is a contraction-mapping for the max-norm. Then for any $f^0 = (f_1^0, \dots, f_n^0)$, the sequence f^t converges pointwise to the unique fixed point f_* of F.

- Asynchronous VI converges to v_*
- (A modification of) PI also works asynchronously

Asynchronous Convergence Therorem

$$f_\ell^{t+1}(x) = \begin{cases} F(f_1^{\tau_{\ell_1}(t)}, \dots, f_n^{\tau_{\ell_1}(t)})(x) & \text{if } t \in \mathcal{W}_\ell \\ f_\ell^t(x) & \text{if } t \notin \mathcal{W}_\ell \end{cases}$$

Theorem

Assume that for all $\ell, j = 1, \dots, n$, $\mathcal{W}_\ell$ is infinite and $\lim_{t\to\infty} \tau_{\ell_j}(t) = \infty$. Assume that F is a contraction-mapping for the max-norm. Then for any $f^0 = (f_1^0, \dots, f_n^0)$, the sequence f^t converges pointwise to the unique fixed point f_* of F.

- Asynchronous VI converges to v_*
- (A modification of) PI also works asynchronously

Real-Time Dynamic Programming

- What can we do when the model is unknown ?
- Learn a model from experience (reinforcement learning)

$$\hat{r}(x, a) = \frac{\sum_t \mathbb{1}_{(x_t, a_t) = x, a} r_t}{\sum_t \mathbb{1}_{(x_t, a_t) = (x, a)}}$$

$$\hat{T}(x, a, x') = \frac{\sum_t \mathbb{1}_{(x_t, a_t, x_{t+1}) = (x, a, x')}}{\sum_t \mathbb{1}_{(x_t, a_t) = (x, a)}}$$

- Run any DP algorithm, e.g. one-state-at-a-time VI
- Convergence if all state-actions keeps being updated (law of large numbers + Asynchronous convergence theorem)

Real-Time Dynamic Programming

- What can we do when the model is unknown ?
- Learn a model from experience (reinforcement learning)

$$\hat{r}(x,a) = \frac{\sum_t \mathbb{1}_{(x_t,a_t)=x,a} r_t}{\sum_t \mathbb{1}_{(x_t,a_t)=(x,a)}}$$

$$\hat{T}(x,a,x') = \frac{\sum_t \mathbb{1}_{(x_t,a_t,x_{t+1})=(x,a,x')}}{\sum_t \mathbb{1}_{(x_t,a_t)=(x,a)}}$$

- Run any DP algorithm, e.g. one-state-at-a-time VI
- Convergence if all state-actions keeps being updated (law of large numbers + Asynchronous convergence theorem)

Real-Time Dynamic Programming

- What can we do when the model is unknown ?
- Learn a model from experience (reinforcement learning)

$$\hat{r}(x, a) = \frac{\sum_t \mathbb{1}_{(x_t, a_t) = x, a} r_t}{\sum_t \mathbb{1}_{(x_t, a_t) = (x, a)}}$$

$$\hat{T}(x, a, x') = \frac{\sum_t \mathbb{1}_{(x_t, a_t, x_{t+1}) = (x, a, x')}}{\sum_t \mathbb{1}_{(x_t, a_t) = (x, a)}}$$

- Run any DP algorithm, e.g. one-state-at-a-time VI
- Convergence if all state-actions keeps being updated (law of large numbers + Asynchronous convergence theorem)

Q-Learning

Initialize $q(\cdot,\cdot)$ arbitrarily. For all $k = 1, 2, \dots,$

- Sampling: Select a state-action pair (x_k, a_k) and simulate a transition: $r_k = r(x, a, w_k)$ and $x'_k = f(x, a, w_k)$
- Update:

$$q(x_k, a_k) = (1-\alpha_k) q(x_k, a_k) + \alpha_k \Big(\underbrace{r_k + \gamma \max_{a'} q(x'_k, a')}_{\text{unbiased estimate of } (Tq)(x_k, a_k)} \Big)$$

- If $a_k = \pi(a_k)$ this is known as TD-Learning.

[simulation]

- This is a stochastic approximation "model-free" algorithm

Theorem

If the stepsizes satisfy $\sum_k \alpha_k = \infty$ and $\sum_k {\alpha_k}^2 < \infty$, and if state-action pairs are selected infinitely often, then Q-Learning converges a.s. to q_*.

Q-Learning

Initialize $q(\cdot,\cdot)$ arbitrarily. For all $k = 1, 2, \ldots,$

- Sampling: Select a state-action pair (x_k, a_k) and simulate a transition: $r_k = r(x, a, w_k)$ and $x'_k = f(x, a, w_k)$
- Update:

$$q(x_k, a_k) = (1-\alpha_k)q(x_k, a_k) + \alpha_k\Big(\underbrace{r_k + \gamma \max_{a'} q(x'_k, a')}_{\text{unbiased estimate of } (Tq)(x_k, a_k)} \Big)$$

- If $a_k = \pi(a_k)$ this is known as TD-Learning.

[simulation]

- This is a stochastic approximation "model-free" algorithm

Theorem

If the stepsizes satisfy $\sum_k \alpha_k = \infty$ and $\sum_k {\alpha_k}^2 < \infty$, and if state-action pairs are selected infinitely often, then Q-Learning converges a.s. to q_*.

Q-Learning

Initialize $q(\cdot,\cdot)$ arbitrarily. For all $k = 1, 2, \ldots,$

- Sampling: Select a state-action pair (x_k, a_k) and simulate a transition: $r_k = r(x, a, w_k)$ and $x'_k = f(x, a, w_k)$
- Update:

$$q(x_k, a_k) = (1-\alpha_k)q(x_k, a_k) + \alpha_k\Big(\underbrace{r_k + \gamma \max_{a'} q(x'_k, a')}_{\text{unbiased estimate of } (Tq)(x_k, a_k)}\Big)$$

- If $a_k = \pi(a_k)$ this is known as TD-Learning.

[simulation]

- This is a stochastic approximation "model-free" algorithm

Theorem

If the stepsizes satisfy $\sum_k \alpha_k = \infty$ and $\sum_k {\alpha_k}^2 < \infty$, and if state-action pairs are selected infinitely often, then Q-Learning converges a.s. to q_*.

Q-Learning

Initialize $q(\cdot,\cdot)$ arbitrarily. For all $k = 1, 2, \ldots,$

- Sampling: Select a state-action pair (x_k, a_k) and simulate a transition: $r_k = r(x, a, w_k)$ and $x'_k = f(x, a, w_k)$
- Update:

$$q(x_k, a_k) = (1-\alpha_k)q(x_k, a_k) + \alpha_k\Big(\underbrace{r_k + \gamma \max_{a'} q(x'_k, a')}_{\text{unbiased estimate of } (Tq)(x_k, a_k)} \Big)$$

- If $a_k = \pi(a_k)$ this is known as TD-Learning.

[simulation]

- This is a stochastic approximation "model-free" algorithm

Theorem

If the stepsizes satisfy $\sum_k \alpha_k = \infty$ and $\sum_k {\alpha_k}^2 < \infty$, and if state-action pairs are selected infinitely often, then Q-Learning converges a.s. to q_*.

About the proof for Q-Learning

- The proof is sophisticated, based on stochastic approximation theory as well as asynchronous algorithms with contraction mappings...
- Extends the standard Robbins-Monro algorithm for solving $f(x) = \mathbb{E}_w[g(x,w)] = C$ with $f' \geq 0$.

$$x_{k+1} = x_k + \alpha_k\left(\quad C \quad - \quad \underbrace{g(x_k, w_k)}_{\text{unbiased estimate of } f(x_n)} \quad \right)$$

- Q-Learning does not work with a "v-value" because the maximization and the expectation are in reverse order

$$(Tq)(x,a) = \mathbb{E}_w\left[r(x,a,w) + \gamma \sum_y p(y|x,a) \max_{a'} q_*(y,a')\right]$$

$$(Tv)(x,a) = \max_a \mathbb{E}_w\left[r(x,a,w) + \gamma \sum_y p(y|x,a) v(y)\right]$$

- TD-Learning works with a "v-value" (because there is no max)

About the proof for Q-Learning

- The proof is sophisticated, based on stochastic approximation theory as well as asynchronous algorithms with contraction mappings...
- Extends the standard Robbins-Monro algorithm for solving $f(x) = \mathbb{E}_w[g(x, w)] = C$ with $f' \geq 0$.

$$x_{k+1} = x_k + \alpha_k\Big(\quad C \quad - \quad \underbrace{g(x_k, w_k)}_{\text{unbiased estimate of } f(x_n)} \quad \Big)$$

- Q-Learning does not work with a "v-value" because the maximization and the expectation are in reverse order

$$(Tq)(x, a) = \mathbb{E}_w\Big[r(x, a, w) + \gamma \sum_y p(y|x, a)\max_{a'} q_*(y, a')\Big]$$

$$(Tv)(x, a) = \max_a \mathbb{E}_w\Big[r(x, a, w) + \gamma \sum_y p(y|x, a)v(y)\Big]$$

- TD-Learning works with a "v-value" (because there is no max)

Exploration-Exploitation Dilemma

- When running on-line RTDP and Q-Learning there is an exploration policy (that needs to try state-action pairs infinitely often)
- Convergence requires a minimum of exploration
- As convergence happens, one would like to exploit the most what we know about the optimal policy.
- The 1-state MDP special case of this exploration-exploitation dilemma is the bandit problem
- An extension of UCB exist for general MDPs: UCRL (RTDP model + exact solution to the model)

Exploration-Exploitation Dilemma

- When running on-line RTDP and Q-Learning there is an exploration policy (that needs to try state-action pairs infinitely often)
- Convergence requires a minimum of exploration
- As convergence happens, one would like to exploit the most what we know about the optimal policy.
- The 1-state MDP special case of this exploration-exploitation dilemma is the bandit problem
- An extension of UCB exist for general MDPs: UCRL (RTDP model + exact solution to the model)

SARSA

- On-policy vs Off-policy: State-Action-Reward-State-Action does a TD-update while the policy evolves:

$$q_{t+1}(x_t, a_t) = (1 - \alpha_t) q_t(x_t, a_t) + \alpha_t \big(r_t + \gamma q_t(x_{k+1}, a_{k+1}) \big)$$

$$\mathbb{P}(a_{t+1} = a | x_{t+1} = x, q_t) = \frac{e^{\beta_t} q_t(x, a)}{\sum_{a'} e^{\beta_t} q_t(x, a')}$$

where β_t is a temperature parameter (when β_t tends to infinity, a_t is greedy wrt to q)

Theorem (Singh, Jaakkola, Littman, Szepesvari, 98)

If $\beta_t(x) = \log \frac{\sum_t \mathbb{1}_{x_t = x}}{\max_{a'} |\max_a'' q_t(x,a'') - q_t(x,a')|}$, $\sum_t \alpha_t = \infty$ and $\sum_t {\alpha_t}^2 < \infty$, then SARSA converges a.s. to the optimal value-policy pair.

SARSA

- On-policy vs Off-policy: State-Action-Reward-State-Action does a TD-update while the policy evolves:

$$q_{t+1}(x_t, a_t) = (1-\alpha_t) q_t(x_t, a_t) + \alpha_t \big(r_t + \gamma q_t(x_{k+1}, a_{k+1}) \big)$$

$$\mathbb{P}(a_{t+1} = a | x_{t+1} = x, q_t) = \frac{e^{\beta_t} q_t(x, a)}{\sum_{a'} e^{\beta_t} q_t(x, a')}$$

where β_t is a temperature parameter (when β_t tends to infinity, a_t is greedy wrt to q)

Theorem (Singh, Jaakkola, Littman, Szepesvari, 98)

If $\beta_t(x) = \log \frac{\sum_t \mathbb{1}_{x_t = x}}{\max_{a'} |\max_a'' q_t(x,a'') - q_t(x,a')|}$, $\sum_t \alpha_t = \infty$ and $\sum_t {\alpha_t}^2 < \infty$, then SARSA converges a.s. to the optimal value-policy pair.

Outline for Part 1

- Finite-Horizon Optimal Control
  - Problem definition
  - Policy evaluation: Value Iteration[1]
  - Policy optimization: Value Iteration[2]

- Stationary Infinite-Horizon Optimal Control
  - Bellman operators
  - Contraction Mappings
  - Stationary policies
  - Policy evaluation
  - Policy optimization: Value Iteration[3], Policy Iteration, Modified/Optimistic Policy Iteration
  - Asynchronous Algorithms
  - Learning from samples: Real-Time Dynamic Programming, Q-Learning, TD-Learning, SARSA

Brief Outline

- Part 1: “Small” problems
  - Optimal control problem definitions
  - Dynamic Programming (DP) principles, standard algorithms
  - Learning (solving from samples)
- Part 2: “Large” problems
  - Approximate DP Algorithms
  - Theoretical guarantees

Outline for Part 2

- Approximate Dynamic Programming
  - Approximate VI: Fitted-Q Iteration
  - Approximate MPI: AMPI-Q, CBMPI
  - Approximate PI: LSPI
    - Projected value estimation: LSTD,LSBR

- Advanced topics
  - Non-stationary policies for stationary MDPs: NSVI, NSPI, NSMPI
  - Max-norm vs L_p-norm, concentrability coefficients: CPI, API(α), PSDP$_\infty$

Algorithms

Value Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow T v_k = T_{\pi_{k+1}} v_k$$

Policy Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow v_{\pi_{k+1}} = (T_{\pi_{k+1}})^\infty v_k$$

Modified Policy Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow (T_{\pi_{k+1}})^m v_k \qquad m \in \mathbb{N}$$

When the problem is big (ex: Tetris, $\simeq 2^{10\times 20} \simeq 10^{60}$ states!), even applying once $T_{\pi_{k+1}}$ of storing the value function is infeasible. ☹

Algorithms

Value Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow T v_k = T_{\pi_{k+1}} v_k$$

Policy Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow v_{\pi_{k+1}} = (T_{\pi_{k+1}})^{\infty} v_k$$

Modified Policy Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow (T_{\pi_{k+1}})^{m} v_k \qquad m \in \mathbb{N}$$

When the problem is big (ex: Tetris, $\simeq 2^{10\times 20} \simeq 10^{60}$ states!), even applying once $T_{\pi_{k+1}}$ of storing the value function is infeasible. ☹

Approximate VI: Fitted Q-Iteration

(q_k) are represented in $\mathcal{F} \subseteq \mathbb{R}^{X \times A}$

- $\pi_{k+1} \leftarrow \mathcal{G} q_k$
- $q_{k+1} \leftarrow T_{\pi_{k+1}} q_k$

Policy update

In state x, the greedy action is estimated by:

$$\pi_{k+1}(x) = \arg\max_{a \in A} q_k(x, a)$$

Value function update

1. **Point-wise estimation through samples:**
   For N state-action pairs $(x^{(i)}, a^{(i)}) \sim \mu$, simulate a transition $(r^{(i)}, x'^{(i)})$ and compute an unbiased estimate of $\left[T_{\pi_{k+1}} q_k\right](x^{(i)}, a^{(i)})$

$$\widehat{q}_{k+1}(x^{(i)}, a^{(i)}) = r_t^{(i)} + \gamma q_k(x'^{(i)}, \pi_{k+1}(x'^{(i)}))$$

2. **Generalisation through regression:**
   q_{k+1} is computed as the best fit of these estimates in $\mathcal{F}$

$$q_{k+1} = \arg\min_{q \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} \left(q(x^{(i)}, a^{(i)}) - \widehat{q}_{k+1}(x^{(i)}, a^{(i)})\right)^2$$

Approximate VI: Fitted Q-Iteration

$\boxed{(q_k) \text{ are represented in } \mathcal{F} \subseteq \mathbb{R}^{X \times A}}$

- $\pi_{k+1} \leftarrow \mathcal{G} q_k$
- $q_{k+1} \leftarrow T_{\pi_{k+1}} q_k$

Policy update

In state x, the greedy action is estimated by:

$$\pi_{k+1}(x) = \arg\max_{a \in A} q_k(x, a)$$

Value function update

1. **Point-wise estimation** through samples**:**
For N state-action pairs $(x^{(i)}, a^{(i)}) \sim \mu$, simulate a transition $(r^{(i)}, x'^{(i)})$ and compute an unbiased estimate of $\left[T_{\pi_{k+1}} q_k\right](x^{(i)}, a^{(i)})$

$$\widehat{q}_{k+1}(x^{(i)}, a^{(i)}) = r_t^{(i)} + \gamma q_k(x'^{(i)}, \pi_{k+1}(x'^{(i)}))$$

2. **Generalisation** through regression**:**
q_{k+1} is computed as the best fit of these estimates in $\mathcal{F}$

$$q_{k+1} = \arg\min_{q \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} \left(q(x^{(i)}, a^{(i)}) - \widehat{q}_{k+1}(x^{(i)}, a^{(i)}) \right)^2$$

Approximate Value Iteration

Fitted Q-Iteration is an instance of Approximate VI:

$$q_{k+1} = T q_k + \epsilon_{k+1}$$

where (regression literature):

$$\|\epsilon_{k+1}\|_{2,\mu} = \|q_{k+1} - T q_k\|_{2,\mu} \leq O\left(\underbrace{\sup_{g\in\mathcal{F}} \inf_{f\in\mathcal{F}} \|f - Tg\|_{2,\mu}}_{Approx.error} + \underbrace{\frac{1}{\sqrt{n}}}_{Estim.error}\right)$$

Theorem

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. The loss due to running policy π_k instead of the optimal policy π_* satisfies

$$\limsup_{k\to\infty} \|q_* - q_{\pi_k}\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2}\epsilon.$$

Approximate Value Iteration

Fitted Q-Iteration is an instance of Approximate VI:

$$q_{k+1} = T q_k + \epsilon_{k+1}$$

where (regression literature):

$$\|\epsilon_{k+1}\|_{2,\mu} = \|q_{k+1} - T q_k\|_{2,\mu} \leq O\left(\underbrace{\sup_{g \in \mathcal{F}} \inf_{f \in \mathcal{F}} \|f - Tg\|_{2,\mu}}_{Approx.error} + \underbrace{\frac{1}{\sqrt{n}}}_{Estim.error} \right)$$

Theorem

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. The loss due to running policy π_k instead of the optimal policy π_* satisfies

$$\limsup_{k \to \infty} \|q_* - q_{\pi_k}\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2} \epsilon.$$

Error propagation for AVI

1. Bounding: $\|q_* - q_k\|_\infty$:

$$\begin{aligned}\|q_* - q_k\|_\infty &= \|q_* - Tq_{k-1} - \epsilon_k\|_\infty \\ &\leq \|Tq_* - Tq_{k-1}\|_\infty + \epsilon \\ &\leq \gamma\|q_* - q_{k-1}\|_\infty + \epsilon \\ &\leq \frac{\epsilon}{1-\gamma}.\end{aligned}$$

2. From $\|q_* - q_k\|_\infty$ to $\|q_* - q_{\pi_{k+1}}\|_\infty$ ($\pi_{k+1} = \mathcal{G} q_k$):

$$\begin{aligned}\|q_* - q_{\pi_{k+1}}\|_\infty &\leq \|Tq_* - T_{\pi_{k+1}} q_k\|_\infty + \|T_{\pi_{k+1}} q_k - T_{\pi_{k+1}} q_{\pi_{k+1}}\|_\infty \\ &\leq \|Tq_* - Tq_k\|_\infty + \gamma\|q_k - q_{\pi_{k+1}}\|_\infty \\ &\leq \gamma\|q_* - q_k\|_\infty + \gamma\left(\|q_k - q_*\|_\infty + \|q_* - q_{\pi_{k+1}}\|_\infty\right) \\ &\leq \frac{2\gamma}{1-\gamma}\|q_* - q_k\|_\infty.\end{aligned}$$

Error propagation for AVI

1. Bounding: $\|q_* - q_k\|_\infty$:

$$\begin{aligned} \|q_* - q_k\|_\infty &= \|q_* - Tq_{k-1} - \epsilon_k\|_\infty \\ &\leq \|Tq_* - Tq_{k-1}\|_\infty + \epsilon \\ &\leq \gamma\|q_* - q_{k-1}\|_\infty + \epsilon \\ &\leq \frac{\epsilon}{1-\gamma}. \end{aligned}$$

2. From $\|q_* - q_k\|_\infty$ to $\|q_* - q_{\pi_{k+1}}\|_\infty$ ($\pi_{k+1} = \mathcal{G}q_k$):

$$\begin{aligned} \|q_* - q_{\pi_{k+1}}\|_\infty &\leq \|Tq_* - T_{\pi_{k+1}}q_k\|_\infty + \|T_{\pi_{k+1}}q_k - T_{\pi_{k+1}}q_{\pi_{k+1}}\|_\infty \\ &\leq \|Tq_* - Tq_k\|_\infty + \gamma\|q_k - q_{\pi_{k+1}}\|_\infty \\ &\leq \gamma\|q_* - q_k\|_\infty + \gamma\left(\|q_k - q_*\|_\infty + \|q_* - q_{\pi_{k+1}}\|_\infty\right) \\ &\leq \frac{2\gamma}{1-\gamma}\|q_* - q_k\|_\infty. \end{aligned}$$

Example: the Optimal Replacement Problem

State: level of wear (x) of an object (e.g., a car).
Action: $\{(\text{R})\text{eplace}, (\text{K})\text{eep}\}$.
Cost:

- $c(x, R) = C$
- $c(x, K) = c(x)$ maintenance plus extra costs.

Dynamics:

- $p(y|x, R) \sim d(y) = \beta \exp^{-\beta y} \mathbb{1}\{y \geq 0\}$,
- $p(y|x, K) \sim d(y - x) = \beta \exp^{-\beta(y-x)} \mathbb{1}\{y \geq x\}$.

Problem: Minimize the discounted expected cost over an infinite horizon.

Example: the Optimal Replacement Problem

State: level of wear (x) of an object (e.g., a car).
Action: $\{(\text{R})\text{eplace}, (\text{K})\text{eep}\}$.
Cost:

- $c(x, R) = C$
- $c(x, K) = c(x)$ maintenance plus extra costs.

Dynamics:

- $p(y|x, R) \sim d(y) = \beta \exp^{-\beta y} \mathbb{1}\{y \geq 0\}$,
- $p(y|x, K) \sim d(y - x) = \beta \exp^{-\beta(y-x)} \mathbb{1}\{y \geq x\}$.

Problem: Minimize the discounted expected cost over an infinite horizon.

Example: the Optimal Replacement Problem

State: level of wear (x) of an object (e.g., a car).
Action: $\{(\text{R})\text{eplace}, (\text{K})\text{eep}\}$.
Cost:

- $c(x, R) = C$
- $c(x, K) = c(x)$ maintenance plus extra costs.

Dynamics:

- $p(y|x, R) \sim d(y) = \beta \exp^{-\beta y} \mathbb{1}\{y \geq 0\}$,
- $p(y|x, K) \sim d(y - x) = \beta \exp^{-\beta(y-x)} \mathbb{1}\{y \geq x\}$.

Problem: Minimize the discounted expected cost over an infinite horizon.

Example: the Optimal Replacement Problem

State: level of wear (x) of an object (e.g., a car).
Action: $\{(\mathrm{R})\mathrm{eplace}, (\mathrm{K})\mathrm{eep}\}$.
Cost:

- $c(x, R) = C$
- $c(x, K) = c(x)$ maintenance plus extra costs.

Dynamics:

- $p(y|x, R) \sim d(y) = \beta \exp^{-\beta y} \mathbb{1}\{y \geq 0\}$,
- $p(y|x, K) \sim d(y - x) = \beta \exp^{-\beta(y-x)} \mathbb{1}\{y \geq x\}$.

Problem: Minimize the discounted expected cost over an infinite horizon.

Example: the Optimal Replacement Problem

State: level of wear (x) of an object (e.g., a car).
Action: $\{(\text{R})\text{eplace}, (\text{K})\text{eep}\}$.
Cost:

- $c(x, R) = C$
- $c(x, K) = c(x)$ maintenance plus extra costs.

Dynamics:

- $p(y|x, R) \sim d(y) = \beta \exp^{-\beta y} \mathbb{1}\{y \geq 0\}$,
- $p(y|x, K) \sim d(y - x) = \beta \exp^{-\beta(y-x)} \mathbb{1}\{y \geq x\}$.

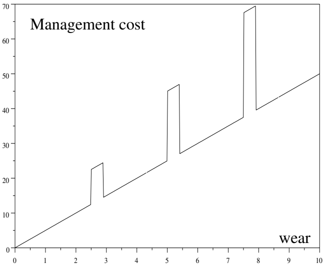
Problem: Minimize the discounted expected cost over an infinite horizon.

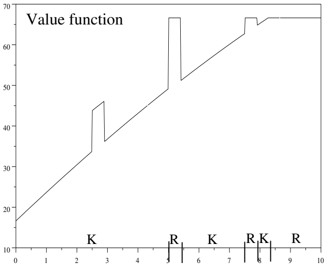
Example: the Optimal Replacement Problem

The optimal value function satisfies

$$v_*(x) = \min\left\{ \underbrace{c(x) + \gamma \int_0^\infty d(y-x)v_*(y)dy}_{(K)eep},\ \underbrace{C + \gamma \int_0^\infty d(y)v_*(y)dy}_{R)eplace} \right\}$$

Optimal policy: action that attains the minimum





Example: the Optimal Replacement Problem

Linear approximation space

$$\mathcal{F} := \left\{ v_n(x) = \sum_{k=1}^{20} \alpha_k \cos(k\pi \frac{x}{x_{\max}}) \right\}.$$

Collect N samples on a uniform grid:

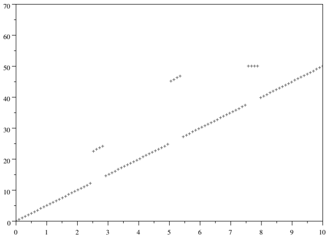


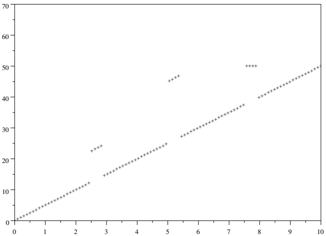
Figure: Left: the *target* values computed as $\{Tv_0(x_n)\}_{1 \leq n \leq N}$. Right: the approximation $v_1 \in \mathcal{F}$ of the target function Tv_0.

Example: the Optimal Replacement Problem

Linear approximation space

$$\mathcal{F} := \left\{ v_n(x) = \sum_{k=1}^{20} \alpha_k \cos(k\pi \frac{x}{x_{\max}}) \right\}.$$

Collect N samples on a uniform grid:



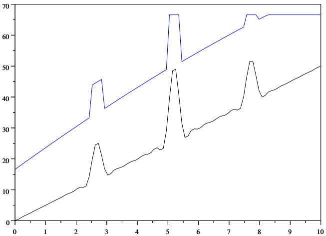
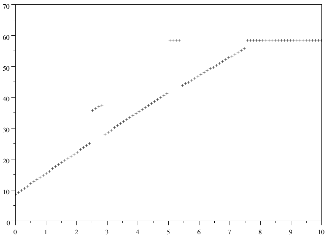


Figure: Left: the *target* values computed as $\{Tv_0(x_n)\}_{1 \le n \le N}$. Right: the approximation $v_1 \in \mathcal{F}$ of the target function Tv_0.

Example: the Optimal Replacement Problem

One more step:



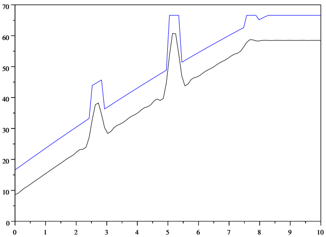


Figure: Left: the *target* values computed as $\{Tv_1(x_n)\}_{1\leq n\leq N}$. Right: the approximation $v_2 \in \mathcal{F}$ of Tv_1.

Example: the Optimal Replacement Problem

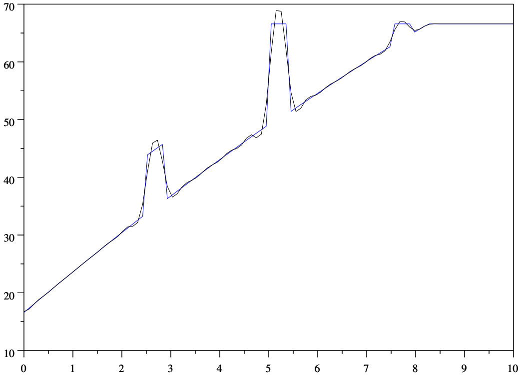


Figure: The approximation $v_{20} \in \mathcal{F}$.

Approximate MPI-Q

$\boxed{(q_k) \text{ are represented in } \mathcal{F} \subseteq \mathbb{R}^{X \times A}}$

- $\pi_{k+1} \leftarrow \mathcal{G} q_k$
- $q_{k+1} \leftarrow (T_{\pi_{k+1}})^m q_k$

Policy update

In state x, the greedy action is estimated by:

$$\pi_{k+1}(x) = \arg\max_{a \in A} q_k(x, a)$$

Value function update

1. **Point-wise estimation** through rollouts of length m**:**
   For N state-action pairs $(x^{(i)}, a^{(i)}) \sim \mu$, compute an unbiased estimate of $\left[\left(T_{\pi_{k+1}}\right)^m q_k\right](x^{(i)}, a^{(i)})$ (using $a^{(i)}$, then π_{k+1} m times)

$$\widehat{q}_{k+1}(x^{(i)}, a^{(i)}) = \sum_{t=0}^{m-1} \gamma^t r_t^{(i)} + \gamma^m q_k(x_m^{(i)}, \pi_{k+1}(x^{(i)}))$$

2. **Generalisation** through regression**:**
   q_{k+1} is computed as the best fit of these estimates in $\mathcal{F}$

$$q_{k+1} = \arg\min_{q \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} \left(q(x^{(i)}, a^{(i)}) - \widehat{q}_{k+1}(x^{(i)}, a^{(i)}) \right)^2$$

Approximate MPI-Q

$\boxed{(q_k) \text{ are represented in } \mathcal{F} \subseteq \mathbb{R}^{X \times A}}$

- $\pi_{k+1} \quad \leftarrow \mathcal{G} q_k$
- $q_{k+1} \quad \leftarrow (T_{\pi_{k+1}})^m q_k$

Policy update

In state x, the greedy action is estimated by:

$$\pi_{k+1}(x) = \arg\max_{a \in A} q_k(x, a)$$

Value function update

1. **Point-wise estimation** through rollouts of length m:
   For N state-action pairs $(x^{(i)}, a^{(i)}) \sim \mu$, compute an unbiased estimate of $\left[(T_{\pi_{k+1}})^m q_k \right] (x^{(i)}, a^{(i)})$ (using $a^{(i)}$, then π_{k+1} m times)

$$\widehat{q}_{k+1}(x^{(i)}, a^{(i)}) = \textstyle\sum_{t=0}^{m-1} \gamma^t r_t^{(i)} + \gamma^m q_k(x_m^{(i)}, \pi_{k+1}(x^{(i)}))$$

2. **Generalisation** through regression:
   q_{k+1} is computed as the best fit of these estimates in $\mathcal{F}$

$$q_{k+1} = \arg\min_{q \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} \left(q(x^{(i)}, a^{(i)}) - \widehat{q}_{k+1}(x^{(i)}, a^{(i)}) \right)^2$$

Approximate Modified Policy Iteration

AMPI-Q is an instance of:

$$\pi_{k+1} = \mathcal{G} q_k$$
$$q_{k+1} = (T_{\pi_{k+1}})^m q_k + \epsilon_{k+1}$$

where (regression literature):

$$\|\epsilon_{k+1}\|_{2,\mu} = \|q_{k+1} - (T_{\pi_{k+1}})^m q_k\|_{2,\mu} \leq O\left(\underbrace{\sup_{g,\pi \in \mathcal{F}} \inf_{f \in \mathcal{F}} \|f - (T_\pi)^m g\|_{2,\mu}}_{Approx.error} + \underbrace{\frac{1}{\sqrt{n}}}_{Estim.error} \right)$$

Theorem (Scherrer et al., 2014)

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. The loss due to running policy π_k instead of the optimal policy π_* satisfies

$$\limsup_{k \to \infty} \|q_* - q_{\pi_k}\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2} \epsilon.$$

Approximate Modified Policy Iteration

AMPI-Q is an instance of:

$$\pi_{k+1} = \mathcal{G} q_k$$
$$q_{k+1} = (T_{\pi_{k+1}})^m q_k + \epsilon_{k+1}$$

where (regression literature):

$$\|\epsilon_{k+1}\|_{2,\mu} = \|q_{k+1} - (T_{\pi_{k+1}})^m q_k\|_{2,\mu} \leq O\left(\underbrace{\sup_{g,\pi \in \mathcal{F}} \inf_{f \in \mathcal{F}} \|f - (T_\pi)^m g\|_{2,\mu}}_{Approx.error} + \underbrace{\frac{1}{\sqrt{n}}}_{Estim.error} \right)$$

Theorem (Scherrer et al., 2014)

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. The loss due to running policy π_k instead of the optimal policy π_* satisfies

$$\limsup_{k\to\infty} \|q_* - q_{\pi_k}\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2}\epsilon.$$

Classification-based MPI

(v_k) represented in $\mathcal{F} \subseteq \mathbb{R}^X$
(π_k) represented in $\Pi \subseteq A^X$

- $v_k \leftarrow (T_{\pi_k})^m v_{k-1}$
- $\pi_{k+1} \leftarrow \mathcal{G}\big[(T_{\pi_k})^m v_{k-1}\big]$

Value function update

Similar to AMPI-Q:

1. **Point-wise estimation** through rollouts of length m:
   Draw N states $x^{(i)} \sim \mu$

$$\widehat{v}_{k+1}(x^{(i)}) = \sum_{t=0}^{m-1} \gamma^t r_t^{(i)} + \gamma^m v_{k-1}(x_m^{(i)})$$

2. **Generalisation** through regression

$$v_k = \arg\min_{v \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} \Big(v(x^{(i)}) - \widehat{v}_k(x^{(i)}) \Big)^2$$

Classification-based MPI

(v_k) represented in $\mathcal{F} \subseteq \mathbb{R}^X$
(π_k) represented in $\Pi \subseteq A^X$

- $v_k \leftarrow (T_{\pi_k})^m v_{k-1}$
- $\pi_{k+1} \leftarrow \mathcal{G}\left[(T_{\pi_k})^m v_{k-1}\right]$

Value function update

Similar to AMPI-Q:

1. **Point-wise estimation** through rollouts of length m:
   Draw N states $x^{(i)} \sim \mu$

$$\widehat{v}_{k+1}(x^{(i)}) = \sum_{t=0}^{m-1} \gamma^t r_t^{(i)} + \gamma^m v_{k-1}(x_m^{(i)})$$

2. **Generalisation** through regression

$$v_k = \arg\min_{v \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} \left(v(x^{(i)}) - \widehat{v}_k(x^{(i)}) \right)^2$$

Classification-based MPI

Policy update

When $\pi = \mathcal{G}\big[(T_{\pi_k})^m v_{k-1}\big]$, for each $x \in \mathcal{X}$, we have

$$\underbrace{\big[T_\pi (T_{\pi_k})^m v_{k-1}\big](x)}_{Q_k\left(x,\pi(x)\right)} = \max_{a \in A} \underbrace{\big[T_a (T_{\pi_k})^m v_{k-1}\big](x)}_{Q_k(x,a)}$$

1. For N states $x^{(i)} \sim \mu$, for all actions a, compute an unbiased estimate of $[T_a (T_{\pi_k})^m v_{k-1}](x^{(i)})$ from M rollouts (using a, then π_{k+1} m times):

$$\widehat{Q}_k(x^{(i)}, a) = \frac{1}{M} \sum_{j=1}^{M} \sum_{t=0}^{m} \gamma^t r_t^{(i,j)} + \gamma^{m+1} v_{k-1}(x_{m+1}^{(i,j)})$$

2. π_{k+1} is the result of the (cost-sensitive) classifier:

$$\pi_{k+1} = \arg\min_{\pi \in \Pi} \frac{1}{N} \sum_{i=1}^{N} \Big[\max_{a \in A} \widehat{Q}_k(x^{(i)}, a) - \widehat{Q}_k\big(x^{(i)}, \pi(x^{(i)})\big)\Big]$$

CBMPI is an instance of:

$$v_k = (T_{\pi_k})^m v_{k-1} + \epsilon_k$$
$$\pi_{k+1} = \hat{\mathcal{G}}_{\epsilon'_{k+1}} (T_{\pi_k})^m v_{k-1}$$

where (regression & classification literature):

$$\|\epsilon_k\|_{2,\mu} = \|v_k - (T_{\pi_k})^m v_{k-1}\|_{2,\mu} \leq O\left(\sup_{g,\pi \in \mathcal{F}} \inf_{f \in \mathcal{F}} \|f - (T_\pi)^m g\|_{2,\mu} + \frac{1}{\sqrt{n}} \right)$$

$$\|\epsilon'_k\|_{1,\mu} = O\left(\sup_{v \in \mathcal{F}, \pi'} \inf_{\pi \in \Pi} \sum_{x \in X} \left[\max_a Q_{\pi',v}(x,a) - Q_{\pi',v}(x,\pi(x)) \right] \mu(x) + \frac{1}{\sqrt{N}} \right)$$

Theorem (Scherrer et al., 2014)

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. The loss due to running policy π_k instead of the optimal policy π_* satisfies

$$\limsup_{k \to \infty} \|q_* - q_{\pi_k}\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2} (2\gamma^{m+1} \epsilon + \epsilon').$$

CBMPI is an instance of:

$$v_k = (T_{\pi_k})^m v_{k-1} + \epsilon_k$$
$$\pi_{k+1} = \hat{\mathcal{G}}_{\epsilon'_{k+1}} (T_{\pi_k})^m v_{k-1}$$

where (regression & classification literature):

$$\|\epsilon_k\|_{2,\mu} = \|v_k - (T_{\pi_k})^m v_{k-1}\|_{2,\mu} \leq O\left(\sup_{g,\pi \in \mathcal{F}} \inf_{f \in \mathcal{F}} \|f - (T_\pi)^m g\|_{2,\mu} + \frac{1}{\sqrt{n}} \right)$$

$$\|{\epsilon'}_k\|_{1,\mu} = O\left(\sup_{v \in \mathcal{F}, \pi'} \inf_{\pi \in \Pi} \sum_{x \in X} \Big[\max_a Q_{\pi',v}(x,a) - Q_{\pi',v}(x,\pi(x)) \Big] \mu(x) + \frac{1}{\sqrt{N}} \right)$$

Theorem (Scherrer et al., 2014)

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. The loss due to running policy π_k instead of the optimal policy π_* satisfies

$$\limsup_{k \to \infty} \|q_* - q_{\pi_k}\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2} (2\gamma^{m+1}\epsilon + \epsilon').$$

Illustration of approximation on Tetris

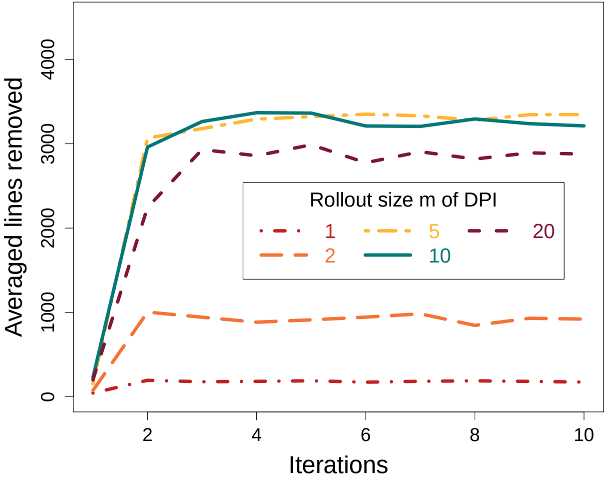
1. Approximation architecture for v:

   "**An expert says that**" for all state x,

$$
\begin{aligned}
v(x) \simeq v_\theta(x) & \\
= \theta_0 & \qquad \text{Constant} \\
+ \theta_1 h_1(x) + \theta_2 h_2(x) + \cdots + \theta_{10} h_{10}(x) & \qquad \text{column height} \\
+ \theta_{11} \Delta h_1(x) + \theta_{12} \Delta h_2(x) + \cdots + \theta_{19} \Delta h_9(x) & \qquad \text{height variation} \\
+ \theta_{20} \max_k h_k(x) & \qquad \text{max height} \\
+ \theta_{21} L(x) & \qquad \# \text{ holes} \\
+ \ldots &
\end{aligned}
$$

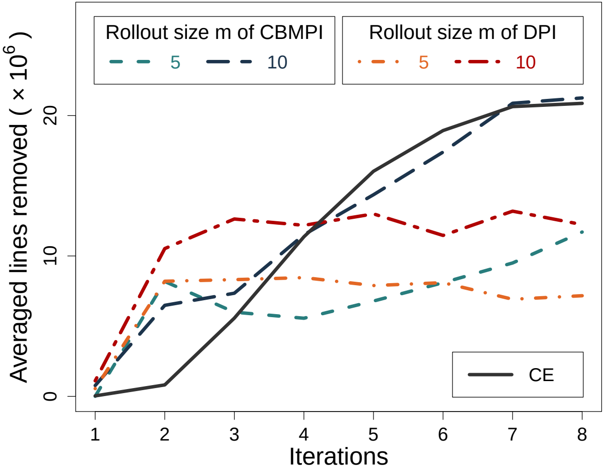
2. The classifier is based on the same features to compute a score function for the (deterministic) next state.
3. Sampling Scheme: play

"Small" Tetris (10×10)



Learning curves of CBMPI algorithm on the small 10×10 board. The results are averaged over 100 runs of the algorithms. $B = 8.10^6$ samples per iteration.

Tetris (10×20)



Learning curves of CE, DPI, and CBMPI algorithms on the large 10×20 board. The results are averaged over 100 runs of the algorithms. $B_{DPI/CBMPI} = 16.10^6$ samples per iteration. $B_{CE} = 1700.10^6$.

Least Squares PI

- Exact PI: $\pi_k = \mathcal{G} v_{k-1}$ and $v_k = v_{\pi_k}$
- The difficult problem is to estimate the value v_π of some policy π:

$$v_\pi = r + \gamma P_\pi v_\pi \quad \Leftrightarrow \quad v_\pi = T_\pi v_\pi \quad \Leftrightarrow \quad v_\pi = (I - \gamma P_\pi)^{-1} r$$

- Look for a linear approximation $\hat{v_\pi}(x) = \sum_{j=1}^m w_j \phi_j(x)$ or $\hat{v_\pi} = \Phi w$

$$\Phi = \begin{pmatrix} \phi(1)' \\ \vdots \\ \phi(N)' \end{pmatrix} = \underbrace{\begin{pmatrix} \phi_1 & \dots & \phi_m \end{pmatrix}}_{\text{linearly independent}} \text{ and } w = \begin{pmatrix} w_1 \\ \vdots \\ w_m \end{pmatrix}$$

Projection

- Projection onto span $(\Phi) = \{\Phi w; w \in \mathbb{R}^m\}$
  - Let $\xi > 0$ be a distribution on the state space $\{1, \cdots, N\}$
  - Quadratic weighted norm: $\|v\|_{2,\xi} = \sqrt{\sum_x \xi(x) v(x)^2}$
  - Orthogonal projection: $\Pi(v) = \arg\min_{\hat{v} \in \text{span}(\Phi)} \|\hat{v} - v\|_{2,\xi}$
  - Writing $\Xi = \text{diag}(\xi)$, Π has the following closed-form:

$$\Pi = \Phi(\Phi' \Xi \Phi)^{-1} \Phi' \Xi$$

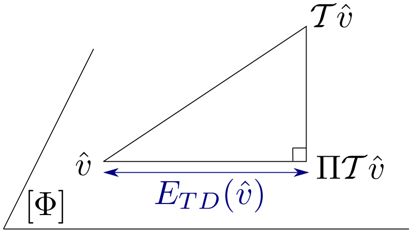
- Ideally, one would like to compute the "best" approximation

$$\hat{v}_{best} = \Phi w_{best} = \Pi v_\pi = \Pi(I - \gamma P_\pi)^{-1} r.$$

  Linear regression + Monte-Carlo (full trajectories), high variance ☹
- Alternatives based on one-step samples: $\hat{v} \simeq T_\pi \hat{v}$

TD fix point method

One looks for $\hat{v}_{TD} \in \text{span}(\Phi)$ satisfying $\hat{v}_{TD} = \Pi T_\pi \hat{v}_{TD}$.



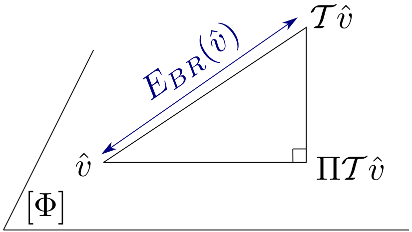
When the inverse exists, it can be proved that $\hat{v}_{TD} = \Phi w_{TD}$ with

$$w_{TD} = (\underbrace{\Phi'\Xi(I - \gamma P)\Phi}_{A \in \mathbb{R}^{m\times m}})^{-1} \underbrace{\Phi'\Xi r}_{b \in \mathbb{R}^m},$$

where
$$A = \mathbb{E}_{x\sim\xi,\ y\sim P_\pi(\cdot|x)}\left[\phi(x)\big(\phi(x) - \gamma\phi(y)\big)'\right]$$
$$b = \mathbb{E}_{x\sim\xi}\left[\phi(x)r(x)\right]$$

Bellman Residual minimization method

One looks for $\hat{v} \in \text{span}(\Phi)$ minimizing $E_{BR}(\hat{v}) := \|\hat{v} - T_\pi \hat{v}\|_{2,\xi}$.



Since $E_{BR}(\Phi w) = \| \underbrace{\Phi w - \gamma P \Phi w}_{\Psi w} - r \|_{2,\xi}$, $\quad \Psi = (I - \gamma P)\Phi$, it can be seen that $\hat{v}_{BR} = \Phi w_{BR}$ with

$$w_{BR} = (\underbrace{\Psi' \Xi \Psi}_{A \in \mathbb{R}^{m \times m}})^{-1} \underbrace{\Psi' \Xi r}_{b \in \mathbb{R}^m},$$

where $$A = \mathbb{E}_{x \sim \xi,\ y, y' \sim P_\pi(\cdot|x)} \left[\left(\phi(x - \gamma \phi(y) \right) \left(\phi(x) - \gamma \phi(y') \right)' \right]$$

$$b = \mathbb{E}_{x \sim \xi,\ y \sim P_\pi(\cdot|x)} \left[\left(\phi(x) - \gamma \phi(y) \right) r(x) \right]$$

Guarantees for BR, TD and LSPI

Proposition (Williams and Baird, 1993)

$$\|v_\pi - \hat{v}_{BR}\|_\infty \leq \frac{1+\gamma}{1-\gamma}\|v_\pi - \hat{v}_{best}\|_\infty.$$

Proposition (Tsitsiklis and Van Roy, 1997)

If ξ is the stationary distribution of P_π, then

$$\|v_\pi - \hat{v}_{TD}\|_{2,\xi} \leq \frac{1}{1-\gamma}\|v_\pi - \hat{v}_{best}\|_{2,\xi}.$$

Approximate PI: $\pi_k = \mathcal{G} v_{k-1}$ and $v_k = v_{\pi_k} + \epsilon_k$

Theorem

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. The loss due to running policy π_k instead of the optimal policy π_* satisfies

$$\limsup_{k\to\infty} \|v_* - v_{\pi_k}\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2}\epsilon.$$

Guarantees for BR, TD and LSPI

Proposition (Williams and Baird, 1993)

$$\|v_\pi - \hat{v}_{BR}\|_\infty \leq \frac{1+\gamma}{1-\gamma}\|v_\pi - \hat{v}_{best}\|_\infty.$$

Proposition (Tsitsiklis and Van Roy, 1997)

If ξ is the stationary distribution of P_π, then

$$\|v_\pi - \hat{v}_{TD}\|_{2,\xi} \leq \frac{1}{1-\gamma}\|v_\pi - \hat{v}_{best}\|_{2,\xi}.$$

Approximate PI: $\pi_k = \mathcal{G} v_{k-1}$ and $v_k = v_{\pi_k} + \epsilon_k$

Theorem

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. The loss due to running policy π_k instead of the optimal policy π_* satisfies

$$\limsup_{k\to\infty} \|v_* - v_{\pi_k}\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2}\epsilon.$$

Error propagation for API

1. Approximate monotonicity:

$$
\begin{aligned}
v_{\pi_{k+1}} - v_{\pi_k} &= (I - \gamma P_{\pi_{k+1}})^{-1} r_{\pi_{k+1}} - v_{\pi_k} \\
&= (I - \gamma P_{\pi_{k+1}})^{-1} \left(r_{\pi_{k+1}} + \gamma P_{\pi_{k+1}} v_{\pi_k} - v_{\pi_k} \right) \\
&= (I - \gamma P_{\pi_{k+1}})^{-1} \left(T_{\pi_{k+1}} v_{\pi_k} - T_{\pi_k} v_{\pi_k} \right) \\
&= (I - \gamma P_{\pi_{k+1}})^{-1} \left(T_{\pi_{k+1}} (v_k - \epsilon_k) - T_{\pi_k} (v_k - \epsilon_k) \right) \\
&\geq (I - \gamma P_{\pi_{k+1}})^{-1} \left(-\gamma P_{\pi_{k+1}} \epsilon_k + \gamma P_{\pi_k} \epsilon_k \right) \geq -\frac{2\gamma}{1-\gamma} \epsilon
\end{aligned}
$$

2. Distance to v_*:

$$
\begin{aligned}
v_* - v_{\pi_{k+1}} &= T_{\pi_*} v_* - T_{\pi_*} v_{\pi_k} + T_{\pi_*} v_{\pi_k} - T_{\pi_{k+1}} v_{\pi_k} + T_{\pi_{k+1}} v_{\pi_k} - T_{\pi_{k+1}} v_{\pi_{k+1}} \\
&\leq \gamma P_{\pi_*} (v_* - v_{\pi_k}) + \gamma P_{\pi_{k+1}} (v_{\pi_k} - v_{\pi_{k+1}})
\end{aligned}
$$

And thus:

$$
\| v_* - v_{\pi_{k+1}} \|_\infty \leq \gamma \| v_* - v_{\pi_k} \|_\infty + \frac{2\gamma}{1-\gamma} \epsilon \leq \frac{2\gamma}{(1-\gamma)^2} \epsilon.
$$

Error propagation for API

1. Approximate monotonicity:

$$\begin{aligned}
v_{\pi_{k+1}} - v_{\pi_k} &= (I - \gamma P_{\pi_{k+1}})^{-1} r_{\pi_{k+1}} - v_{\pi_k} \\
&= (I - \gamma P_{\pi_{k+1}})^{-1} \left(r_{\pi_{k+1}} + \gamma P_{\pi_{k+1}} v_{\pi_k} - v_{\pi_k} \right) \\
&= (I - \gamma P_{\pi_{k+1}})^{-1} \left(T_{\pi_{k+1}} v_{\pi_k} - T_{\pi_k} v_{\pi_k} \right) \\
&= (I - \gamma P_{\pi_{k+1}})^{-1} \left(T_{\pi_{k+1}} (v_k - \epsilon_k) - T_{\pi_k} (v_k - \epsilon_k) \right) \\
&\geq (I - \gamma P_{\pi_{k+1}})^{-1} \left(-\gamma P_{\pi_{k+1}} \epsilon_k + \gamma P_{\pi_k} \epsilon_k \right) \geq - \frac{2\gamma}{1-\gamma} \epsilon
\end{aligned}$$

2. Distance to v_*:

$$\begin{aligned}
v_* - v_{\pi_{k+1}} &= T_{\pi_*} v_* - T_{\pi_*} v_{\pi_k} + T_{\pi_*} v_{\pi_k} - T_{\pi_{k+1}} v_{\pi_k} + T_{\pi_{k+1}} v_{\pi_k} - T_{\pi_{k+1}} v_{\pi_{k+1}} \\
&\leq \gamma P_{\pi_*} (v_* - v_{\pi_k}) + \gamma P_{\pi_{k+1}} (v_{\pi_k} - v_{\pi_{k+1}})
\end{aligned}$$

And thus:

$$\|v_* - v_{\pi_{k+1}}\|_\infty \leq \gamma \|v_* - v_{\pi_k}\|_\infty + \frac{2\gamma}{1-\gamma} \epsilon \leq \frac{2\gamma}{(1-\gamma)^2} \epsilon.$$

Error propagation for API

1. Approximate monotonicity:

$$\begin{aligned} v_{\pi_{k+1}} - v_{\pi_k} &= (I - \gamma P_{\pi_{k+1}})^{-1} r_{\pi_{k+1}} - v_{\pi_k} \\ &= (I - \gamma P_{\pi_{k+1}})^{-1} \left(r_{\pi_{k+1}} + \gamma P_{\pi_{k+1}} v_{\pi_k} - v_{\pi_k} \right) \\ &= (I - \gamma P_{\pi_{k+1}})^{-1} \left(T_{\pi_{k+1}} v_{\pi_k} - T_{\pi_k} v_{\pi_k} \right) \\ &= (I - \gamma P_{\pi_{k+1}})^{-1} \left(T_{\pi_{k+1}}(v_k - \epsilon_k) - T_{\pi_k}(v_k - \epsilon_k) \right) \\ &\geq (I - \gamma P_{\pi_{k+1}})^{-1} \left(-\gamma P_{\pi_{k+1}} \epsilon_k + \gamma P_{\pi_k} \epsilon_k \right) \geq -\frac{2\gamma}{1-\gamma} \epsilon \end{aligned}$$

2. Distance to v_*:

$$\begin{aligned} v_* - v_{\pi_{k+1}} &= T_{\pi_*} v_* - T_{\pi_*} v_{\pi_k} + T_{\pi_*} v_{\pi_k} - T_{\pi_{k+1}} v_{\pi_k} + T_{\pi_{k+1}} v_{\pi_k} - T_{\pi_{k+1}} v_{\pi_{k+1}} \\ &\leq \gamma P_{\pi_*}(v_* - v_{\pi_k}) + \gamma P_{\pi_{k+1}}(v_{\pi_k} - v_{\pi_{k+1}}) \end{aligned}$$

And thus:

$$\|v_* - v_{\pi_{k+1}}\|_\infty \leq \gamma \|v_* - v_{\pi_k}\|_\infty + \frac{2\gamma}{1-\gamma}\epsilon \leq \frac{2\gamma}{(1-\gamma)^2}\epsilon.$$

Outline for Part 2

- Approximate Dynamic Programming
  - Approximate VI: Fitted-Q Iteration
  - Approximate MPI: AMPI-Q, CBMPI
  - Approximate PI: LSPI
    - Projected value estimation: LSTD,LSBR

- Advanced topics
  - Non-stationary policies for stationary MDPs: NSVI, NSPI, NSMPI
  - Max-norm vs L_p-norm, concentrability coefficients: CPI, API(α), PSDP$_\infty$

Algorithms

App. Value Iteration

$\pi_{k+1} \leftarrow \mathcal{G} v_k$
$v_{k+1} \leftarrow T v_k + \epsilon_k = T_{\pi_{k+1}} v_k + \epsilon_k$

App. Policy Iteration

$\pi_{k+1} \leftarrow \mathcal{G} v_k$
$v_{k+1} \leftarrow v_{\pi_{k+1}} = (T_{\pi_{k+1}})^{\infty} v_k + \epsilon_k$

App. Modified Policy Iteration

$\pi_{k+1} \leftarrow \mathcal{G} v_k$
$v_{k+1} \leftarrow (T_{\pi_{k+1}})^m v_k + \epsilon_k \qquad (1 \leq m \leq \infty)$

Theorem

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. The loss due to running policy π_k instead of the optimal policy π_* satisfies

$$\limsup_{k \to \infty} \|v_* - v_{\pi_k}\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2} \epsilon.$$

Algorithms

App. Value Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow T v_k + \epsilon_k = T_{\pi_{k+1}} v_k + \epsilon_k$$

App. Policy Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow v_{\pi_{k+1}} = (T_{\pi_{k+1}})^\infty v_k + \epsilon_k$$

App. Modified Policy Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow (T_{\pi_{k+1}})^m v_k + \epsilon_k \qquad (1 \le m \le \infty)$$

Theorem

Assume $\|\epsilon_k\|_\infty \le \epsilon$. The loss due to running policy π_k instead of the optimal policy π_* satisfies

$$\limsup_{k \to \infty} \|v_* - v_{\pi_k}\|_\infty \le \frac{2\gamma}{(1-\gamma)^2}\epsilon.$$

Algorithms

App. Value Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow T v_k + \epsilon_k = T_{\pi_{k+1}} v_k + \epsilon_k$$

App. Policy Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow v_{\pi_{k+1}} = (T_{\pi_{k+1}})^\infty v_k + \epsilon_k$$

App. Modified Policy Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow (T_{\pi_{k+1}})^m v_k + \epsilon_k \qquad (1 \leq m \leq \infty)$$

Theorem

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. The loss due to running policy π_k instead of the optimal policy π_* satisfies

$$\limsup_{k\to\infty} \|v_* - v_{\pi_k}\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2}\epsilon.$$

Algorithms

App. Value Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow T v_k + \epsilon_k = T_{\pi_{k+1}} v_k + \epsilon_k$$

App. Policy Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow v_{\pi_{k+1}} = (T_{\pi_{k+1}})^\infty v_k + \epsilon_k$$

App. Modified Policy Iteration

$$\pi_{k+1} \leftarrow \mathcal{G} v_k$$
$$v_{k+1} \leftarrow (T_{\pi_{k+1}})^m v_k + \epsilon_k \qquad (1 \leq m \leq \infty)$$

Theorem

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. The loss due to running policy π_k instead of the optimal policy π_* satisfies

$$\limsup_{k\to\infty} \|v_* - v_{\pi_k}\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2}\epsilon.$$

Algorithms

App. Value Iteration

$\pi_{k+1} \leftarrow \mathcal{G} v_k$
$v_{k+1} \leftarrow T v_k + \epsilon_k = T_{\pi_{k+1}} v_k + \epsilon_k$

App. Policy Iteration

$\pi_{k+1} \leftarrow \mathcal{G} v_k$
$v_{k+1} \leftarrow v_{\pi_{k+1}} = (T_{\pi_{k+1}})^\infty v_k + \epsilon_k$

App. Modified Policy Iteration

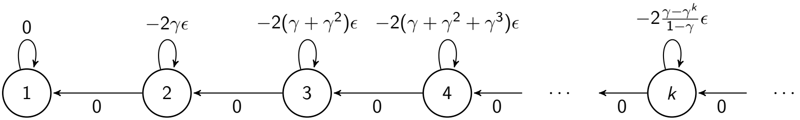
$\pi_{k+1} \leftarrow \mathcal{G} v_k$
$v_{k+1} \leftarrow (T_{\pi_{k+1}})^m v_k + \epsilon_k \qquad (1 \le m \le \infty)$

Theorem

Assume $\|\epsilon_k\|_\infty \le \epsilon$. The loss due to running policy π_k instead of the optimal policy π_* satisfies

$$\limsup_{k \to \infty} \|v_* - v_{\pi_k}\|_\infty \le \frac{2\gamma}{(1-\gamma)^2}\epsilon.$$

Tightness of the bound for AVI

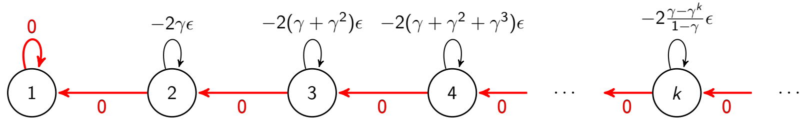


	1	2	3	4	$\ldots$
v_0	0	0	0	0	$\ldots$
v_1	$-\epsilon$	ϵ	0	0	$\ldots$
v_2	$-\gamma\epsilon$	$-\epsilon-\gamma\epsilon$	$\epsilon+\gamma\epsilon$	0	$\ldots$
v_3	$-\gamma^2\epsilon$	$-\gamma^2\epsilon$	$-\epsilon-\gamma\epsilon-\gamma^2\epsilon$	$\epsilon+\gamma\epsilon+\gamma^2\epsilon$	$\ldots$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$

State 2: $0+\gamma(-\epsilon) = -2\gamma\epsilon+\gamma\epsilon$
State 3: $0+\gamma(-\epsilon-\gamma\epsilon) = -2(\gamma+\gamma^2)\epsilon+\gamma(\epsilon+\gamma\epsilon)$

$$v_{\pi_k}(k) = \sum_{t=0}^{\infty}\gamma^t\left(-2\frac{\gamma-\gamma^k}{1-\gamma}\epsilon\right) = -2\frac{\gamma-\gamma^k}{(1-\gamma)^2}\epsilon \xrightarrow{k\to\infty} -\frac{2\gamma}{(1-\gamma)^2}\epsilon$$

Tightness of the bound for AVI



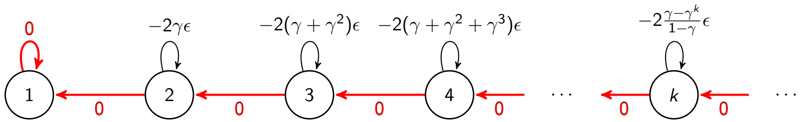
	1	2	3	4	$\ldots$
v_0	0	0	0	0	$\ldots$
v_1	$-\epsilon$	ϵ	0	0	$\ldots$
v_2	$-\gamma\epsilon$	$-\epsilon-\gamma\epsilon$	$\epsilon+\gamma\epsilon$	0	$\ldots$
v_3	$-\gamma^2\epsilon$	$-\gamma^2\epsilon$	$-\epsilon-\gamma\epsilon-\gamma^2\epsilon$	$\epsilon+\gamma\epsilon+\gamma^2\epsilon$	$\ldots$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$

State 2: $0+\gamma(-\epsilon) = -2\gamma\epsilon+\gamma\epsilon$

State 3: $0+\gamma(-\epsilon-\gamma\epsilon) = -2(\gamma+\gamma^2)\epsilon+\gamma(\epsilon+\gamma\epsilon)$

$$v_{\pi_k}(k) = \sum_{t=0}^{\infty}\gamma^t\left(-2\frac{\gamma-\gamma^k}{1-\gamma}\epsilon\right) = -2\frac{\gamma-\gamma^k}{(1-\gamma)^2}\epsilon \xrightarrow{k\to\infty} -\frac{2\gamma}{(1-\gamma)^2}\epsilon$$

Tightness of the bound for AVI

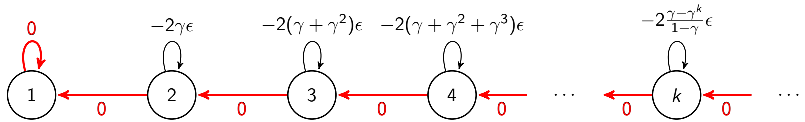


	1	2	3	4	...
v_0	0	0	0	0	...
v_1	$-\epsilon$	ϵ	0	0	...
v_2	$-\gamma\epsilon$	$-\epsilon-\gamma\epsilon$	$\epsilon+\gamma\epsilon$	0	...
v_3	$-\gamma^2\epsilon$	$-\gamma^2\epsilon$	$-\epsilon-\gamma\epsilon-\gamma^2\epsilon$	$\epsilon+\gamma\epsilon+\gamma^2\epsilon$	...
...	...	...	...	...	...

State 2: $0+\gamma(-\epsilon) = -2\gamma\epsilon + \gamma\epsilon$
State 3: $0+\gamma(-\epsilon-\gamma\epsilon) = -2(\gamma+\gamma^2)\epsilon + \gamma(\epsilon+\gamma\epsilon)$

$$v_{\pi_k}(k) = \sum_{t=0}^{\infty} \gamma^t \left(-2\frac{\gamma-\gamma^k}{1-\gamma}\epsilon\right) = -2\frac{\gamma-\gamma^k}{(1-\gamma)^2}\epsilon \overset{k\to\infty}{\longrightarrow} -\frac{2\gamma}{(1-\gamma)^2}\epsilon$$

Tightness of the bound for AVI

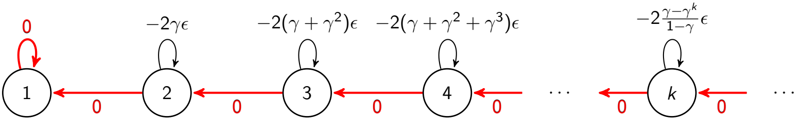


	1	2	3	4	...
v_0	0	0	0	0	...
v_1	$-\epsilon$	ϵ	0	0	...
v_2	$-\gamma\epsilon$	$-\epsilon-\gamma\epsilon$	$\epsilon+\gamma\epsilon$	0	...
v_3	$-\gamma^2\epsilon$	$-\gamma^2\epsilon$	$-\epsilon-\gamma\epsilon-\gamma^2\epsilon$	$\epsilon+\gamma\epsilon+\gamma^2\epsilon$	...
...	...	...	...	...	...

State 2: $0+\gamma(-\epsilon) = -2\gamma\epsilon+\gamma\epsilon$
State 3: $0+\gamma(-\epsilon-\gamma\epsilon) = -2(\gamma+\gamma^2)\epsilon+\gamma(\epsilon+\gamma\epsilon)$

$$v_{\pi_k}(k) = \sum_{t=0}^{\infty}\gamma^t\left(-2\frac{\gamma-\gamma^k}{1-\gamma}\epsilon\right) = -2\frac{\gamma-\gamma^k}{(1-\gamma)^2}\epsilon \xrightarrow{k\to\infty} -\frac{2\gamma}{(1-\gamma)^2}\epsilon$$

Tightness of the bound for AVI



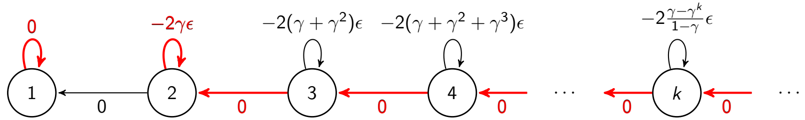
	1	2	3	4	...
v_0	0	0	0	0	...
v_1	$-\epsilon$	ϵ	0	0	...
v_2	$-\gamma\epsilon$	$-\epsilon-\gamma\epsilon$	$\epsilon+\gamma\epsilon$	0	...
v_3	$-\gamma^2\epsilon$	$-\gamma^2\epsilon$	$-\epsilon-\gamma\epsilon-\gamma^2\epsilon$	$\epsilon+\gamma\epsilon+\gamma^2\epsilon$	...
...	...	...	...	...	...

State 2: $0+\gamma(-\epsilon) = -2\gamma\epsilon + \gamma\epsilon$

State 3: $0+\gamma(-\epsilon-\gamma\epsilon) = -2(\gamma+\gamma^2)\epsilon + \gamma(\epsilon+\gamma\epsilon)$

$$v_{\pi_k}(k) = \sum_{t=0}^{\infty} \gamma^t \left(-2\frac{\gamma-\gamma^k}{1-\gamma}\epsilon\right) = -2\frac{\gamma-\gamma^k}{(1-\gamma)^2}\epsilon \overset{k\to\infty}{\longrightarrow} -\frac{2\gamma}{(1-\gamma)^2}\epsilon$$

Tightness of the bound for AVI



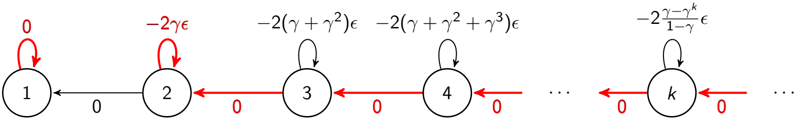
	1	2	3	4	...
v_0	0	0	0	0	...
v_1	$-\epsilon$	ϵ	0	0	...
v_2	$-\gamma\epsilon$	$-\epsilon-\gamma\epsilon$	$\epsilon+\gamma\epsilon$	0	...
v_3	$-\gamma^2\epsilon$	$-\gamma^2\epsilon$	$-\epsilon-\gamma\epsilon-\gamma^2\epsilon$	$\epsilon+\gamma\epsilon+\gamma^2\epsilon$	...
...	...	...	...	...	...

State 2: $0+\gamma(-\epsilon) = -2\gamma\epsilon+\gamma\epsilon$

State 3: $0+\gamma(-\epsilon-\gamma\epsilon) = -2(\gamma+\gamma^2)\epsilon+\gamma(\epsilon+\gamma\epsilon)$

$$v_{\pi_k}(k) = \sum_{t=0}^{\infty}\gamma^t\left(-2\frac{\gamma-\gamma^k}{1-\gamma}\epsilon\right) = -2\frac{\gamma-\gamma^k}{(1-\gamma)^2}\epsilon \xrightarrow{k\to\infty} -\frac{2\gamma}{(1-\gamma)^2}\epsilon$$

Tightness of the bound for AVI

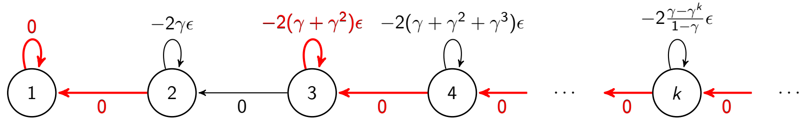


	1	2	3	4	...
v_0	0	0	0	0	...
v_1	$-\epsilon$	ϵ	0	0	...
v_2	$-\gamma\epsilon$	$-\epsilon-\gamma\epsilon$	$\epsilon+\gamma\epsilon$	0	...
v_3	$-\gamma^2\epsilon$	$-\gamma^2\epsilon$	$-\epsilon-\gamma\epsilon-\gamma^2\epsilon$	$\epsilon+\gamma\epsilon+\gamma^2\epsilon$	...
...	...	...	...	...	...

State 2: $0+\gamma(-\epsilon) = -2\gamma\epsilon+\gamma\epsilon$
State 3: $0+\gamma(-\epsilon-\gamma\epsilon) = -2(\gamma+\gamma^2)\epsilon+\gamma(\epsilon+\gamma\epsilon)$

$$v_{\pi_k}(k) = \sum_{t=0}^{\infty}\gamma^t\left(-2\frac{\gamma-\gamma^k}{1-\gamma}\epsilon\right) = -2\frac{\gamma-\gamma^k}{(1-\gamma)^2}\epsilon \xrightarrow{k\to\infty} -\frac{2\gamma}{(1-\gamma)^2}\epsilon$$

Tightness of the bound for AVI

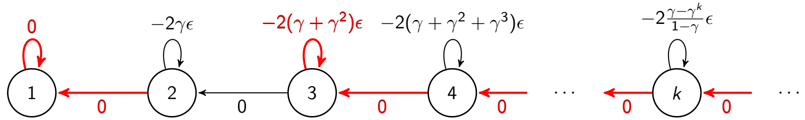


	1	2	3	4	...
v_0	0	0	0	0	...
v_1	$-\epsilon$	ϵ	0	0	...
v_2	$-\gamma\epsilon$	$-\epsilon-\gamma\epsilon$	$\epsilon+\gamma\epsilon$	0	...
v_3	$-\gamma^2\epsilon$	$-\gamma^2\epsilon$	$-\epsilon-\gamma\epsilon-\gamma^2\epsilon$	$\epsilon+\gamma\epsilon+\gamma^2\epsilon$	...
...	...	...	...	...	...

State 2: $0+\gamma(-\epsilon) = -2\gamma\epsilon + \gamma\epsilon$
State 3: $0+\gamma(-\epsilon-\gamma\epsilon) = -2(\gamma+\gamma^2)\epsilon + \gamma(\epsilon+\gamma\epsilon)$

$$v_{\pi_k}(k) = \sum_{t=0}^{\infty}\gamma^t\left(-2\frac{\gamma-\gamma^k}{1-\gamma}\epsilon\right) = -2\frac{\gamma-\gamma^k}{(1-\gamma)^2}\epsilon \xrightarrow{k\to\infty} -\frac{2\gamma}{(1-\gamma)^2}\epsilon$$

Tightness of the bound for AVI



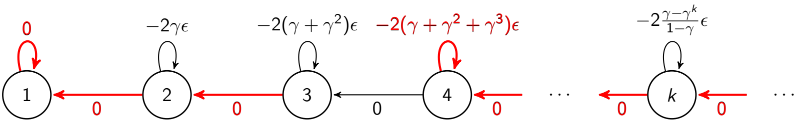
	1	2	3	4	...
v_0	0	0	0	0	...
v_1	$-\epsilon$	ϵ	0	0	...
v_2	$-\gamma\epsilon$	$-\epsilon-\gamma\epsilon$	$\epsilon+\gamma\epsilon$	0	...
v_3	$-\gamma^2\epsilon$	$-\gamma^2\epsilon$	$-\epsilon-\gamma\epsilon-\gamma^2\epsilon$	$\epsilon+\gamma\epsilon+\gamma^2\epsilon$	...
...	...	...	...	...	...

State 2: $0+\gamma(-\epsilon) = -2\gamma\epsilon+\gamma\epsilon$

State 3: $0+\gamma(-\epsilon-\gamma\epsilon) = -2(\gamma+\gamma^2)\epsilon+\gamma(\epsilon+\gamma\epsilon)$

$$v_{\pi_k}(k) = \sum_{t=0}^{\infty}\gamma^t\left(-2\frac{\gamma-\gamma^k}{1-\gamma}\epsilon\right) = -2\frac{\gamma-\gamma^k}{(1-\gamma)^2}\epsilon \xrightarrow{k\to\infty} -\frac{2\gamma}{(1-\gamma)^2}\epsilon$$

Tightness of the bound for AVI

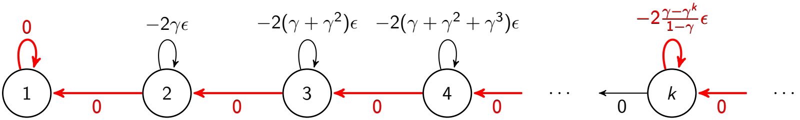


	1	2	3	4	...
v_0	0	0	0	0	...
v_1	$-\epsilon$	ϵ	0	0	...
v_2	$-\gamma\epsilon$	$-\epsilon-\gamma\epsilon$	$\epsilon+\gamma\epsilon$	0	...
v_3	$-\gamma^2\epsilon$	$-\gamma^2\epsilon$	$-\epsilon-\gamma\epsilon-\gamma^2\epsilon$	$\epsilon+\gamma\epsilon+\gamma^2\epsilon$	...
...	...	...	...	...	...

State 2: $0+\gamma(-\epsilon) = -2\gamma\epsilon + \gamma\epsilon$
State 3: $0+\gamma(-\epsilon-\gamma\epsilon) = -2(\gamma+\gamma^2)\epsilon + \gamma(\epsilon+\gamma\epsilon)$

$$v_{\pi_k}(k) = \sum_{t=0}^{\infty} \gamma^t \left(-2\frac{\gamma-\gamma^k}{1-\gamma}\epsilon\right) = -2\frac{\gamma-\gamma^k}{(1-\gamma)^2}\epsilon \overset{k\to\infty}{\longrightarrow} -\frac{2\gamma}{(1-\gamma)^2}\epsilon$$

Tightness of the bound for AVI



	1	2	3	4	...
v_0	0	0	0	0	...
v_1	$-\epsilon$	ϵ	0	0	...
v_2	$-\gamma\epsilon$	$-\epsilon-\gamma\epsilon$	$\epsilon+\gamma\epsilon$	0	...
v_3	$-\gamma^2\epsilon$	$-\gamma^2\epsilon$	$-\epsilon-\gamma\epsilon-\gamma^2\epsilon$	$\epsilon+\gamma\epsilon+\gamma^2\epsilon$	...
...	...	...	...	...	...

State 2: $0+\gamma(-\epsilon) = -2\gamma\epsilon+\gamma\epsilon$
State 3: $0+\gamma(-\epsilon-\gamma\epsilon) = -2(\gamma+\gamma^2)\epsilon+\gamma(\epsilon+\gamma\epsilon)$

$$v_{\pi_k}(k) = \sum_{t=0}^{\infty}\gamma^t\left(-2\frac{\gamma-\gamma^k}{1-\gamma}\epsilon\right) = -2\frac{\gamma-\gamma^k}{(1-\gamma)^2}\epsilon \xrightarrow{k\to\infty} -\frac{2\gamma}{(1-\gamma)^2}\epsilon$$

Non-Stationary Value Iteration

AVI generates a sequence of values/policies ($\pi_{i+1} \in \mathcal{G} v_i$)

$$\begin{array}{cccccccc} v_0 & v_1 & v_2 & \dots & v_{k-\ell} & \dots & v_{k-2} & v_{k-1} \\ \pi_1 & \pi_2 & \pi_3 & \dots & \pi_{k-\ell+1} & \dots & \pi_{k-1} & \pi_k \end{array}$$

Return the following periodic non-stationary policy

$$\pi_{k,\ell} = \underbrace{\pi_k \ \pi_{k-1} \ \cdots \ \pi_{k-\ell+1}}_{\ell \text{ last policies}} \ \underbrace{\pi_k \ \pi_{k-1} \ \cdots \pi_{k-\ell+1}}_{\ell \text{ last policies}} \cdots$$

Theorem (Scherrer and Lesner, 2012)

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. For all ℓ, the loss due to running the non-stationary policy $\pi_{k,\ell}$ instead of the optimal policy π_* satisfies:

$$\limsup_{k\to\infty} \|v_* - v_{\pi_{k,\ell}}\|_\infty \leq \frac{2\gamma}{(1-\gamma^\ell)(1-\gamma)}\epsilon.$$

Corollary: looping on all policies from the start leads to

$$\limsup_{k\to\infty} \|v_* - v_{\pi_{k,k}}\|_\infty \leq \frac{2\gamma}{1-\gamma}\epsilon$$

Non-Stationary Value Iteration

AVI generates a sequence of values/policies ($\pi_{i+1} \in \mathcal{G} v_i$)

$$\begin{array}{cccccccc} v_0 & v_1 & v_2 & \dots & v_{k-\ell} & \dots & v_{k-2} & v_{k-1} \\ \pi_1 & \pi_2 & \pi_3 & \dots & \pi_{k-\ell+1} & \dots & \pi_{k-1} & \pi_k \end{array}$$

Return the following periodic non-stationary policy

$$\pi_{k,\ell} = \underbrace{\pi_k \ \pi_{k-1} \ \cdots \ \pi_{k-\ell+1}}_{\ell \text{ last policies}} \ \underbrace{\pi_k \ \pi_{k-1} \ \cdots \pi_{k-\ell+1}}_{\ell \text{ last policies}} \cdots$$

Theorem (Scherrer and Lesner, 2012)

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. For all ℓ, the loss due to running the non-stationary policy $\pi_{k,\ell}$ instead of the optimal policy π_* satisfies:

$$\limsup_{k \to \infty} \|v_* - v_{\pi_{k,\ell}}\|_\infty \leq \frac{2\gamma}{(1-\gamma^\ell)(1-\gamma)}\epsilon.$$

Corollary: looping on all policies from the start leads to

$$\limsup_{k \to \infty} \|v_* - v_{\pi_{k,k}}\|_\infty \leq \frac{2\gamma}{1-\gamma}\epsilon$$

Non-Stationary Value Iteration

AVI generates a sequence of values/policies ($\pi_{i+1} \in \mathcal{G} v_i$)

$$\begin{array}{cccccccc} v_0 & v_1 & v_2 & \dots & v_{k-\ell} & \dots & v_{k-2} & v_{k-1} \\ \pi_1 & \pi_2 & \pi_3 & \dots & \pi_{k-\ell+1} & \dots & \pi_{k-1} & \pi_k \end{array}$$

Return the following periodic non-stationary policy

$$\pi_{k,\ell} = \underbrace{\pi_k \ \pi_{k-1} \ \cdots \ \pi_{k-\ell+1}}_{\ell \text{ last policies}} \ \underbrace{\pi_k \ \pi_{k-1} \ \cdots \pi_{k-\ell+1}}_{\ell \text{ last policies}} \cdots$$

Theorem (Scherrer and Lesner, 2012)

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. For all ℓ, the loss due to running the non-stationary policy $\pi_{k,\ell}$ instead of the optimal policy π_* satisfies:

$$\limsup_{k \to \infty} \|v_* - v_{\pi_{k,\ell}}\|_\infty \leq \frac{2\gamma}{(1-\gamma^\ell)(1-\gamma)} \epsilon.$$

Corollary: looping on all policies from the start leads to

$$\limsup_{k \to \infty} \|v_* - v_{\pi_{k,k}}\|_\infty \leq \frac{2\gamma}{1-\gamma} \epsilon$$

Non-Stationary Value Iteration

AVI generates a sequence of values/policies ($\pi_{i+1} \in \mathcal{G} v_i$)

$$\begin{array}{cccccccc} v_0 & v_1 & v_2 & \dots & v_{k-\ell} & \dots & v_{k-2} & v_{k-1} \\ \pi_1 & \pi_2 & \pi_3 & \dots & \pi_{k-\ell+1} & \dots & \pi_{k-1} & \pi_k \end{array}$$

Return the following periodic non-stationary policy

$$\pi_{k,\ell} = \underbrace{\pi_k \ \pi_{k-1} \ \cdots \ \pi_{k-\ell+1}}_{\ell \text{ last policies}} \ \underbrace{\pi_k \ \pi_{k-1} \ \cdots \pi_{k-\ell+1}}_{\ell \text{ last policies}} \cdots$$

Theorem (Scherrer and Lesner, 2012)

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. For all ℓ, the loss due to running the non-stationary policy $\pi_{k,\ell}$ instead of the optimal policy π_* satisfies:

$$\limsup_{k \to \infty} \|v_* - v_{\pi_{k,\ell}}\|_\infty \leq \frac{2\gamma}{(1-\gamma^\ell)(1-\gamma)} \epsilon.$$

Corollary: looping on all policies from the start leads to

$$\limsup_{k \to \infty} \|v_* - v_{\pi_{k,k}}\|_\infty \leq \frac{2\gamma}{1-\gamma} \epsilon$$

Non-Stationary Value Iteration

AVI generates a sequence of values/policies ($\pi_{i+1} \in \mathcal{G} v_i$)

$$\begin{array}{cccccccc} v_0 & v_1 & v_2 & \dots & v_{k-\ell} & \dots & v_{k-2} & v_{k-1} \\ \pi_1 & \pi_2 & \pi_3 & \dots & \pi_{k-\ell+1} & \dots & \pi_{k-1} & \pi_k \end{array}$$

Return the following periodic non-stationary policy

$$\pi_{k,\ell} = \underbrace{\pi_k \ \pi_{k-1} \ \cdots \ \pi_{k-\ell+1}}_{\ell \text{ last policies}} \ \underbrace{\pi_k \ \pi_{k-1} \ \cdots \pi_{k-\ell+1}}_{\ell \text{ last policies}} \cdots$$

Theorem (Scherrer and Lesner, 2012)

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. For all ℓ, the loss due to running the non-stationary policy $\pi_{k,\ell}$ instead of the optimal policy π_* satisfies:

$$\limsup_{k \to \infty} \|v_* - v_{\pi_{k,\ell}}\|_\infty \leq \frac{2\gamma}{(1-\gamma^\ell)(1-\gamma)}\epsilon.$$

Corollary: looping on <u>all</u> policies from the start leads to

$$\limsup_{k \to \infty} \|v_* - v_{\pi_{k,k}}\|_\infty \leq \frac{2\gamma}{1-\gamma}\epsilon$$

Proof idea (NSVI)

By "usual" contraction arguments, v_k is close to v_*:

$$\begin{aligned}\|v_* - v_k\|_\infty &= \|v_* - T v_{k-1} - \epsilon_k\|_\infty \\ &\leq \|T v_* - T v_{k-1}\|_\infty + \epsilon \\ &\leq \gamma \|v_* - v_{k-1}\|_\infty + \epsilon \\ &\overset{k \gg 1}{\lesssim} \quad \frac{\epsilon}{1-\gamma}.\end{aligned}$$

For sufficiently big ℓ, v_k is a rather good approximation of the value $v_{\pi_{k,\ell}}$ (whereas v_k is in general a poor approximation of v_{π_k}):

$$\|v_k - v_{\pi_{k,\ell}}\|_\infty \leq \gamma^\ell \|v_{k-\ell} - v_{\pi_{k,\ell}}\|_\infty + \frac{1-\gamma^\ell}{1-\gamma}\epsilon \overset{\ell \gg 1}{\lesssim} \quad \frac{\epsilon}{1-\gamma}$$

Then, the loss is bounded using the triangle inequality:

$$\|v_* - v_{\pi_{k,\ell}}\|_\infty \leq \|v_* - v_k\|_\infty + \|v_k - v_{\pi_{k,\ell}}\|_\infty \sim \frac{2\epsilon}{1-\gamma}$$

Proof idea (NSVI)

By "usual" contraction arguments, v_k is close to v_*:

$$
\begin{aligned}
\|v_* - v_k\|_\infty &= \|v_* - T v_{k-1} - \epsilon_k\|_\infty \\
&\leq \|T v_* - T v_{k-1}\|_\infty + \epsilon \\
&\leq \gamma \|v_* - v_{k-1}\|_\infty + \epsilon \\
&\overset{k \gg 1}{\lesssim} \frac{\epsilon}{1-\gamma}.
\end{aligned}
$$

For sufficiently big ℓ, v_k is a rather good approximation of the value $v_{\pi_{k,\ell}}$ (whereas v_k is in general a poor approximation of v_{π_k}):

$$
\|v_k - v_{\pi_{k,\ell}}\|_\infty \leq \gamma^\ell \|v_{k-\ell} - v_{\pi_{k,\ell}}\|_\infty + \frac{1-\gamma^\ell}{1-\gamma}\epsilon \overset{\ell \gg 1}{\lesssim} \frac{\epsilon}{1-\gamma}
$$

Then, the loss is bounded using the triangle inequality:

$$
\|v_* - v_{\pi_{k,\ell}}\|_\infty \leq \|v_* - v_k\|_\infty + \|v_k - v_{\pi_{k,\ell}}\|_\infty \sim \frac{2\epsilon}{1-\gamma}
$$

Proof idea (NSVI)

By "usual" contraction arguments, v_k is close to v_*:

$$\begin{aligned}\|v_* - v_k\|_\infty &= \|v_* - T v_{k-1} - \epsilon_k\|_\infty \\ &\leq \|T v_* - T v_{k-1}\|_\infty + \epsilon \\ &\leq \gamma \|v_* - v_{k-1}\|_\infty + \epsilon \\ &\overset{k \gg 1}{\lesssim} \frac{\epsilon}{1-\gamma}.\end{aligned}$$

For sufficiently big ℓ, v_k is a rather good approximation of the value $v_{\pi_{k,\ell}}$ (whereas v_k is in general a poor approximation of v_{π_k}):

$$\|v_k - v_{\pi_{k,\ell}}\|_\infty \leq \gamma^\ell \|v_{k-\ell} - v_{\pi_{k,\ell}}\|_\infty + \frac{1-\gamma^\ell}{1-\gamma}\epsilon \overset{\ell \gg 1}{\lesssim} \frac{\epsilon}{1-\gamma}$$

Then, the loss is bounded using the triangle inequality:

$$\|v_* - v_{\pi_{k,\ell}}\|_\infty \leq \|v_* - v_k\|_\infty + \|v_k - v_{\pi_{k,\ell}}\|_\infty \sim \frac{2\epsilon}{1-\gamma}$$

API with a non-stationary policy of period ℓ

$\pi_{k+1} \leftarrow \mathcal{G} v_k$
$v_{k+1} \leftarrow v_{\pi_{k+1,\ell}} + \epsilon_k$ (by solving $v_{k+1} \simeq T_{\pi_{k+1,\ell}} v_{k+1}$)

where $\pi_{\ell,\ell} = \pi_\ell \; \pi_{\ell-1} \; \dots \; \pi_1 \; \pi_\ell \; \pi_{\ell-1} \; \dots \; \pi_1 \; \dots$
with arbitrary $\pi_1, \pi_2, \dots \pi_\ell$ and

$$\forall v, \quad T_{\pi_{k,\ell}} v = T_{\pi_k} T_{\pi_{k-1}} \dots T_{\pi_{k-\ell+1}} v.$$

Theorem (Scherrer and Lesner, 2012)

Assume $\|\epsilon_k\|_\infty \le \epsilon$. The loss due to running the non-stationary policy $\pi_{k,\ell}$ instead of the optimal policy π_* satisfies:

$$\limsup_{k \to \infty} \|v_* - v_{\pi_{k,\ell}}\|_\infty \le \frac{2\gamma}{(1-\gamma^\ell)(1-\gamma)} \epsilon.$$

Non-Stationary PI

API with a non-stationary policy of period ℓ

$\pi_{k+1} \leftarrow \mathcal{G} v_k$
$v_{k+1} \leftarrow v_{\pi_{k+1,\ell}} + \epsilon_k$ (by solving $v_{k+1} \simeq T_{\pi_{k+1,\ell}} v_{k+1}$)

where $\pi_{\ell,\ell} = \pi_\ell \; \pi_{\ell-1} \; \dots \; \pi_1 \; \pi_\ell \; \pi_{\ell-1} \; \dots \; \pi_1 \; \dots$
with arbitrary $\pi_1, \pi_2, \dots \pi_\ell$ and

$$\forall v, \quad T_{\pi_{k,\ell}} v = T_{\pi_k} T_{\pi_{k-1}} \dots T_{\pi_{k-\ell+1}} v.$$

Theorem (Scherrer and Lesner, 2012)

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. The loss due to running the non-stationary policy $\pi_{k,\ell}$ instead of the optimal policy π_* satisfies:

$$\limsup_{k \to \infty} \|v_* - v_{\pi_{k,\ell}}\|_\infty \leq \frac{2\gamma}{(1-\gamma^\ell)(1-\gamma)} \epsilon.$$

Proof idea (NSPI): Approximate Monotonicity

At each iteration, one moves from

$$
\begin{aligned}
\pi_{k,\ell} &= \pi_k \ \dots \ \pi_{k+2-\ell} \ \pi_{k-\ell+1} \ \pi_k \ \dots \ \pi_{k-\ell+2} \pi_{k-\ell+1} \ \dots \\
\text{to } \pi_{k+1,\ell} &= \pi_{k+1} \ \pi_k \ \dots \ \pi_{k+2-\ell} \ \pi_{k+1} \ \pi_k \ \dots \ \pi_{k-\ell+2} \dots
\end{aligned}
$$

The new policy $\pi_{k+1,\ell}$ cannot be much worse than $\pi'_{k,\ell}$ (a "1-step rotation" of $\pi_{k,\ell}$)

$$\pi'_{k,\ell} = \pi_{k-\ell+1} \ \pi_k \ \dots \ \pi_{k+2-\ell} \ \pi_{k-\ell+1} \ \pi_k \ \dots \ \pi_{k-\ell+2} \dots$$

in the precise following sense:

$$v_{\pi_{k+1,\ell}} \geq v_{\pi'_{k,\ell}} - \frac{2\gamma}{1-\gamma^\ell}\epsilon.$$

Non Stationary MPI

NS Value Iteration

$\pi_{k+1} \leftarrow \mathcal{G} v_k$
$v_{k+1} \leftarrow T_{\pi_{k+1}} v_k + \epsilon_k$

NS Policy Iteration

$\pi_{k+1} \leftarrow \mathcal{G} v_k$
$v_{k+1} \leftarrow (T_{\pi_{k+1,\ell}})^{\infty} T_{\pi_{k+1}} v_k + \epsilon_k$

NS Modified Policy Iteration

$\pi_{k+1} \leftarrow \mathcal{G} v_k$
$v_{k+1} \leftarrow (T_{\pi_{k+1,\ell}})^{m} T_{\pi_{k+1}} v_k + \epsilon_k \qquad (0 \leq m \leq \infty)$

Theorem (Lesner and Scherrer, 2014)

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. The loss due to running policy π_k instead of the optimal policy π_* satisfies

$$\limsup_{k \to \infty} \|v_* - v_{\pi_k}\|_\infty \leq \frac{2\gamma}{(1-\gamma^\ell)(1-\gamma)} \epsilon.$$

These are algorithms for ℓ-periodic MDPs (Broken stationarity)

Non Stationary MPI

NS Value Iteration

$\pi_{k+1} \leftarrow \mathcal{G} v_k$
$v_{k+1} \leftarrow T_{\pi_{k+1}} v_k + \epsilon_k$

NS Policy Iteration

$\pi_{k+1} \leftarrow \mathcal{G} v_k$
$v_{k+1} \leftarrow (T_{\pi_{k+1,\ell}})^{\infty} T_{\pi_{k+1}} v_k + \epsilon_k$

NS Modified Policy Iteration

$\pi_{k+1} \leftarrow \mathcal{G} v_k$
$v_{k+1} \leftarrow (T_{\pi_{k+1,\ell}})^{m} T_{\pi_{k+1}} v_k + \epsilon_k \qquad (0 \leq m \leq \infty)$

Theorem (Lesner and Scherrer, 2014)

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. The loss due to running policy π_k instead of the optimal policy π_* satisfies

$$\limsup_{k\to\infty} \|v_* - v_{\pi_k}\|_\infty \leq \frac{2\gamma}{(1-\gamma^\ell)(1-\gamma)}\epsilon.$$

These are algorithms for ℓ-periodic MDPs (Broken stationarity)

Non Stationary MPI

NS Value Iteration

$\pi_{k+1} \leftarrow \mathcal{G} v_k$
$v_{k+1} \leftarrow T_{\pi_{k+1}} v_k + \epsilon_k$

NS Policy Iteration

$\pi_{k+1} \leftarrow \mathcal{G} v_k$
$v_{k+1} \leftarrow (T_{\pi_{k+1,\ell}})^{\infty} T_{\pi_{k+1}} v_k + \epsilon_k$

NS Modified Policy Iteration

$\pi_{k+1} \leftarrow \mathcal{G} v_k$
$v_{k+1} \leftarrow (T_{\pi_{k+1,\ell}})^{m} T_{\pi_{k+1}} v_k + \epsilon_k \qquad (0 \leq m \leq \infty)$

Theorem (Lesner and Scherrer, 2014)

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. The loss due to running policy π_k instead of the optimal policy π_* satisfies

$$\limsup_{k \to \infty} \|v_* - v_{\pi_k}\|_\infty \leq \frac{2\gamma}{(1-\gamma^\ell)(1-\gamma)} \epsilon.$$

These are algorithms for ℓ-periodic MDPs (Broken stationarity)

Non Stationary MPI

NS Value Iteration

$\pi_{k+1} \leftarrow \mathcal{G} v_k$
$v_{k+1} \leftarrow T_{\pi_{k+1}} v_k + \epsilon_k$

NS Policy Iteration

$\pi_{k+1} \leftarrow \mathcal{G} v_k$
$v_{k+1} \leftarrow (T_{\pi_{k+1,\ell}})^\infty T_{\pi_{k+1}} v_k + \epsilon_k$

NS Modified Policy Iteration

$\pi_{k+1} \leftarrow \mathcal{G} v_k$
$v_{k+1} \leftarrow (T_{\pi_{k+1,\ell}})^m T_{\pi_{k+1}} v_k + \epsilon_k \qquad (0 \leq m \leq \infty)$

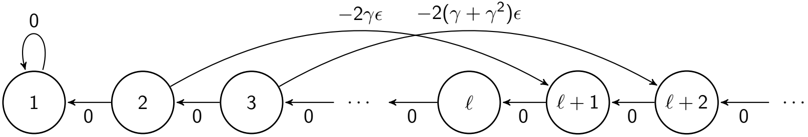
Theorem (Lesner and Scherrer, 2014)

Assume $\|\epsilon_k\|_\infty \leq \epsilon$. The loss due to running policy π_k instead of the optimal policy π_* satisfies

$$\limsup_{k \to \infty} \|v_* - v_{\pi_k}\|_\infty \leq \frac{2\gamma}{(1-\gamma^\ell)(1-\gamma)}\epsilon.$$

These are algorithms for ℓ-periodic MDPs (Broken stationarity)

Tightness of the bound (Lesner and Scherrer, 2014)



For any m and ℓ, NSMPI generates a sequence of policies $(\pi_k)_{k\geq 1}$ such that π_k acts optimally except in state k.
Thus, $\pi_{k,\ell} = \pi_k \pi_{k-1} \dots \pi_{k-\ell+1}$ gets stuck in the loop

$$k, \quad k+\ell-1, \quad k+\ell-2, \quad k+1, \quad k, \quad \dots$$

and therefore

$$v_{\pi_{k,\ell}}(k) = -\frac{2\gamma - \gamma^k}{(1-\gamma)(1-\gamma^\ell)}\epsilon.$$

Empirical Illustration

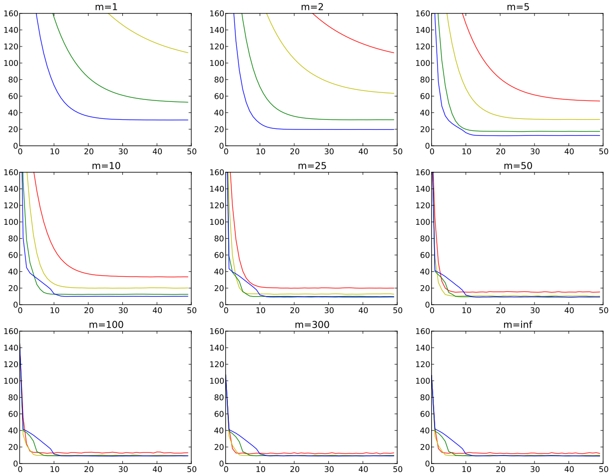


Figure: Average error of policy $\pi_{k,\ell}$ per iteration k of NS-AMPI, for $\ell = 1$, $\ell = 2$, $\ell = 5$ and $\ell = 10$.

Concentrability coefficients

- The analysis of Approximate DP algorithms is done wrt $\| \cdot \|_\infty$
- The analysis of the error ϵ_k is done wrt $\| \cdot \|_{2,\mu}$
- The performance bounds are in fact:

$$\limsup_{k \to \infty} \| v_* - v_{\pi_{k,\ell}} \|_{2,\nu} \leq \frac{2\sqrt{C}\gamma}{(1-\gamma^\ell)(1-\gamma)} \max_k \|\epsilon_k\|_{2,\mu}.$$

where

$$C = (1-\gamma)(1-\gamma) \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} \gamma^{i+j\ell} c(i + j\ell + k)$$

with $c(i) = \max_{\pi_1,\pi_2,\ldots,\pi_i} \left\| \frac{\mu P_{\pi_1} P_{\pi_2} \ldots P_{\pi_i}}{\nu} \right\|_{2,\mu}$.

The Approximate Greedy Operator

(Exact) Policy Iteration

$\pi_{k+1} \leftarrow \mathcal{G} v_{\pi_k}$ (where $v_{\pi_k} = T_{\pi_k} v_{\pi_k}$)

- π is (ϵ, ν)-**approximately greedy with respect to** v, written $\pi = \mathcal{G}_\epsilon(\nu, v)$, iff

$$\mathbb{E}_{x\sim\nu} \{[Tv](x) - [T_\pi v](x)\} \leq \epsilon.$$

Can be implemented through

- $l_{1,\nu}/l_\infty$-regression of the Q-function (Kakade and Langford, 2002)
- $l_{2,\nu}$ fixed point LSTD approach
- $l_{1,\nu}$ cost-sensitive classification

The Approximate Greedy Operator

(Exact) Policy Iteration

$\pi_{k+1} \leftarrow \mathcal{G} v_{\pi_k}$ (where $v_{\pi_k} = T_{\pi_k} v_{\pi_k}$)

- π is (ϵ, ν)-**approximately greedy with respect to** v, written $\pi = \mathcal{G}_\epsilon(\nu, v)$, iff

$$\mathbb{E}_{x \sim \nu} \{[T v](x) - [T_\pi v](x)\} \leq \epsilon.$$

Can be implemented through

- $l_{1,\nu}/l_\infty$-regression of the Q-function (Kakade and Langford, 2002)
- $l_{2,\nu}$ fixed point LSTD approach
- $l_{1,\nu}$ cost-sensitive classification

The Approximate Greedy Operator

(Exact) Policy Iteration

$\pi_{k+1} \leftarrow \mathcal{G} v_{\pi_k}$ (where $v_{\pi_k} = T_{\pi_k} v_{\pi_k}$)

- π is (ϵ, ν)-**approximately greedy with respect to** v, written $\pi = \mathcal{G}_\epsilon(\nu, v)$, iff

$$\mathbb{E}_{x \sim \nu} \{[Tv](x) - [T_\pi v](x)\} \leq \epsilon.$$

Can be implemented through

- $l_{1,\nu}/l_\infty$-regression of the Q-function (Kakade and Langford, 2002)
- $l_{2,\nu}$ fixed point LSTD approach
- $l_{1,\nu}$ cost-sensitive classification

Approximate/Conservative Policy Iteration

API

$\pi_{k+1} \leftarrow \mathcal{G}_{\epsilon_k}(\nu, v_{\pi_k})$

CPI/CPI+/CPI(α) (Kakade and Langford, 2002)

$\pi_{k+1} \leftarrow (1-\alpha_{k+1})\pi_k + \alpha_{k+1}\mathcal{G}_{\epsilon_k}(d_{\nu,\pi_k}, v_{\pi_k})$

- $d_{\nu,\pi_k}(x') = (1-\gamma)\mathbb{E}_{x_0\sim\nu}\left[\,\sum_{t=0}^{\infty}\gamma^t \mathbb{1}_{x_t=x'} \mid a_t = \pi_k(x_t)\,\right]$
- If the α_k are sufficiently small, then $(E_{x\sim\nu}[v_{\pi_k}(x)])_k$ is non-decreasing
- In practice: set α_k by line search (CPI+) or to a small value (e.g. $\alpha = 0.1$) (CPI(α))

API(α) (Lagoudakis, 2003)

$\pi_{k+1} \leftarrow (1-\alpha)\pi_k + \alpha\mathcal{G}_{\epsilon_k}(\nu, v_{\pi_k})$

Approximate/Conservative Policy Iteration

API

$\pi_{k+1} \leftarrow \mathcal{G}_{\epsilon_k}(\nu, v_{\pi_k})$

CPI/CPI+/CPI(α) (Kakade and Langford, 2002)

$\pi_{k+1} \leftarrow (1-\alpha_{k+1})\pi_k + \alpha_{k+1}\mathcal{G}_{\epsilon_k}(d_{\nu,\pi_k}, v_{\pi_k})$

- $d_{\nu,\pi_k}(x') = (1-\gamma)\mathbb{E}_{x_0\sim\nu}\left[\,\sum_{t=0}^{\infty}\gamma^t \mathbb{1}_{x_t=x'} \mid a_t = \pi_k(x_t)\,\right]$
- If the α_k are sufficiently small, then $(E_{x\sim\nu}[v_{\pi_k}(x)])_k$ is non-decreasing
- In practice: set α_k by line search (CPI+) or to a small value (e.g. $\alpha = 0.1$) (CPI(α))

API(α) (Lagoudakis, 2003)

$\pi_{k+1} \leftarrow (1-\alpha)\pi_k + \alpha\mathcal{G}_{\epsilon_k}(\nu, v_{\pi_k})$

Approximate/Conservative Policy Iteration

API

$\pi_{k+1} \leftarrow \mathcal{G}_{\epsilon_k}(\nu, v_{\pi_k})$

CPI/CPI+/CPI(α) (Kakade and Langford, 2002)

$\pi_{k+1} \leftarrow (1 - \alpha_{k+1})\pi_k + \alpha_{k+1}\mathcal{G}_{\epsilon_k}(d_{\nu,\pi_k}, v_{\pi_k})$

- $d_{\nu,\pi_k}(x') = (1-\gamma)\mathbb{E}_{x_0\sim\nu}\left[\sum_{t=0}^{\infty}\gamma^t \mathbb{1}_{x_t=x'} \mid a_t = \pi_k(x_t)\right]$
- If the α_k are sufficiently small, then $(E_{x\sim\nu}[v_{\pi_k}(x)])_k$ is non-decreasing
- In practice: set α_k by line search (CPI+) or to a small value (e.g. $\alpha = 0.1$) (CPI(α))

API(α) (Lagoudakis, 2003)

$\pi_{k+1} \leftarrow (1-\alpha)\pi_k + \alpha\mathcal{G}_{\epsilon_k}(\nu, v_{\pi_k})$

Policy Search by Dynamic Programming for infinite-horizon problems

PSDP$_\infty$ (based on PSDP, Bagnell et al., 2003)

$\pi_{k+1} \leftarrow \mathcal{G}_{\epsilon_k}(\nu, v_{\sigma_k})$

- $\sigma_k = \pi_k \ \pi_{k-1} \ \ldots \ \pi_1$ is a finite $(k-)$horizon policy $\quad (\sigma_0 = \varnothing)$
- $v_{\sigma_k} = T_{\pi_k} T_{\pi_{k-1}} \ldots T_{\pi_1} 0, \quad (v_{\sigma_0} = 0)$
- **Output**: Turn the finite-horizon policy σ_k to the following infinite-horizon policy:

$$\sigma * \ = \ \pi_1 * \qquad (*\text{=anything})$$

For CPI, and PSDP$_\infty$, the memory used grows linearly with the number of iterations!

Policy Search by Dynamic Programming for infinite-horizon problems

PSDP$_\infty$ (based on PSDP, Bagnell et al., 2003)

$\pi_{k+1} \leftarrow \mathcal{G}_{\epsilon_k}(\nu, v_{\sigma_k})$

- $\sigma_k = \pi_k \ \pi_{k-1} \ \ldots \ \pi_1$ is a finite ($k-$)horizon policy $\quad (\sigma_0 = \varnothing)$
- $v_{\sigma_k} = T_{\pi_k} T_{\pi_{k-1}} \ldots T_{\pi_1} 0, \quad (v_{\sigma_0} = 0)$
- **Output**: Turn the finite-horizon policy σ_k to the following infinite-horizon policy:

$$\sigma_1 * \ = \ \pi_1 * \qquad (*\text{=anything})$$

For CPI, and PSDP$_\infty$, the memory used grows linearly with the number of iterations!

Policy Search by Dynamic Programming for infinite-horizon problems

PSDP$_\infty$ (based on PSDP, Bagnell et al., 2003)

$\pi_{k+1} \leftarrow \mathcal{G}_{\epsilon_k}(\nu, v_{\sigma_k})$

- $\sigma_k = \pi_k \; \pi_{k-1} \; \ldots \; \pi_1$ is a finite $(k-)$horizon policy $\quad (\sigma_0 = \emptyset)$
- $v_{\sigma_k} = T_{\pi_k} T_{\pi_{k-1}} \ldots T_{\pi_1} 0, \quad (v_{\sigma_0} = 0)$
- **Output**: Turn the finite-horizon policy σ_k to the following infinite-horizon policy:

$$\sigma_2 * \; = \; \pi_2 \; \pi_1 * \qquad\qquad (*\text{=anything})$$

For CPI, and PSDP$_\infty$, the memory used grows linearly with the number of iterations!

Policy Search by Dynamic Programming for infinite-horizon problems

PSDP$_\infty$ (based on PSDP, Bagnell et al., 2003)

$$\pi_{k+1} \leftarrow \mathcal{G}_{\epsilon_k}(\nu, v_{\sigma_k})$$

- $\sigma_k = \pi_k \; \pi_{k-1} \; \ldots \; \pi_1$ is a finite $(k-)$horizon policy $\quad (\sigma_0 = \emptyset)$
- $v_{\sigma_k} = T_{\pi_k} T_{\pi_{k-1}} \ldots T_{\pi_1} 0, \quad (v_{\sigma_0} = 0)$
- **Output**: Turn the finite-horizon policy σ_k to the following infinite-horizon policy:

$$\sigma_3 * \; = \; \pi_3 \; \pi_2 \; \pi_1 * \qquad\qquad (*\text{=anything})$$

For CPI, and PSDP$_\infty$, the memory used grows linearly with the number of iterations!

Policy Search by Dynamic Programming for infinite-horizon problems

PSDP$_\infty$ (based on PSDP, Bagnell et al., 2003)

$\pi_{k+1} \leftarrow \mathcal{G}_{\epsilon_k}(\nu, v_{\sigma_k})$

- $\sigma_k = \pi_k \; \pi_{k-1} \; \ldots \; \pi_1$ is a finite ($k-$)horizon policy $\quad (\sigma_0 = \emptyset)$
- $v_{\sigma_k} = T_{\pi_k} T_{\pi_{k-1}} \ldots T_{\pi_1} 0, \quad (v_{\sigma_0} = 0)$
- **Output**: Turn the finite-horizon policy σ_k to the following infinite-horizon policy:

$$\sigma_4 * \; = \; \pi_4 \; \pi_3 \; \pi_2 \; \pi_1 * \qquad (*\text{=anything})$$

For CPI, and PSDP$_\infty$, the memory used grows linearly with the number of iterations!

Policy Search by Dynamic Programming for infinite-horizon problems

PSDP$_\infty$ (based on PSDP, Bagnell et al., 2003)

$\pi_{k+1} \leftarrow \mathcal{G}_{\epsilon_k}(\nu, v_{\sigma_k})$

- $\sigma_k = \pi_k \; \pi_{k-1} \; \ldots \; \pi_1$ is a finite $(k-)$horizon policy $\quad (\sigma_0 = \emptyset)$
- $v_{\sigma_k} = T_{\pi_k} T_{\pi_{k-1}} \ldots T_{\pi_1} 0, \quad (v_{\sigma_0} = 0)$
- **Output**: Turn the finite-horizon policy σ_k to the following infinite-horizon policy:

$$\sigma_k * \; = \; \pi_k \; \pi_{k-1} \; \ldots \; \pi_2 \; \pi_1 * \qquad (*\text{=anything})$$

For CPI, and PSDP$_\infty$, the memory used grows linearly with the number of iterations!

Policy Search by Dynamic Programming for infinite-horizon problems

PSDP$_\infty$ (based on PSDP, Bagnell et al., 2003)

$\pi_{k+1} \leftarrow \mathcal{G}_{\epsilon_k}(\nu, v_{\sigma_k})$

- $\sigma_k = \pi_k \; \pi_{k-1} \; \ldots \; \pi_1$ is a finite $(k-)$horizon policy $\quad (\sigma_0 = \emptyset)$
- $v_{\sigma_k} = T_{\pi_k} T_{\pi_{k-1}} \ldots T_{\pi_1} 0, \quad (v_{\sigma_0} = 0)$
- **Output**: Turn the finite-horizon policy σ_k to the following infinite-horizon policy:

$$\sigma_k * \; = \; \pi_k \; \pi_{k-1} \; \ldots \; \pi_2 \; \pi_1 * \qquad (*\text{=anything})$$

For CPI, and PSDP$_\infty$, the memory used grows linearly with the number of iterations!

Non-Stationary Policy Iteration

NSPI(ℓ)

$\pi_{k+1} \leftarrow \mathcal{G}_{\epsilon_k}(\nu, v_{(\sigma_k^\ell)^\infty})$

- $(\sigma_k^\ell)^\infty = (\pi_k\ \pi_{k-1}\ \ldots\ \pi_{k-\ell+1})^\infty$ is an infinite-horizon (ℓ-)periodic policy
- $v_{(\sigma_k^\ell)^\infty} = T_{\pi_k} T_{\pi_{k-1}} \ldots T_{\pi_{k-\ell+1}} v_{(\sigma_k^\ell)^\infty}$
- **Output:**

$$
\begin{aligned}
(\sigma_0^\ell)^\infty &= (\pi_0\ \pi_{-1}\ \ldots\ \pi_{-\ell+2}\ )^\infty \\
(\sigma_1^\ell)^\infty &= (\ \pi_0\ \ldots\ \pi_{-\ell+3}\ )^\infty \\
(\sigma_2^\ell)^\infty &= (\ \pi_1\ \ldots\ \pi_{-\ell+4}\ \pi_{-\ell+3})^\infty \\
&\ \ \vdots \qquad\qquad\qquad \vdots \\
(\sigma_k^\ell)^\infty &= (\pi_k\ \pi_{k-1}\ \ldots\ \pi_{k-\ell+2}\ \pi_{k-\ell+1})^\infty
\end{aligned}
$$

Bridge between API=NSPI(1) and PSDP$_\infty$≃NSPI(∞).

Non-Stationary Policy Iteration

NSPI(ℓ)

$\pi_{k+1} \leftarrow \mathcal{G}_{\epsilon_k}(\nu, v_{(\sigma_k^\ell)^\infty})$

- $(\sigma_k^\ell)^\infty = (\pi_k \; \pi_{k-1} \; \ldots \; \pi_{k-\ell+1})^\infty$ is an infinite-horizon (ℓ-)periodic policy
- $v_{(\sigma_k^\ell)^\infty} = T_{\pi_k} T_{\pi_{k-1}} \ldots T_{\pi_{k-\ell+1}} v_{(\sigma_k^\ell)^\infty}$
- **Output:**
  $(\sigma_0^\ell)^\infty = (\pi_0 \; \pi_{-1} \; \ldots \; \pi_{-\ell+2} \; \pi_{-\ell+1})^\infty$
  $(\sigma_1^\ell)^\infty = (\; \pi_0 \; \ldots \; \pi_{-\ell+3} \; \pi_{-\ell+2})^\infty$
  $(\sigma_2^\ell)^\infty = (\pi_2 \; \pi_1 \; \ldots \; \pi_{-\ell+4} \; \pi_{-\ell+3})^\infty$
  $\vdots \qquad \vdots$
  $(\sigma_k^\ell)^\infty = (\pi_k \; \pi_{k-1} \; \ldots \; \pi_{k-\ell+2} \; \pi_{k-\ell+1})^\infty$

Bridge between API=NSPI(1) and PSDP$_\infty$$\simeq$NSPI($\infty$).

Non-Stationary Policy Iteration

NSPI(ℓ)

$\pi_{k+1} \leftarrow \mathcal{G}_{\epsilon_k}(\nu, v_{(\sigma_k^\ell)^\infty})$

- $(\sigma_k^\ell)^\infty = (\pi_k \ \pi_{k-1} \ \ldots \ \pi_{k-\ell+1})^\infty$ is an infinite-horizon (ℓ-)periodic policy
- $v_{(\sigma_k^\ell)^\infty} = T_{\pi_k} T_{\pi_{k-1}} \ldots T_{\pi_{k-\ell+1}} v_{(\sigma_k^\ell)^\infty}$
- **Output:**

$$
\begin{aligned}
(\sigma_0^\ell)^\infty &= (\pi_0 \ \pi_{-1} \ \ldots \ \pi_{-\ell+2} \ \pi_{-\ell+1})^\infty \\
(\sigma_1^\ell)^\infty &= (\pi_1 \ \pi_0 \ \ldots \ \pi_{-\ell+3} \ \pi_{-\ell+2})^\infty \\
(\sigma_2^\ell)^\infty &= (\pi_2 \ \pi_1 \ \ldots \ \pi_{-\ell+4} \ \pi_{-\ell+3})^\infty \\
&\vdots \\
(\sigma_k^\ell)^\infty &= (\pi_k \ \pi_{k-1} \ \ldots \ \pi_{k-\ell+2} \ \pi_{k-\ell+1})^\infty
\end{aligned}
$$

Bridge between API=NSPI(1) and PSDP$_\infty$≃NSPI(∞).

Non-Stationary Policy Iteration

NSPI(ℓ)

$$\pi_{k+1} \leftarrow \mathcal{G}_{\epsilon_k}(\nu, v_{(\sigma_k^\ell)^\infty})$$

- $(\sigma_k^\ell)^\infty = (\pi_k \ \pi_{k-1} \ \ldots \ \pi_{k-\ell+1})^\infty$ is an infinite-horizon (ℓ-)periodic policy
- $v_{(\sigma_k^\ell)^\infty} = T_{\pi_k} T_{\pi_{k-1}} \ldots T_{\pi_{k-\ell+1}} v_{(\sigma_k^\ell)^\infty}$
- **Output:**

$$\begin{aligned}
(\sigma_0^\ell)^\infty &= (\pi_0 \ \pi_{-1} \ \ldots \ \pi_{-\ell+2} \ \pi_{-\ell+1})^\infty \\
(\sigma_1^\ell)^\infty &= (\pi_1 \ \pi_0 \ \ldots \ \pi_{-\ell+3} \ \pi_{-\ell+2})^\infty \\
(\sigma_2^\ell)^\infty &= (\pi_2 \ \pi_1 \ \ldots \ \pi_{-\ell+4} \ \pi_{-\ell+3})^\infty \\
&\vdots \\
(\sigma_k^\ell)^\infty &= (\pi_k \ \pi_{k-1} \ \ldots \ \pi_{k-\ell+2} \ \pi_{k-\ell+1})^\infty
\end{aligned}$$

Bridge between API=NSPI(1) and PSDP$_\infty$≃NSPI(∞).

Non-Stationary Policy Iteration

NSPI(ℓ)

$\pi_{k+1} \leftarrow \mathcal{G}_{\epsilon_k}(\nu, v_{(\sigma_k^\ell)^\infty})$

- $(\sigma_k^\ell)^\infty = (\pi_k \ \pi_{k-1} \ \ldots \ \pi_{k-\ell+1})^\infty$ is an infinite-horizon (ℓ-)periodic policy
- $v_{(\sigma_k^\ell)^\infty} = T_{\pi_k} T_{\pi_{k-1}} \ldots T_{\pi_{k-\ell+1}} v_{(\sigma_k^\ell)^\infty}$
- **Output:**

$$\begin{aligned}
(\sigma_0^\ell)^\infty &= (\pi_0 \ \pi_{-1} \ \ldots \ \pi_{-\ell+2} \ \pi_{-\ell+1})^\infty \\
(\sigma_1^\ell)^\infty &= (\pi_1 \ \pi_0 \ \ldots \ \pi_{-\ell+3} \ \pi_{-\ell+2})^\infty \\
(\sigma_2^\ell)^\infty &= (\pi_2 \ \pi_1 \ \ldots \ \pi_{-\ell+4} \ \pi_{-\ell+3})^\infty \\
&\ \vdots \\
(\sigma_k^\ell)^\infty &= (\pi_k \ \pi_{k-1} \ \ldots \ \pi_{k-\ell+2} \ \pi_{k-\ell+1})^\infty
\end{aligned}$$

Bridge between API=NSPI(1) and PSDP$_\infty$$\simeq$NSPI($\infty$).

Non-Stationary Policy Iteration

NSPI(ℓ)

$\pi_{k+1} \leftarrow \mathcal{G}_{\epsilon_k}(\nu, v_{(\sigma_k^\ell)^\infty})$

- $(\sigma_k^\ell)^\infty = (\pi_k \ \pi_{k-1} \ \ldots \ \pi_{k-\ell+1})^\infty$ is an infinite-horizon (ℓ-)periodic policy
- $v_{(\sigma_k^\ell)^\infty} = T_{\pi_k} T_{\pi_{k-1}} \ldots T_{\pi_{k-\ell+1}} v_{(\sigma_k^\ell)^\infty}$
- **Output:**

$$
\begin{aligned}
(\sigma_0^\ell)^\infty &= (\pi_0 \ \pi_{-1} \ \ldots \ \pi_{-\ell+2} \ \pi_{-\ell+1})^\infty \\
(\sigma_1^\ell)^\infty &= (\pi_1 \ \pi_0 \ \ldots \ \pi_{-\ell+3} \ \pi_{-\ell+2})^\infty \\
(\sigma_2^\ell)^\infty &= (\pi_2 \ \pi_1 \ \ldots \ \pi_{-\ell+4} \ \pi_{-\ell+3})^\infty \\
&\ \vdots \qquad\qquad\qquad \vdots \\
(\sigma_k^\ell)^\infty &= (\pi_k \ \pi_{k-1} \ \ldots \ \pi_{k-\ell+2} \ \pi_{k-\ell+1})^\infty
\end{aligned}
$$

Bridge between API=NSPI(1) and PSDP$_\infty \simeq$NSPI(∞).

Analysis (1/2): Worst-case bounds wrt resources

Algorithm	Performance Bound in $l_{1,\mu}$ norm			# Iter.	Memory
API	$C^{(2,1,0)}$	$\frac{1}{(1-\gamma)^2}$	ϵ	$\frac{1}{1-\gamma}\log\frac{1}{\epsilon}$	1
	$C^{(1,0)}$	$\frac{1}{(1-\gamma)^2}$	$\epsilon\log\frac{1}{\epsilon}$		
API(α)	$C^{(1,0)}$	$\frac{1}{(1-\gamma)^2}$	ϵ	$\frac{1}{\alpha(1-\gamma)}\log\frac{1}{\epsilon}$	
CPI(α)	$C^{(1,0)}$	$\frac{1}{(1-\gamma)^3}$	ϵ	$\frac{1}{\alpha(1-\gamma)}\log\frac{1}{\epsilon}$	
CPI	$C^{(1,0)}$	$\frac{1}{(1-\gamma)^3}$	$\epsilon\log\frac{1}{\epsilon}$	$\frac{1}{1-\gamma}\frac{1}{\epsilon}\log\frac{1}{\epsilon}$	
	C_{π_*}	$\frac{1}{(1-\gamma)^2}$	ϵ	$\frac{\gamma}{\epsilon^2}$	
PSDP$_\infty$	C_{π_*}	$\frac{1}{(1-\gamma)^2}$	$\epsilon\log\frac{1}{\epsilon}$	$\frac{1}{1-\gamma}\log\frac{1}{\epsilon}$	
	$C_{\pi_*}^{(1)}$	$\frac{1}{1-\gamma}$	ϵ	$\frac{1}{1-\gamma}\log\frac{1}{\epsilon}$	
NSPI(ℓ)	$C^{(2,\ell,0)}$	$\frac{1}{(1-\gamma)(1-\gamma^\ell)}$	ϵ	$\frac{1}{1-\gamma}\log\frac{1}{\epsilon}$	ℓ
	$\frac{C^{(1,0)}}{\ell}$	$\frac{1}{(1-\gamma)^2(1-\gamma^\ell)}$	$\epsilon\log\frac{1}{\epsilon}$	$\frac{1}{1-\gamma}\log\frac{1}{\epsilon}$	
	$C_{\pi_*}^{(1)} + \gamma^\ell\frac{C^{(2,\ell,\ell)}}{1-\gamma^\ell}$	$\frac{1}{1-\gamma}$	ϵ	$\frac{1}{1-\gamma}\log\frac{1}{\epsilon}$	
	$C_{\pi_*} + \gamma^\ell\frac{C^{(2,\ell,0)}}{\ell(1-\gamma^\ell)}$	$\frac{1}{(1-\gamma)^2}$	$\epsilon\log\frac{1}{\epsilon}$	$\frac{1}{1-\gamma}\log\frac{1}{\epsilon}$	

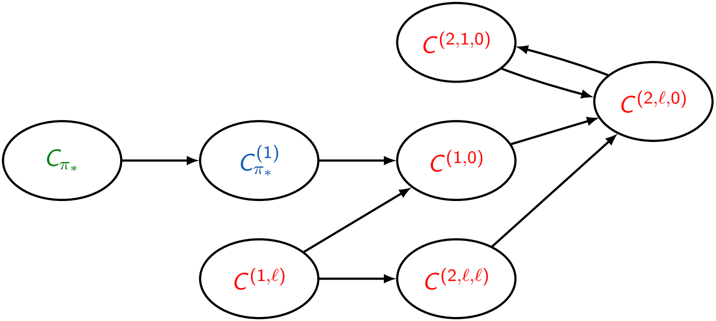
Analysis (2/2): Hierarchy of constants

Given coefficients that satisfy $\mu P_{\pi_1} P_{\pi_2} \dots P_{\pi_i} \leq c(i)\nu$ and $\mu (P_{\pi_*})^i \leq c_{\pi_*}(i)\nu$,

$$C^{(1,k)} = (1-\gamma)\sum_{i=0}^{\infty}\gamma^i c(i+k), \qquad C^{(1)}_{\pi_*} = (1-\gamma)\sum_{i=0}^{\infty}\gamma^i c_{\pi_*}(i),$$

$$C^{(2,\ell,k)} = (1-\gamma)(1-\gamma^\ell)\sum_{i=0}^{\infty}\sum_{j=0}^{\infty}\gamma^{i+j\ell} c(i+j\ell+k).$$

Define the coefficient that satisfies $d_{\pi_*,\mu} \leq C_{\pi_*}\nu$.



$A \to B$ <u>if and only if</u> $\{B < \infty \Rightarrow A < \infty\}$

Analysis (1'/2): Worst-case bounds wrt resources

Algorithm	Performance Bound in $l_{1,\mu}$ norm			# Iter.	Memory
API	$C^{(1,0)}$	$\frac{1}{(1-\gamma)^2}$	$\epsilon \log \frac{1}{\epsilon}$	$\frac{1}{1-\gamma} \log \frac{1}{\epsilon}$	1
CPI	C_{π_*}	$\frac{1}{(1-\gamma)^2}$	ϵ	$\frac{\gamma}{\epsilon^2}$	
PSDP_∞	C_{π_*}	$\frac{1}{(1-\gamma)^2}$	$\epsilon \log \frac{1}{\epsilon}$	$\frac{1}{1-\gamma} \log \frac{1}{\epsilon}$	
$\text{NSPI}(\ell)$	$C_{\pi_*} + \gamma^\ell \frac{C^{(2,\ell,0)}}{\ell(1-\gamma^\ell)}$	$\frac{1}{(1-\gamma)^2}$	$\epsilon \log \frac{1}{\epsilon}$	$\frac{1}{1-\gamma} \log \frac{1}{\epsilon}$	ℓ

- CPI arbitrarily better than API, but with exponentially more iterations
- PSDP_∞ enjoys the best of both worlds
- CPI and PSDP_∞ may require a lot of memory
  $\Rightarrow$ $\text{NSPI}(\ell)$ makes a trade-off between API and PSDP_∞

Analysis (1'/2): Worst-case bounds wrt resources

Algorithm	Performance Bound in $l_{1,\mu}$ norm			# Iter.	Memory
API	$C^{(1,0)}$	$\frac{1}{(1-\gamma)^2}$	$\epsilon \log \frac{1}{\epsilon}$	$\frac{1}{1-\gamma} \log \frac{1}{\epsilon}$	1
CPI	C_{π_*}	$\frac{1}{(1-\gamma)^2}$	ϵ	$\frac{\gamma}{\epsilon^2}$	
PSDP_∞	C_{π_*}	$\frac{1}{(1-\gamma)^2}$	$\epsilon \log \frac{1}{\epsilon}$	$\frac{1}{1-\gamma} \log \frac{1}{\epsilon}$	
$\text{NSPI}(\ell)$	$C_{\pi_*} + \gamma^\ell \frac{C^{(2,\ell,0)}}{\ell(1-\gamma^\ell)}$	$\frac{1}{(1-\gamma)^2}$	$\epsilon \log \frac{1}{\epsilon}$	$\frac{1}{1-\gamma} \log \frac{1}{\epsilon}$	ℓ

- CPI arbitrarily better than API, but with exponentially more iterations
- PSDP_∞ enjoys the best of both worlds
- CPI and PSDP_∞ may require a lot of memory
  $\Rightarrow$ $\text{NSPI}(\ell)$ makes a trade-off between API and PSDP_∞

Analysis (1'/2): Worst-case bounds wrt resources

Algorithm	Performance Bound in $l_{1,\mu}$ norm	# Iter.	Memory
API	$C^{(1,0)} \quad \frac{1}{(1-\gamma)^2} \quad \epsilon \log \frac{1}{\epsilon}$	$\frac{1}{1-\gamma} \log \frac{1}{\epsilon}$	1
CPI	$C_{\pi_*} \quad \frac{1}{(1-\gamma)^2} \quad \epsilon$	$\frac{\gamma}{\epsilon^2}$	
PSDP_∞	$C_{\pi_*} \quad \frac{1}{(1-\gamma)^2} \quad \epsilon \log \frac{1}{\epsilon}$	$\frac{1}{1-\gamma} \log \frac{1}{\epsilon}$	
$\text{NSPI}(\ell)$	$C_{\pi_*} + \gamma^\ell \frac{C^{(2,\ell,0)}}{\ell(1-\gamma^\ell)} \quad \frac{1}{(1-\gamma)^2} \quad \epsilon \log \frac{1}{\epsilon}$	$\frac{1}{1-\gamma} \log \frac{1}{\epsilon}$	ℓ

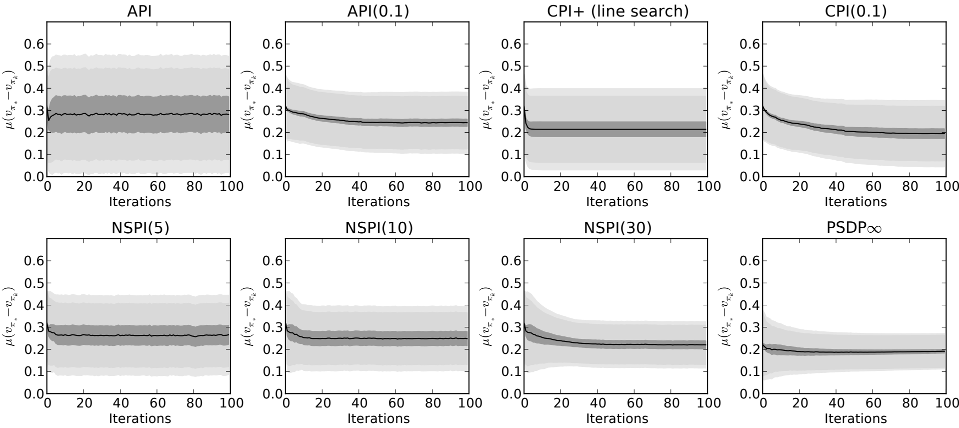
- CPI arbitrarily better than API, but with exponentially more iterations
- PSDP_∞ enjoys the best of both worlds
- CPI and PSDP_∞ may require a lot of memory
  $\Rightarrow$ $\text{NSPI}(\ell)$ makes a trade-off between API and PSDP_∞

Analysis (1'/2): Worst-case bounds wrt resources

Algorithm	Performance Bound in $l_{1,\mu}$ norm	# Iter.	Memory
API	$C^{(1,0)} \quad \frac{1}{(1-\gamma)^2} \quad \epsilon \log \frac{1}{\epsilon}$	$\frac{1}{1-\gamma} \log \frac{1}{\epsilon}$	1
CPI	$C_{\pi_*} \quad \frac{1}{(1-\gamma)^2} \quad \epsilon$	$\frac{\gamma}{\epsilon^2}$	
PSDP_∞	$C_{\pi_*} \quad \frac{1}{(1-\gamma)^2} \quad \epsilon \log \frac{1}{\epsilon}$	$\frac{1}{1-\gamma} \log \frac{1}{\epsilon}$	
$\text{NSPI}(\ell)$	$C_{\pi_*} + \gamma^\ell \frac{C^{(2,\ell,0)}}{\ell(1-\gamma^\ell)} \quad \frac{1}{(1-\gamma)^2} \quad \epsilon \log \frac{1}{\epsilon}$	$\frac{1}{1-\gamma} \log \frac{1}{\epsilon}$	ℓ

- CPI arbitrarily better than API, but with exponentially more iterations
- PSDP_∞ enjoys the best of both worlds
- CPI and PSDP_∞ may require a lot of memory
  $\Rightarrow$ $\text{NSPI}(\ell)$ makes a trade-off between API and PSDP_∞

Numerical Simulations



Experiments made on $3^3 * 30 \simeq 800$ Garnet problems.

For each problem, one runs 30 times each algorithm.