Performance bounds for Approximate Dynamic Programming in Large Scale Infinite-Horizon Optimal Control Problems

Bruno Scherrer, Victor Gabillon, Mohammad Ghavamzadeh, Matthieu Geist

INRIA Lorraine, LORIA, MAIA Team

Stochastic Optimization Workshop, Cadarache, June 28, 2012.

Markov Decision Process

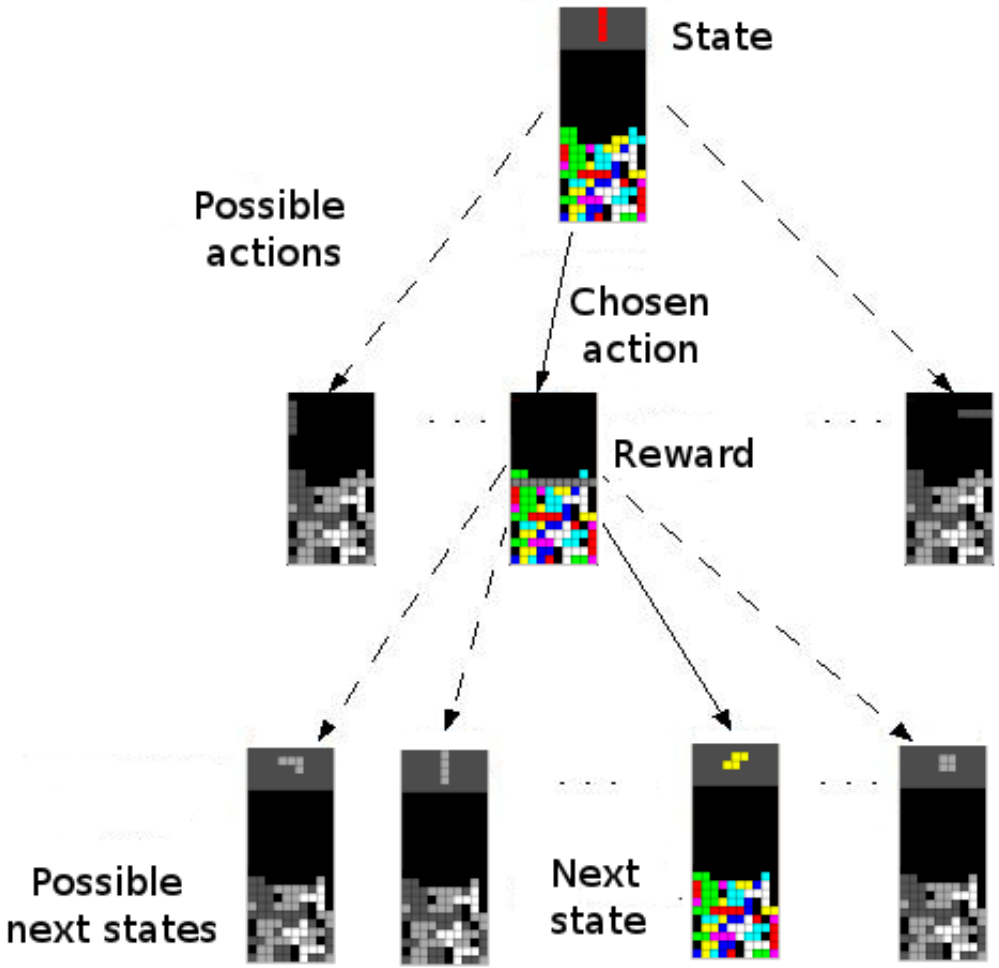
A MDP $\mathcal{M}$ is a tuple $\langle \mathcal{X}, \mathcal{A}, r, p, \gamma \rangle$, where

- $\mathcal{X}$ is a (finite) state space $\mathcal{X}$,
- $\mathcal{A}$ is a **finite** action space,
- for all $x \in \mathcal{X}$, $p(\cdot|x, a)$ is a distribution over $\mathcal{X}$,
- $r : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ is a (deterministic) reward function
- $\gamma \in (0, 1)$ is a discount factor.

Goal : Find a policy $\pi : \mathcal{X} \to \mathcal{A}$ maximizing the **value function** v_π, <u>for all</u> x :

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

An illustrative example : Tetris



Bellman Equations, Notations I

- For any policy π, v_π is the unique solution of the **Bellman equation** :

$$\forall x, \quad v_\pi(x) = r(x, \pi(x)) + \gamma \sum_{y \in \mathcal{X}} p(y|x, \pi(x)) v_\pi(y)$$

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

- The **optimal value** v_* is the unique solution of the **Bellman optimality equation** :

$$\forall x, \quad v_*(x) = \max_{a \in \mathcal{A}} \left(r(x, a) + \gamma \sum_{y \in \mathcal{X}} p(y|x, a) v_*(y) \right)$$

$$\Leftrightarrow \quad v_* = T v_* \quad \Leftrightarrow \quad v_* = \max_\pi T_\pi v_*.$$

- π is a **greedy policy** w.r.t. v , written $\pi = \mathcal{G} v$, iff

$$\forall x, \quad \pi(x) \in \arg\max_{a \in \mathcal{A}} \left(r(x, a) + \gamma \sum_{y \in \mathcal{X}} p(y|x, a) v(y) \right)$$

$$\Leftrightarrow \quad T_\pi v = T v.$$

Bellman Equations, Notations II

- The **action value** function q_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

$$\forall x, \quad r(x,a) + \gamma \sum_{y \in \mathcal{X}} p(y|x,a) q_\pi(y,\pi(y)) \quad \Leftrightarrow \quad q_\pi = T_\pi q_\pi$$

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

- And the following relations hold :

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

Bellman Optimality Equation

We just need to solve the following system

$$
\begin{aligned}
v_*(x_1) &= \max_{a\in\mathcal{A}} \Big(r(x_1, a) + \gamma \sum_{y\in\mathcal{X}} p(y|x_1, a) v_*(y) \Big) \\
v_*(x_2) &= \max_{a\in\mathcal{A}} \Big(r(x_2, a) + \gamma \sum_{y\in\mathcal{X}} p(y|x_2, a) v_*(y) \Big) \\
v_*(x_3) &= \max_{a\in\mathcal{A}} \Big(r(x_3, a) + \gamma \sum_{y\in\mathcal{X}} p(y|x_3, a) v_*(y) \Big) \\
v_*(x_4) &= \max_{a\in\mathcal{A}} \Big(r(x_4, a) + \gamma \sum_{y\in\mathcal{X}} p(y|x_4, a) v_*(y) \Big) \\
&\vdots \qquad\qquad \vdots \\
v_*(x_N) &= \max_{a\in\mathcal{A}} \Big(r(x_N, a) + \gamma \sum_{y\in\mathcal{X}} p(y|x_N, a) v_*(y) \Big)
\end{aligned}
$$

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 (Puterman & Shin, 1978)

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

λ Policy Iteration (Bertsekas & Ioffe, 1996; Scherrer, 2007)

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

Optimistic Policy Iteration (Thiery & Scherrer, 2009; Thiery & Scherrer, 2010)

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

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 (Puterman & Shin, 1978)

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

λ Policy Iteration (Bertsekas & Ioffe, 1996; Scherrer, 2007)

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

Optimistic Policy Iteration (Thiery & Scherrer, 2009; Thiery & Scherrer, 2010)

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

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 (Puterman & Shin, 1978)

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

λ Policy Iteration (Bertsekas & Ioffe, 1996; Scherrer, 2007)

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

Optimistic Policy Iteration (Thiery & Scherrer, 2009; Thiery & Scherrer, 2010)

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

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 (Puterman & Shin, 1978)

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

λ Policy Iteration (Bertsekas & Ioffe, 1996; Scherrer, 2007)

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

Optimistic Policy Iteration (Thiery & Scherrer, 2009; Thiery & Scherrer, 2010)

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

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 (Puterman & Shin, 1978)

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

λ Policy Iteration (Bertsekas & Ioffe, 1996; Scherrer, 2007)

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

Optimistic Policy Iteration (Thiery & Scherrer, 2009; Thiery & Scherrer, 2010)

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

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 (Puterman & Shin, 1978)

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

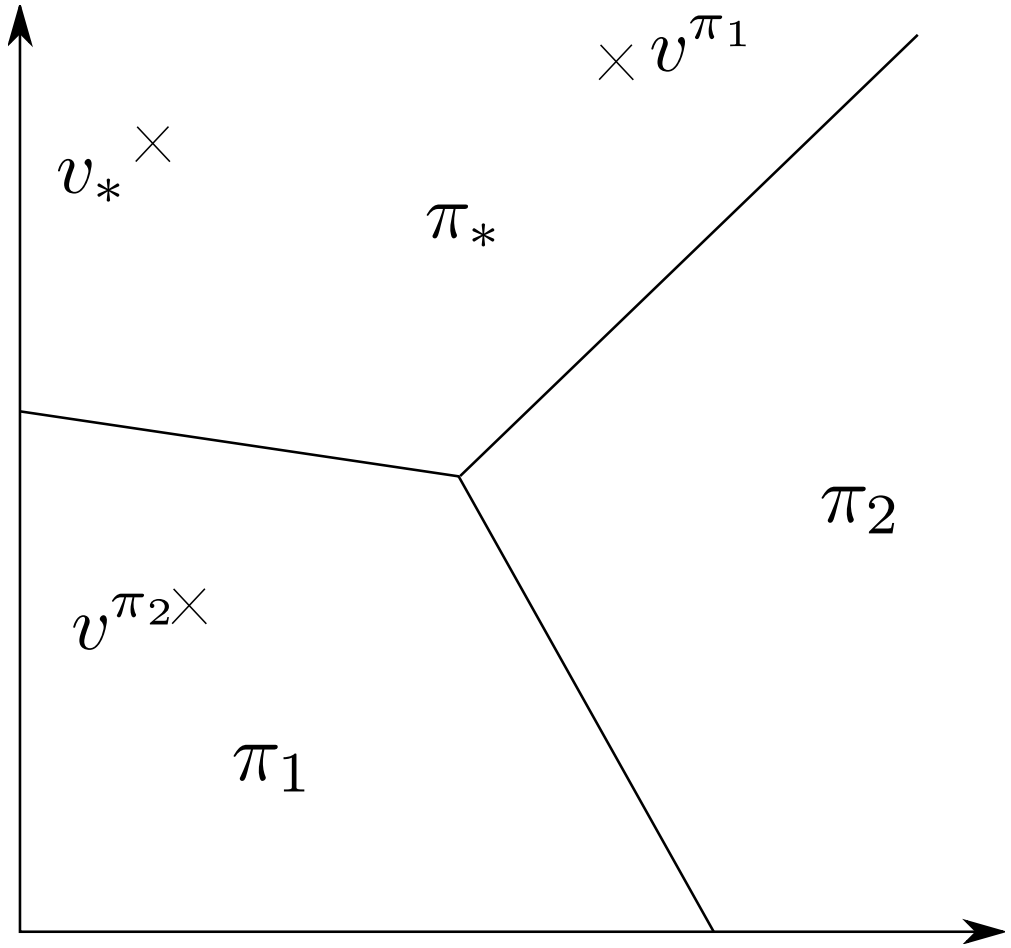
λ Policy Iteration (Bertsekas & Ioffe, 1996; Scherrer, 2007)

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

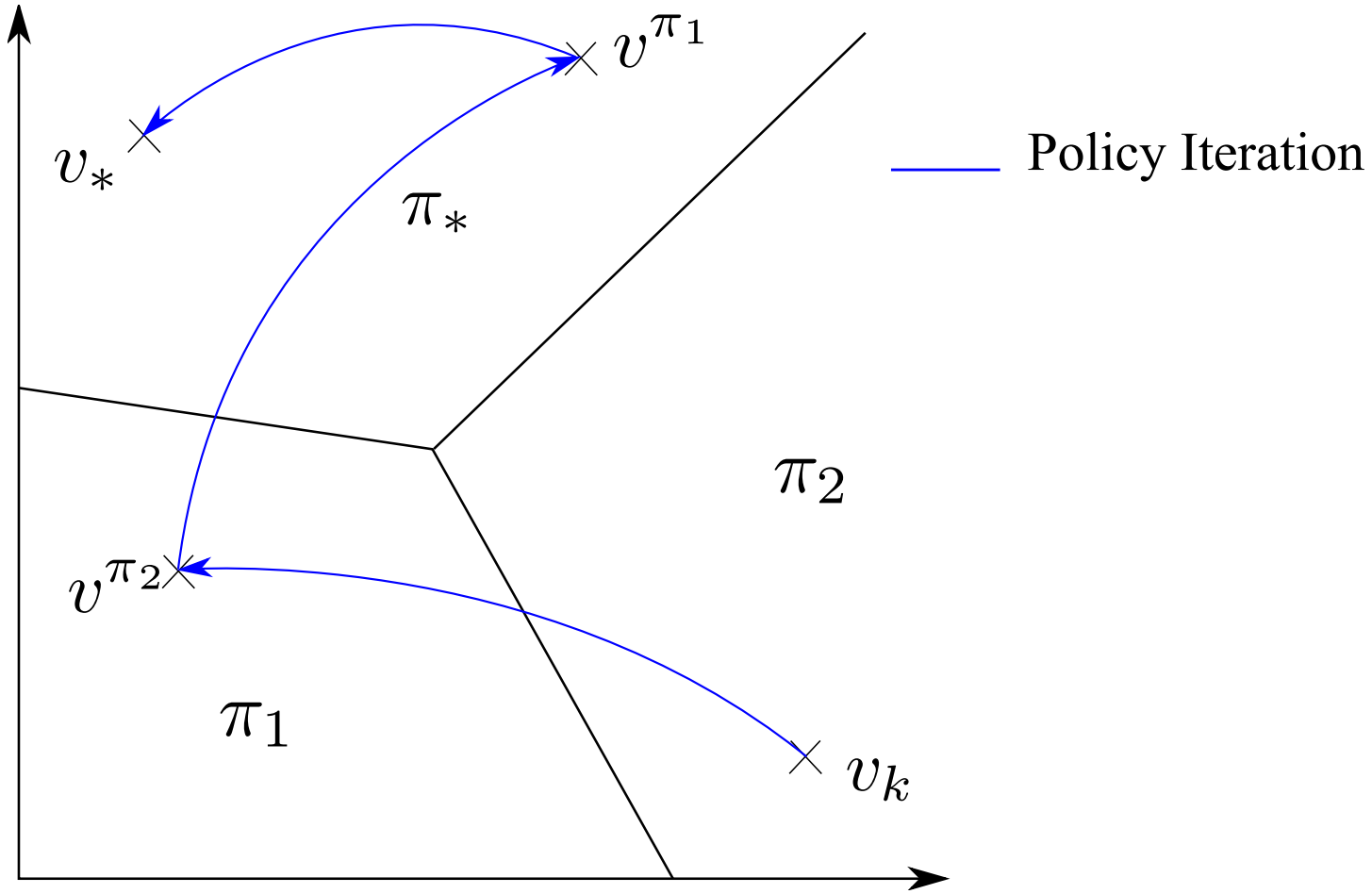
Optimistic Policy Iteration (Thiery & Scherrer, 2009; Thiery & Scherrer, 2010)

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

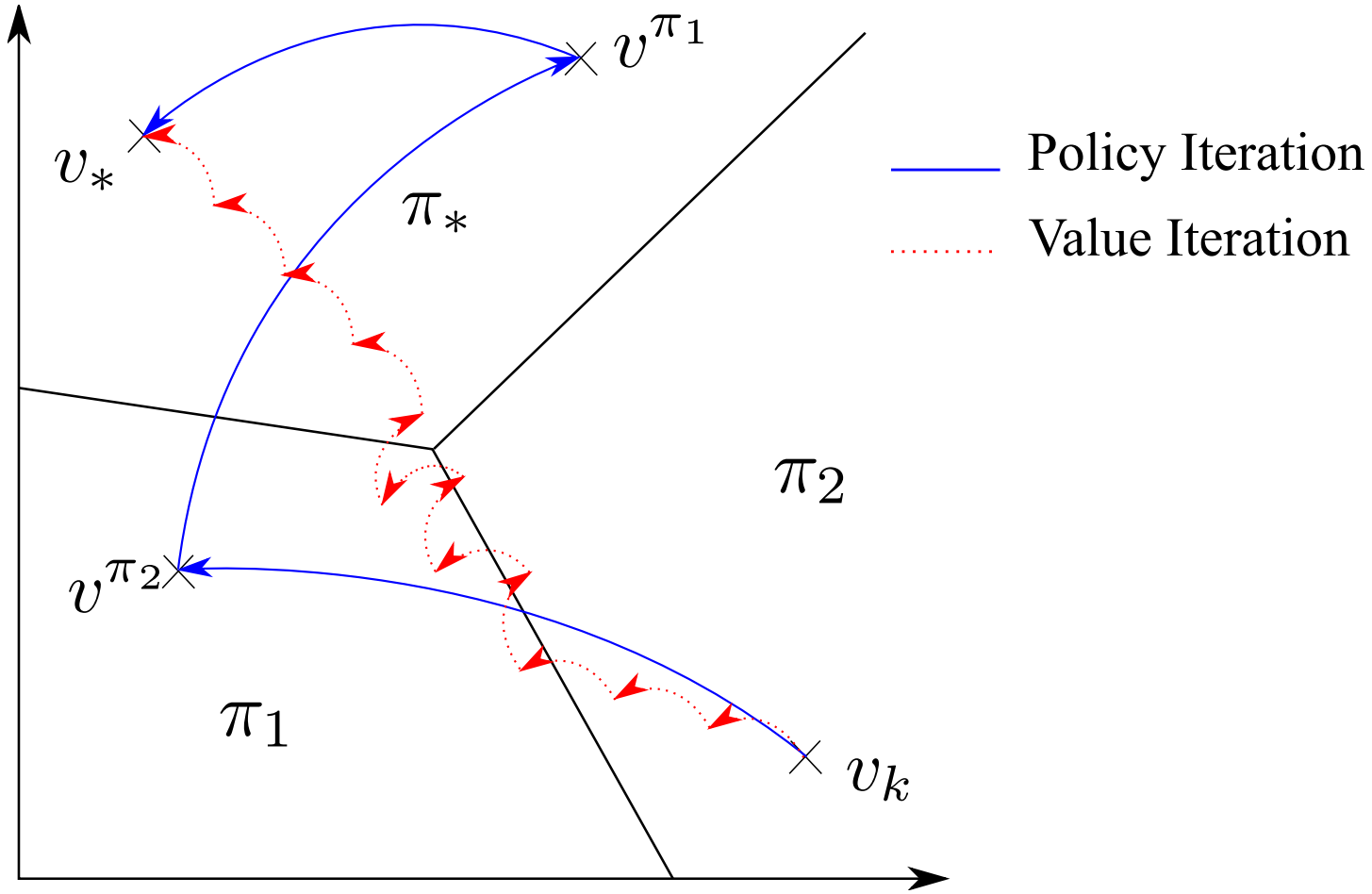
Optimism in the greedy partition (Bertsekas & Tsitsiklis, 1996)



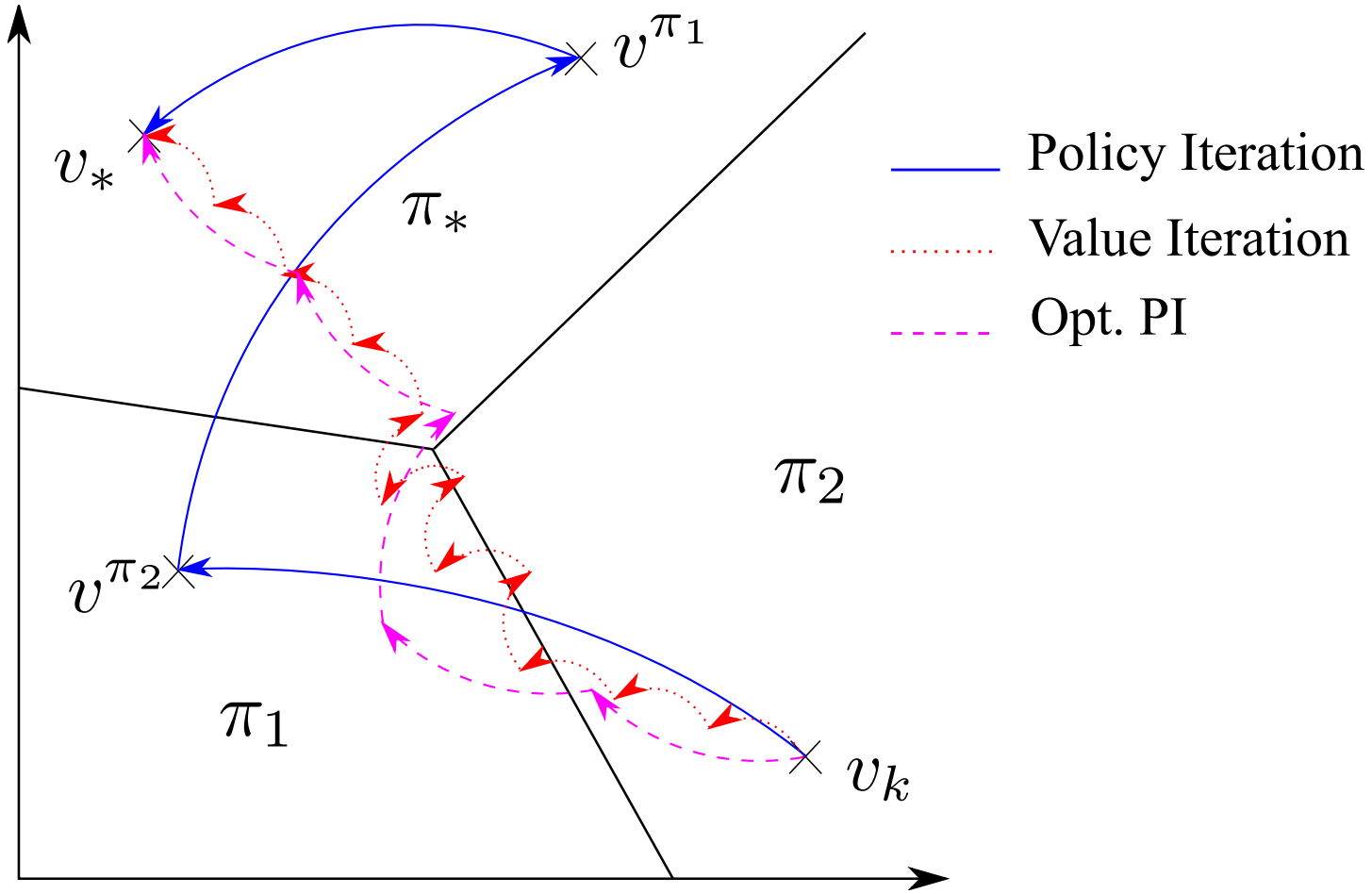
Optimism in the greedy partition (Bertsekas & Tsitsiklis, 1996)



Optimism in the greedy partition (Bertsekas & Tsitsiklis, 1996)



Optimism in the greedy partition (Bertsekas & Tsitsiklis, 1996)



Bellman Optimality Equation

We simply need to solve the following system

$$
\begin{aligned}
v_*(x_1) &= \max_{a\in\mathcal{A}} \Big(r(x_1,a) + \gamma \sum_{y\in\mathcal{X}} p(y|x_1,a) v_*(y) \Big) \\
v_*(x_2) &= \max_{a\in\mathcal{A}} \Big(r(x_2,a) + \gamma \sum_{y\in\mathcal{X}} p(y|x_2,a) v_*(y) \Big) \\
v_*(x_3) &= \max_{a\in\mathcal{A}} \Big(r(x_3,a) + \gamma \sum_{y\in\mathcal{X}} p(y|x_3,a) v_*(y) \Big) \\
v_*(x_4) &= \max_{a\in\mathcal{A}} \Big(r(x_4,a) + \gamma \sum_{y\in\mathcal{X}} p(y|x_4,a) v_*(y) \Big) \\
&\vdots \qquad\qquad \vdots \\
v_*(x_N) &= \max_{a\in\mathcal{A}} \Big(r(x_N,a) + \gamma \sum_{y\in\mathcal{X}} p(y|x_N,a) v_*(y) \Big)
\end{aligned}
$$

Tetris : $N \simeq 2^{10\times 20} \simeq 10^{60}$ states !

Bellman Optimality Equation

We simply need to solve the following system

$$
\begin{aligned}
v_\theta(x_1) &\simeq \max_{a\in\mathcal{A}} \Big(r(x_1,a) + \gamma \sum_{y\in\mathcal{X}} p(y|x_1,a) v_\theta(y) \Big) \\
v_\theta(x_2) &\simeq \max_{a\in\mathcal{A}} \Big(r(x_2,a) + \gamma \sum_{y\in\mathcal{X}} p(y|x_2,a) v_\theta(y) \Big) \\
v_\theta(x_3) &\simeq \max_{a\in\mathcal{A}} \Big(r(x_3,a) + \gamma \sum_{y\in\mathcal{X}} p(y|x_3,a) v_\theta(y) \Big) \\
v_\theta(x_4) &\simeq \max_{a\in\mathcal{A}} \Big(r(x_4,a) + \gamma \sum_{y\in\mathcal{X}} p(y|x_4,a) v_\theta(y) \Big) \\
&\vdots \qquad\qquad \vdots \\
v_\theta(x_N) &\simeq \max_{a\in\mathcal{A}} \Big(r(x_N,a) + \gamma \sum_{y\in\mathcal{X}} p(y|x_N,a) v_\theta(y) \Big)
\end{aligned}
$$

Tetris : $N \simeq 2^{10\times 20} \simeq 10^{60}$ states ! $\theta \in \mathbb{R}^p$, $p \ll N$

Bellman Optimality Equation

We simply need to solve the following system

$$
\begin{aligned}
v_\theta(x_1) &\simeq \max_{a\in\mathcal{A}} \Big(r(x_1,a) + \gamma \sum_{y\in\mathcal{X}} p(y|x_1,a) v_\theta(y) \Big) \\
v_\theta(x_2) &\simeq \max_{a\in\mathcal{A}} \Big(r(x_2,a) + \gamma \sum_{y\in\mathcal{X}} p(y|x_2,a) v_\theta(y) \Big) \\
v_\theta(x_3) &\simeq \max_{a\in\mathcal{A}} \Big(r(x_3,a) + \gamma \sum_{y\in\mathcal{X}} p(y|x_3,a) v_\theta(y) \Big) \\
v_\theta(x_4) &\simeq \max_{a\in\mathcal{A}} \Big(r(x_4,a) + \gamma \sum_{y\in\mathcal{X}} p(y|x_4,a) v_\theta(y) \Big) \\
&\vdots \qquad\qquad \vdots \\
v_\theta(x_N) &\simeq \max_{a\in\mathcal{A}} \Big(r(x_N,a) + \gamma \sum_{y\in\mathcal{X}} p(y|x_N,a) v_\theta(y) \Big)
\end{aligned}
$$

Tetris : $N \simeq 2^{10\times 20} \simeq 10^{60}$ states ! $\theta \in \mathbb{R}^p$, $p \ll N$, $k \ll N$ samples

Illustration of approximation on Tetris

1. Approximation architecture for v_* :

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

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

2. Sampling Scheme : play !

Outline

Outline

Approximate MPI

- $\pi_{k+1} \leftarrow \mathcal{G} v_k$ **policy (actor) update**
- $v_{k+1} \leftarrow (T_{\pi_{k+1}})^m v_k$ **value function (critic) update**

Actual implementations depend on the way the policy and the value function are represented

Approximate MPI-V

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

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

Value function update

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

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

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

$$v_{k+1} = \arg\min_{v \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} \left(v(x^{(i)}) - \widehat{v}_{k+1}(x^{(i)}) \right)^2 .$$

Approximate MPI-V

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

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

Value function update

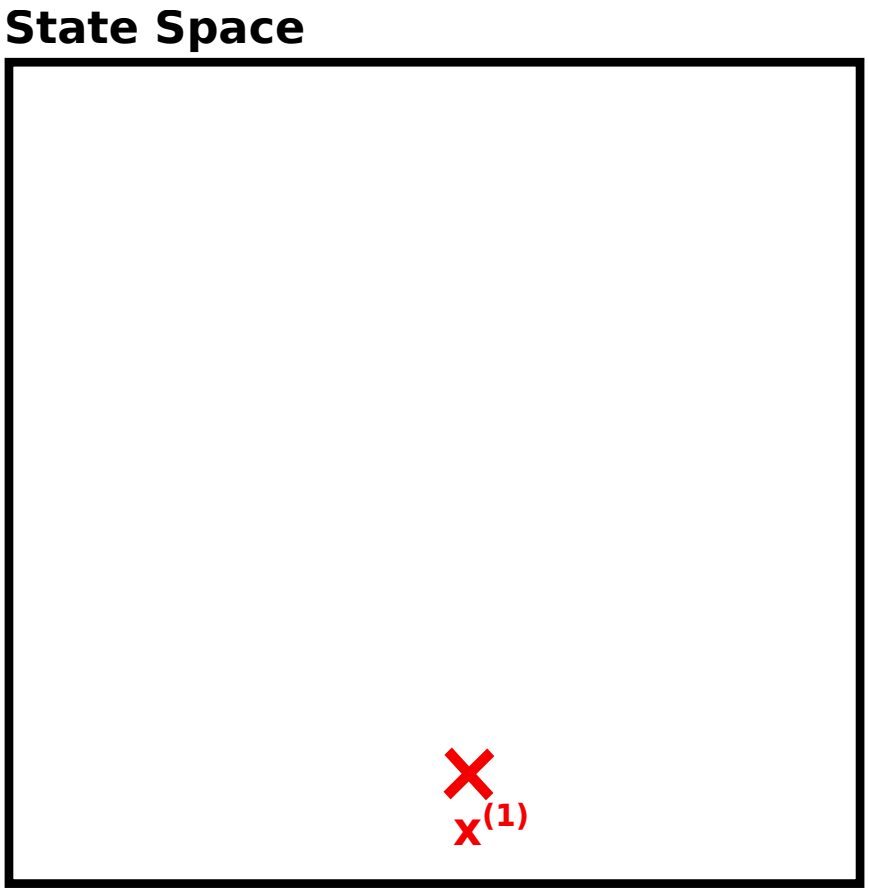
1. **Point-wise estimation** through rollouts of length m :
   For N states $x^{(i)} \sim \mu$, compute an unbiased estimate of $\left[\left(T_{\pi_{k+1}}\right)^m v_k\right](x^{(i)})$ (using π_{k+1} m times)

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

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

$$v_{k+1} = \arg\min_{v \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} \left(v(x^{(i)}) - \widehat{v}_{k+1}(x^{(i)}) \right)^2 .$$

Approximate MPI-V



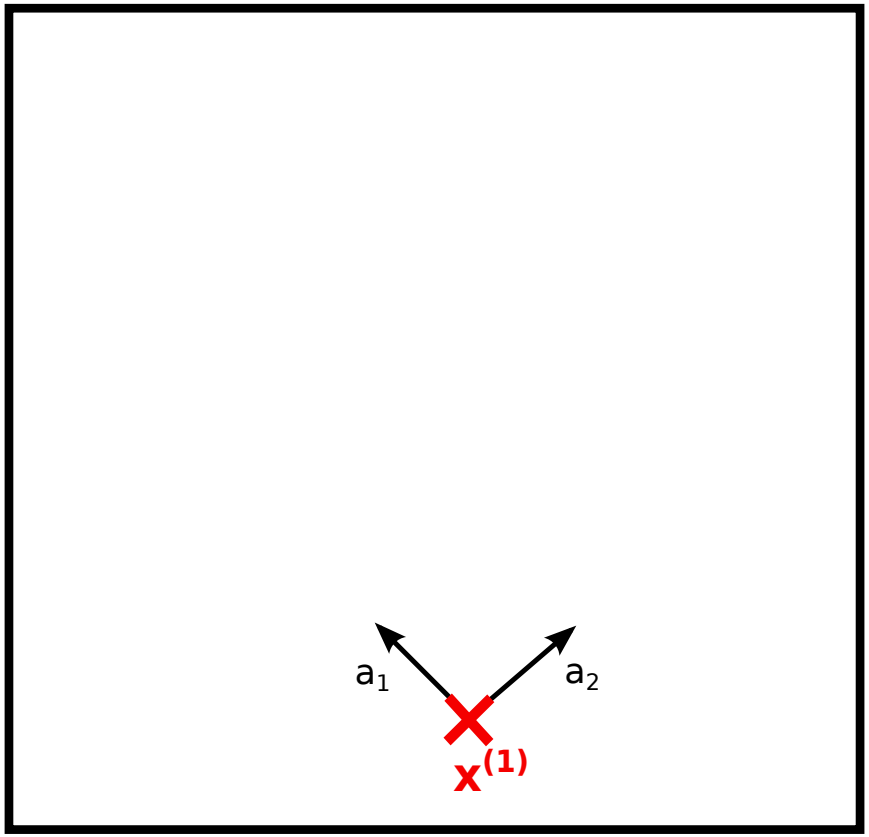
Policy update

In state x, the greedy action is estimated by :

$$\pi_{k+1}(x) = \arg\max_{a \in \mathcal{A}} \frac{1}{M} \Big(\sum_{j=1}^{M} r_a^{(j)} + \gamma v_k(x_a^{(j)}) \Big)$$

Approximate MPI-V

State Space



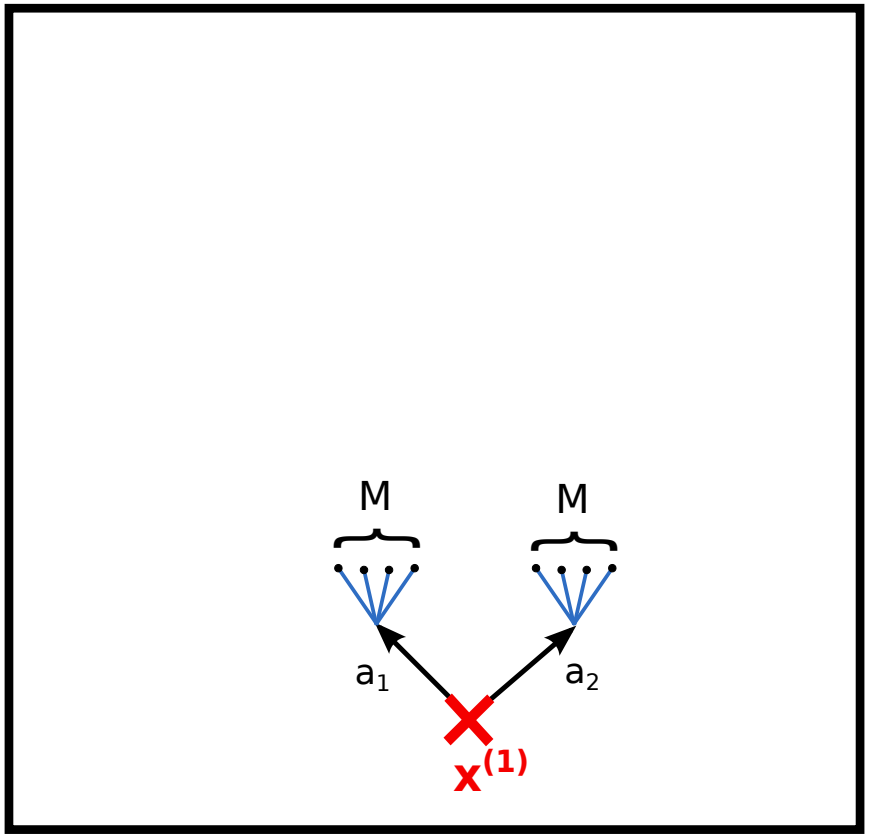
Policy update

In state x, the greedy action is estimated by :

$$\pi_{k+1}(x) = \arg\max_{a \in \mathcal{A}} \frac{1}{M} \Big(\sum_{j=1}^{M} r_a^{(j)} + \gamma v_k(x_a^{(j)}) \Big)$$

Approximate MPI-V

State Space



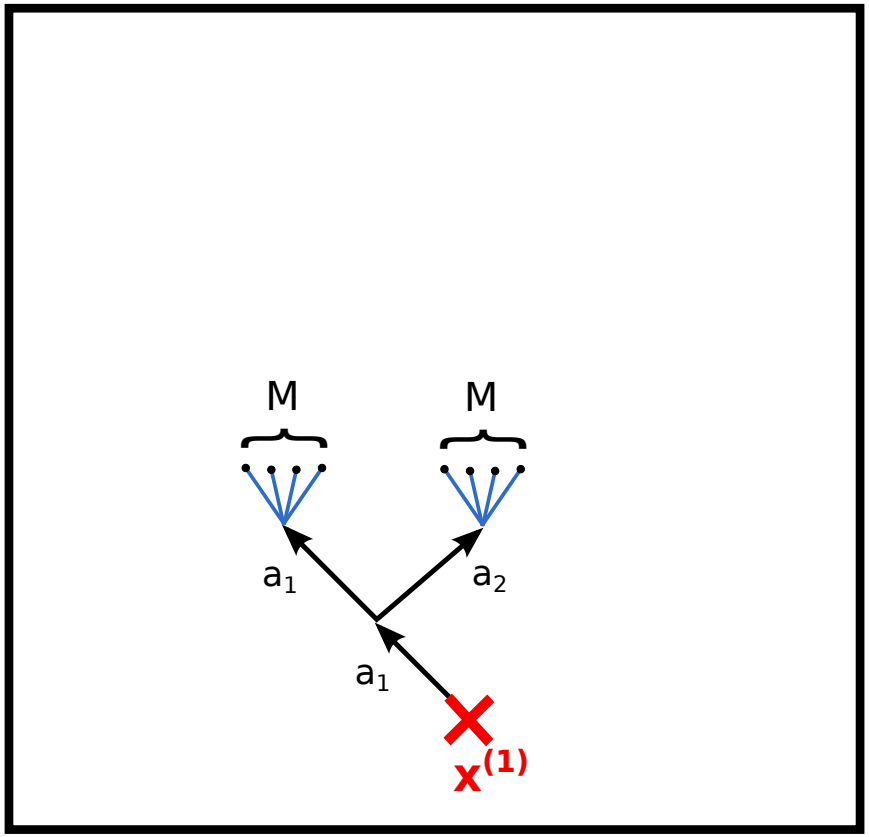
Policy update

In state x, the greedy action is estimated by :

$$\pi_{k+1}(x) = \arg\max_{a \in \mathcal{A}} \frac{1}{M} \Big(\sum_{j=1}^{M} r_a^{(j)} + \gamma v_k(x_a^{(j)}) \Big)$$

Approximate MPI-V

State Space



Policy update

In state x, the greedy action is estimated by :

$$\pi_{k+1}(x) = \arg\max_{a \in \mathcal{A}} \frac{1}{M} \Big(\sum_{j=1}^{M} r_a^{(j)} + \gamma v_k(x_a^{(j)}) \Big)$$

Approximate MPI-V

Related algorithms : Fitted VI (Munos & Szepesvári, 2008) is a special case of AMPI-V when $m = 1$.

Sample complexity : $Nm(M|A| + 1)$ transition samples.

Approximate MPI-Q

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

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

Policy update

In state x, the greedy action is estimated by :

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

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

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

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

Policy update

In state x, the greedy action is estimated by :

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

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

Related algorithms : Fitted Q (Ernst et al. , 2005; Antos et al. , 2007) is a special case of AMPI-Q when $m = 1$.

Sample complexity : Nm transition samples.

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

Same as in AMPI-V !

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

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

2. **Generalisation through regression**

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

Same as in AMPI-V !

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

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

2. **Generalisation through regression**

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

Same as in AMPI-V !

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

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

2. **Generalisation** through regression

Classification-based MPI

Policy update

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

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

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 :

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

Classification-based MPI

Related algorithms : DPI (Lazaric et al. , 2010) is a special case of CBMPI when $m = +\infty$ (see also (Fern et al. , 2006; Lagoudakis & Parr, 2003; Gabillon et al. , 2011)).

Sample complexity : $nm + NM|\mathcal{A}|(m+1)$ or $NM|\mathcal{A}|(m+1)$ transition samples in case of reuse of the rollouts.

Outline

Outline

The need for a new analysis

- **VI** analysis is based on the fact that the Bellman optimality operator is a γ-**contraction** in max-norm.
- **PI** analysis relies on the fact that the sequence of the generated values is **non-decreasing**.
- None of these lines of proofs apply to MPI !

An Abstract Approximate AMPI Algorithm

Greedy step error : ϵ'_k

$\epsilon'_k > 0$ is the error in fitting the greedy policy, written

$$\pi_k = \widehat{\mathcal{G}}_{\epsilon'_k} v_{k-1},$$

meaning that for all π,

$$T_{\pi'} v_{k-1} \leq T_{\pi_k} v_{k-1} + \epsilon'_k$$

Evaluation step error : ϵ_k

ϵ_k is the error in fitting the $(T_{\pi_{k+1}})^m v_k$.

$$v_k \leftarrow (T_{\pi_k})^m v_{k-1} + \epsilon_k$$

Point-wise Error Propagation Overview

We study the loss due to running π_k instead of π_* :

$$0 \leq v_* - v_{\pi_k} \leq \underbrace{v_* - (v_k - \epsilon_k)}_{\mathbf{d_k}} + \underbrace{(v_k - \epsilon_k) - v_{\pi_k}}_{\mathbf{s_k}}$$

Lemma Point-wise inequalities

Define $b_k \triangleq v_k - T_{\pi_{k+1}} v_k$. The following inequalities hold

$$b_k \leq (\gamma P_{\pi_k})^m b_{k-1} + x_k,$$

$$d_{k+1} \leq \gamma P_{\pi_*} d_k + y_k + \sum_{j=1}^{m-1} (\gamma P_{\pi_{k+1}})^j b_k,$$

$$s_k = (\gamma P_{\pi_k})^m (I - \gamma P_{\pi_k})^{-1} b_{k-1},$$

where $x_k \triangleq (I - \gamma P_{\pi_k})\epsilon_k + \epsilon'_{k+1}$ and $y_k \triangleq -\gamma P_{\pi_*} \epsilon_k + \epsilon'_{k+1}$.

If $|\epsilon_k|$ and $|\epsilon'_{k+1}|$ are $O(\epsilon)$, then $b_k \leq O(\epsilon)$, which implies that $d_k \leq O(\epsilon)$ and $s_k \leq O(\epsilon)$. So Finally $v_* - v_{\pi_k} \leq O(\epsilon)$.

Proof of Lemma : Bounding b_k

$$
\begin{aligned}
b_k &= v_k - T_{\pi_{k+1}} v_k \\
&= v_k - T_{\pi_k} v_k + \underbrace{T_{\pi_k} v_k - T_{\pi_{k+1}} v_k}_{\leq \epsilon'_{k+1}} \\
&\leq v_k - T_{\pi_k} v_k + \epsilon'_{k+1} \\
&= v_k - \epsilon_k - T_{\pi_k} v_k + \gamma P_{\pi_k} \epsilon_k + \epsilon_k - \gamma P_{\pi_k} \epsilon_k + \epsilon'_{k+1} \\
&= \underbrace{v_k - \epsilon_k}_{(T_{\pi_k})^m v_{k-1}} - T_{\pi_k}(\underbrace{v_k - \epsilon_k}_{(T_{\pi_k})^m v_{k-1}}) + \underbrace{(I - \gamma P_{\pi_k})\epsilon_k + \epsilon'_{k+1}}_{x_k} \\
&= (T_{\pi_k})^m v_{k-1} - T_{\pi_k}(T_{\pi_k})^m v_{k-1} + x_k \\
&= (T_{\pi_k})^m v_{k-1} - (T_{\pi_k})^m (T_{\pi_k} v_{k-1}) + x_k \\
&= (\gamma P_{\pi_k})^m (v_{k-1} - T_{\pi_k} v_{k-1}) + x_k \\
&= (\gamma P_{\pi_k})^m b_{k-1} + x_k.
\end{aligned}
$$

Proof of Lemma : Bounding b_k

$$
\begin{aligned}
b_k &= v_k - T_{\pi_{k+1}} v_k \\
&= v_k - T_{\pi_k} v_k + \underbrace{T_{\pi_k} v_k - T_{\pi_{k+1}} v_k}_{\leq \epsilon'_{k+1}} \\
&\leq v_k - T_{\pi_k} v_k + \epsilon'_{k+1} \\
&= v_k - \epsilon_k - T_{\pi_k} v_k + \gamma P_{\pi_k} \epsilon_k + \epsilon_k - \gamma P_{\pi_k} \epsilon_k + \epsilon'_{k+1} \\
&= \underbrace{v_k - \epsilon_k}_{(T_{\pi_k})^m v_{k-1}} - T_{\pi_k}(\underbrace{v_k - \epsilon_k}_{(T_{\pi_k})^m v_{k-1}}) + \underbrace{(I - \gamma P_{\pi_k})\epsilon_k + \epsilon'_{k+1}}_{x_k} \\
&= (T_{\pi_k})^m v_{k-1} - T_{\pi_k}(T_{\pi_k})^m v_{k-1} + x_k \\
&= (T_{\pi_k})^m v_{k-1} - (T_{\pi_k})^m (T_{\pi_k} v_{k-1}) + x_k \\
&= (\gamma P_{\pi_k})^m (v_{k-1} - T_{\pi_k} v_{k-1}) + x_k \\
&= (\gamma P_{\pi_k})^m b_{k-1} + x_k.
\end{aligned}
$$

Proof of Lemma : Bounding b_k

$$
\begin{aligned}
b_k &= v_k - T_{\pi_{k+1}} v_k \\
&= v_k - T_{\pi_k} v_k + \underbrace{T_{\pi_k} v_k - T_{\pi_{k+1}} v_k}_{\leq \epsilon'_{k+1}} \\
&\leq v_k - T_{\pi_k} v_k + \epsilon'_{k+1} \\
&= v_k - \epsilon_k - T_{\pi_k} v_k + \gamma P_{\pi_k} \epsilon_k + \epsilon_k - \gamma P_{\pi_k} \epsilon_k + \epsilon'_{k+1} \\
&= \underbrace{v_k - \epsilon_k}_{(T_{\pi_k})^m v_{k-1}} - T_{\pi_k}(\underbrace{v_k - \epsilon_k}_{(T_{\pi_k})^m v_{k-1}}) + \underbrace{(I - \gamma P_{\pi_k})\epsilon_k + \epsilon'_{k+1}}_{x_k} \\
&= (T_{\pi_k})^m v_{k-1} - T_{\pi_k}(T_{\pi_k})^m v_{k-1} + x_k \\
&= (T_{\pi_k})^m v_{k-1} - (T_{\pi_k})^m (T_{\pi_k} v_{k-1}) + x_k \\
&= (\gamma P_{\pi_k})^m (v_{k-1} - T_{\pi_k} v_{k-1}) + x_k \\
&= (\gamma P_{\pi_k})^m b_{k-1} + x_k.
\end{aligned}
$$

Proof of Lemma : Bounding b_k

$$
\begin{aligned}
b_k &= v_k - T_{\pi_{k+1}} v_k \\
&= v_k - T_{\pi_k} v_k + \underbrace{T_{\pi_k} v_k - T_{\pi_{k+1}} v_k}_{\leq \epsilon'_{k+1}} \\
&\leq v_k - T_{\pi_k} v_k + \epsilon'_{k+1} \\
&= v_k - \epsilon_k - T_{\pi_k} v_k + \gamma P_{\pi_k} \epsilon_k + \epsilon_k - \gamma P_{\pi_k} \epsilon_k + \epsilon'_{k+1} \\
&= \underbrace{v_k - \epsilon_k}_{(T_{\pi_k})^m v_{k-1}} - T_{\pi_k}(\underbrace{v_k - \epsilon_k}_{(T_{\pi_k})^m v_{k-1}}) + \underbrace{(I - \gamma P_{\pi_k})\epsilon_k + \epsilon'_{k+1}}_{x_k} \\
&= (T_{\pi_k})^m v_{k-1} - T_{\pi_k}(T_{\pi_k})^m v_{k-1} + x_k \\
&= (T_{\pi_k})^m v_{k-1} - (T_{\pi_k})^m (T_{\pi_k} v_{k-1}) + x_k \\
&= (\gamma P_{\pi_k})^m (v_{k-1} - T_{\pi_k} v_{k-1}) + x_k \\
&= (\gamma P_{\pi_k})^m b_{k-1} + x_k.
\end{aligned}
$$

Proof of Lemma : Bounding b_k

$$
\begin{aligned}
b_k &= v_k - T_{\pi_{k+1}} v_k \\
&= v_k - T_{\pi_k} v_k + \underbrace{T_{\pi_k} v_k - T_{\pi_{k+1}} v_k}_{\leq \epsilon'_{k+1}} \\
&\leq v_k - T_{\pi_k} v_k + \epsilon'_{k+1} \\
&= v_k - \epsilon_k - T_{\pi_k} v_k + \gamma P_{\pi_k} \epsilon_k + \epsilon_k - \gamma P_{\pi_k} \epsilon_k + \epsilon'_{k+1} \\
&= \underbrace{v_k - \epsilon_k}_{(T_{\pi_k})^m v_{k-1}} - T_{\pi_k}(\underbrace{v_k - \epsilon_k}_{(T_{\pi_k})^m v_{k-1}}) + \underbrace{(I - \gamma P_{\pi_k})\epsilon_k + \epsilon'_{k+1}}_{x_k} \\
&= (T_{\pi_k})^m v_{k-1} - T_{\pi_k}(T_{\pi_k})^m v_{k-1} + x_k \\
&= (T_{\pi_k})^m v_{k-1} - (T_{\pi_k})^m (T_{\pi_k} v_{k-1}) + x_k \\
&= (\gamma P_{\pi_k})^m (v_{k-1} - T_{\pi_k} v_{k-1}) + x_k \\
&= (\gamma P_{\pi_k})^m b_{k-1} + x_k.
\end{aligned}
$$

Proof of Lemma : Bounding b_k

$$
\begin{aligned}
b_k &= v_k - T_{\pi_{k+1}} v_k \\
&= v_k - T_{\pi_k} v_k + \underbrace{T_{\pi_k} v_k - T_{\pi_{k+1}} v_k}_{\leq \epsilon'_{k+1}} \\
&\leq v_k - T_{\pi_k} v_k + \epsilon'_{k+1} \\
&= v_k - \epsilon_k - T_{\pi_k} v_k + \gamma P_{\pi_k} \epsilon_k + \epsilon_k - \gamma P_{\pi_k} \epsilon_k + \epsilon'_{k+1} \\
&= \underbrace{v_k - \epsilon_k}_{(T_{\pi_k})^m v_{k-1}} - T_{\pi_k}(\underbrace{v_k - \epsilon_k}_{(T_{\pi_k})^m v_{k-1}}) + \underbrace{(I - \gamma P_{\pi_k})\epsilon_k + \epsilon'_{k+1}}_{x_k} \\
&= (T_{\pi_k})^m v_{k-1} - T_{\pi_k}(T_{\pi_k})^m v_{k-1} + x_k \\
&= (T_{\pi_k})^m v_{k-1} - (T_{\pi_k})^m (T_{\pi_k} v_{k-1}) + x_k \\
&= (\gamma P_{\pi_k})^m (v_{k-1} - T_{\pi_k} v_{k-1}) + x_k \\
&= (\gamma P_{\pi_k})^m b_{k-1} + x_k.
\end{aligned}
$$

Proof of Lemma : Bounding b_k

$$
\begin{aligned}
b_k &= v_k - T_{\pi_{k+1}} v_k \\
&= v_k - T_{\pi_k} v_k + \underbrace{T_{\pi_k} v_k - T_{\pi_{k+1}} v_k}_{\leq \epsilon'_{k+1}} \\
&\leq v_k - T_{\pi_k} v_k + \epsilon'_{k+1} \\
&= v_k - \epsilon_k - T_{\pi_k} v_k + \gamma P_{\pi_k} \epsilon_k + \epsilon_k - \gamma P_{\pi_k} \epsilon_k + \epsilon'_{k+1} \\
&= \underbrace{v_k - \epsilon_k}_{(T_{\pi_k})^m v_{k-1}} - T_{\pi_k}(\underbrace{v_k - \epsilon_k}_{(T_{\pi_k})^m v_{k-1}}) + \underbrace{(I - \gamma P_{\pi_k})\epsilon_k + \epsilon'_{k+1}}_{x_k} \\
&= (T_{\pi_k})^m v_{k-1} - T_{\pi_k}(T_{\pi_k})^m v_{k-1} + x_k \\
&= (T_{\pi_k})^m v_{k-1} - (T_{\pi_k})^m (T_{\pi_k} v_{k-1}) + x_k \\
&= (\gamma P_{\pi_k})^m (v_{k-1} - T_{\pi_k} v_{k-1}) + x_k \\
&= (\gamma P_{\pi_k})^m b_{k-1} + x_k.
\end{aligned}
$$

Proof of Lemma : Bounding b_k

$$
\begin{aligned}
b_k &= v_k - T_{\pi_{k+1}} v_k \\
&= v_k - T_{\pi_k} v_k + \underbrace{T_{\pi_k} v_k - T_{\pi_{k+1}} v_k}_{\leq \epsilon'_{k+1}} \\
&\leq v_k - T_{\pi_k} v_k + \epsilon'_{k+1} \\
&= v_k - \epsilon_k - T_{\pi_k} v_k + \gamma P_{\pi_k} \epsilon_k + \epsilon_k - \gamma P_{\pi_k} \epsilon_k + \epsilon'_{k+1} \\
&= \underbrace{v_k - \epsilon_k}_{(T_{\pi_k})^m v_{k-1}} - T_{\pi_k}(\underbrace{v_k - \epsilon_k}_{(T_{\pi_k})^m v_{k-1}}) + \underbrace{(I - \gamma P_{\pi_k})\epsilon_k + \epsilon'_{k+1}}_{x_k} \\
&= (T_{\pi_k})^m v_{k-1} - T_{\pi_k}(T_{\pi_k})^m v_{k-1} + x_k \\
&= (T_{\pi_k})^m v_{k-1} - (T_{\pi_k})^m (T_{\pi_k} v_{k-1}) + x_k \\
&= (\gamma P_{\pi_k})^m (v_{k-1} - T_{\pi_k} v_{k-1}) + x_k \\
&= (\gamma P_{\pi_k})^m b_{k-1} + x_k.
\end{aligned}
$$

Proof of Lemma : Bounding b_k

$$
\begin{aligned}
b_k &= v_k - T_{\pi_{k+1}} v_k \\
&= v_k - T_{\pi_k} v_k + \underbrace{T_{\pi_k} v_k - T_{\pi_{k+1}} v_k}_{\leq \epsilon'_{k+1}} \\
&\leq v_k - T_{\pi_k} v_k + \epsilon'_{k+1} \\
&= v_k - \epsilon_k - T_{\pi_k} v_k + \gamma P_{\pi_k} \epsilon_k + \epsilon_k - \gamma P_{\pi_k} \epsilon_k + \epsilon'_{k+1} \\
&= \underbrace{v_k - \epsilon_k}_{(T_{\pi_k})^m v_{k-1}} - T_{\pi_k}(\underbrace{v_k - \epsilon_k}_{(T_{\pi_k})^m v_{k-1}}) + \underbrace{(I - \gamma P_{\pi_k})\epsilon_k + \epsilon'_{k+1}}_{x_k} \\
&= (T_{\pi_k})^m v_{k-1} - T_{\pi_k}(T_{\pi_k})^m v_{k-1} + x_k \\
&= (T_{\pi_k})^m v_{k-1} - (T_{\pi_k})^m (T_{\pi_k} v_{k-1}) + x_k \\
&= (\gamma P_{\pi_k})^m (v_{k-1} - T_{\pi_k} v_{k-1}) + x_k \\
&= (\gamma P_{\pi_k})^m b_{k-1} + x_k.
\end{aligned}
$$

From Point-wise to L_p norm bounds

Using Γ as a generic notation for γQ, where Q is some stochastic matrix, we get :

Lemma Point-wise performance loss bound

The loss of the AMPI-V/Q algorithms after k iterations satisfies

$$v_* - v_{\pi_k} \leq 2 \sum_{i=1}^{k-1} \sum_{j=i}^{\infty} \Gamma^j |\epsilon_{k-i}| + \sum_{i=0}^{k-1} \sum_{j=i}^{\infty} \Gamma^j |\epsilon'_{k-i}| + h(k),$$

where $h(k) = 2 \sum_{j=k}^{\infty} \Gamma^j |d_0|$ or $h(k) = 2 \sum_{j=k}^{\infty} \Gamma^j |b_0|$.

Error Propagation for AMPI-V/Q

Theorem L_p **bounds for the loss of AMPI-V/Q**

After k iterations, the loss of AMPI-V/Q satisfies

$$\|v_* - v_{\pi_k}\|_{p,\rho} \leq \frac{2(\gamma - \gamma^k)\left(\mathcal{C}_q\right)^{\frac{1}{p}}}{(1-\gamma)^2} \sup_{1 \leq j \leq k-1} \|\epsilon_j\|_{pq',\mu} + \frac{(1-\gamma^k)\left(\mathcal{C}'_q\right)^{\frac{1}{p}}}{(1-\gamma)^2} \sup_{1 \leq j \leq k} \|\epsilon'_j\|_{pq',\mu} + g(k),$$

with $\frac{1}{q} + \frac{1}{q'} = 1$ and $g(k) = \frac{2\gamma^k}{1-\gamma}\left(\mathcal{C}''_q\right)^{\frac{1}{p}} \min\left(\|v_* - v_0\|_{pq',\mu}, \|v_0 - T_{\pi_1} v_0\|_{pq',\mu}\right)$

Generalisations of previous results

- When k and p tend to infinity the bound of the Th. gives us a generalization of the API ($m = \infty$) bound of Bertsekas & Tsitsiklis (1996, Prop. 6.2), $\limsup_{k\to\infty} \|v_* - v_{\pi_k}\|_\infty \leq \frac{2\gamma \sup_j \|\epsilon_j\|_\infty + \sup_j \|\epsilon'_j\|_\infty}{(1-\gamma)^2}$.
- In p norm, our results unifies AVI and API with $\epsilon'_j = 0$ as analysed in (Munos, 2003; Munos, 2007).

Error Propagation for CBMPI

Theorem L_p bounds for the loss of CBMPI

After k iterations, the loss of CBMPI satisfies

$$\|v_* - v_{\pi_k}\|_{p,\rho} \leq \frac{2\,\gamma^{\mathbf{m}}\left(\gamma - \gamma^{k-1}\right)\left(\mathcal{C}_q\right)^{\frac{1}{p}}}{(1-\gamma)^2} \sup_{1\leq j\leq k-1} \|\epsilon_j\|_{pq',\mu} + \frac{\left(1-\gamma^k\right)\left(\mathcal{C}'_q\right)^{\frac{1}{p}}}{(1-\gamma)^2} \sup_{1\leq j\leq k} \|\epsilon'_j\|_{pq',\mu} + g(k),$$

with $\frac{1}{q} + \frac{1}{q'} = 1$ and $g(k) = \frac{2\gamma^k}{1-\gamma}\left(\mathcal{C}''_q\right)^{\frac{1}{p}} \min\left(\|v_* - v_0\|_{pq',\mu}, \|v_0 - T_{\pi_1} v_0\|_{pq',\mu}\right)$

Remarks

- m controls the influence of the value function approximator, cancelling it out in the limit when m tends to infinity ($\rightarrow$ API).
- Generalizes the analysis of (Lazaric *et al.* , 2010) for $m = \infty$.

Finite sample analysis of CBMPI

greedy step error : Using classification analysis tools we get :

$$\|\epsilon'_k\|_{1,\mu} \leq \mathbf{d'_m} + O(V_{\max}\sqrt{\frac{1}{n}}) + O(V_{\max}\sqrt{\frac{1}{Mn}})$$

with the **approximation error of the actor :**

$$\mathbf{d'_m} = \sup_{v\in\mathcal{F},\pi'} \inf_{\pi\in\Pi} \sum_{x\in\mathcal{X}} \left[\max_a Q_{\pi',v}(x,a) - Q_{\pi',v}(x,\pi(x))\right]\mu(x).$$

evaluation step error : Using regression analysis tools we get :

$$\|\epsilon_k\|_{2,\mu} \leq \mathbf{d_m} + O(V_{\max}\sqrt{\frac{1}{n}})$$

with the **approximation error of the critic :**

$$\mathbf{d_m} = \sup_{g\in\mathcal{F},\pi} \inf_{f\in\mathcal{F}} \|(T_\pi)^m g - f\|_{2,\mu}$$

Error Propagation for CBMPI (detailed)

Corollary

Assume a fixed budget of samples $B = nm = NMA(m+1)$ for each iteration, after k iterations, and with high probability, the expected loss of CBMPI satisfies

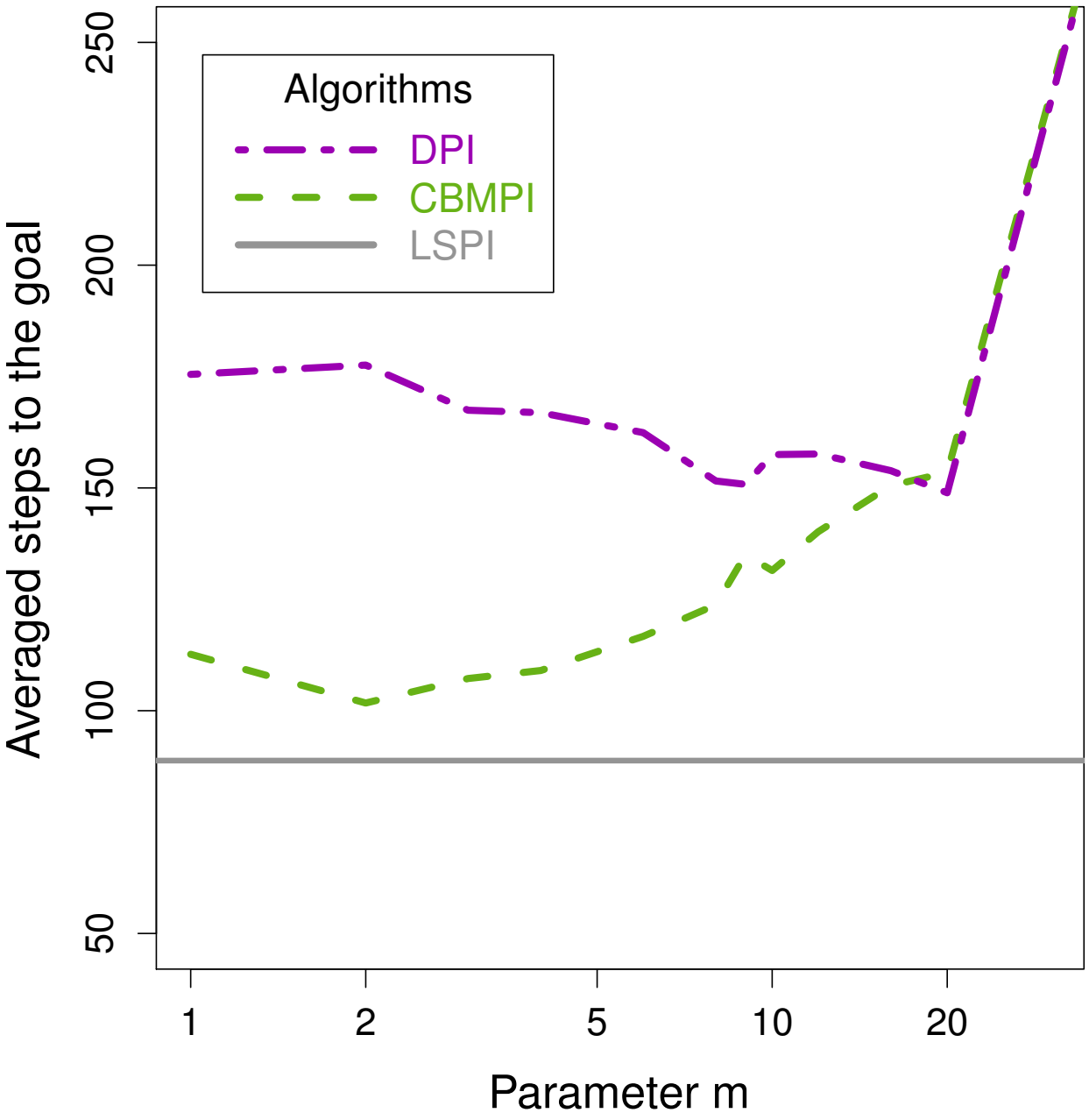
$$\|v_* - v_{\pi_k}\|_{1,\mu} \leq \tilde{O}\left(\gamma^{\mathbf{m}}\left(\mathbf{d_m} + \sqrt{\frac{m}{B}}\right) + \mathbf{d'_m} + \sqrt{\frac{|\mathcal{A}|m}{B}}\right).$$

Trade-off in the tuning of m : a large value of m induces large estimations errors, but a smaller overall influence of the value estimate.

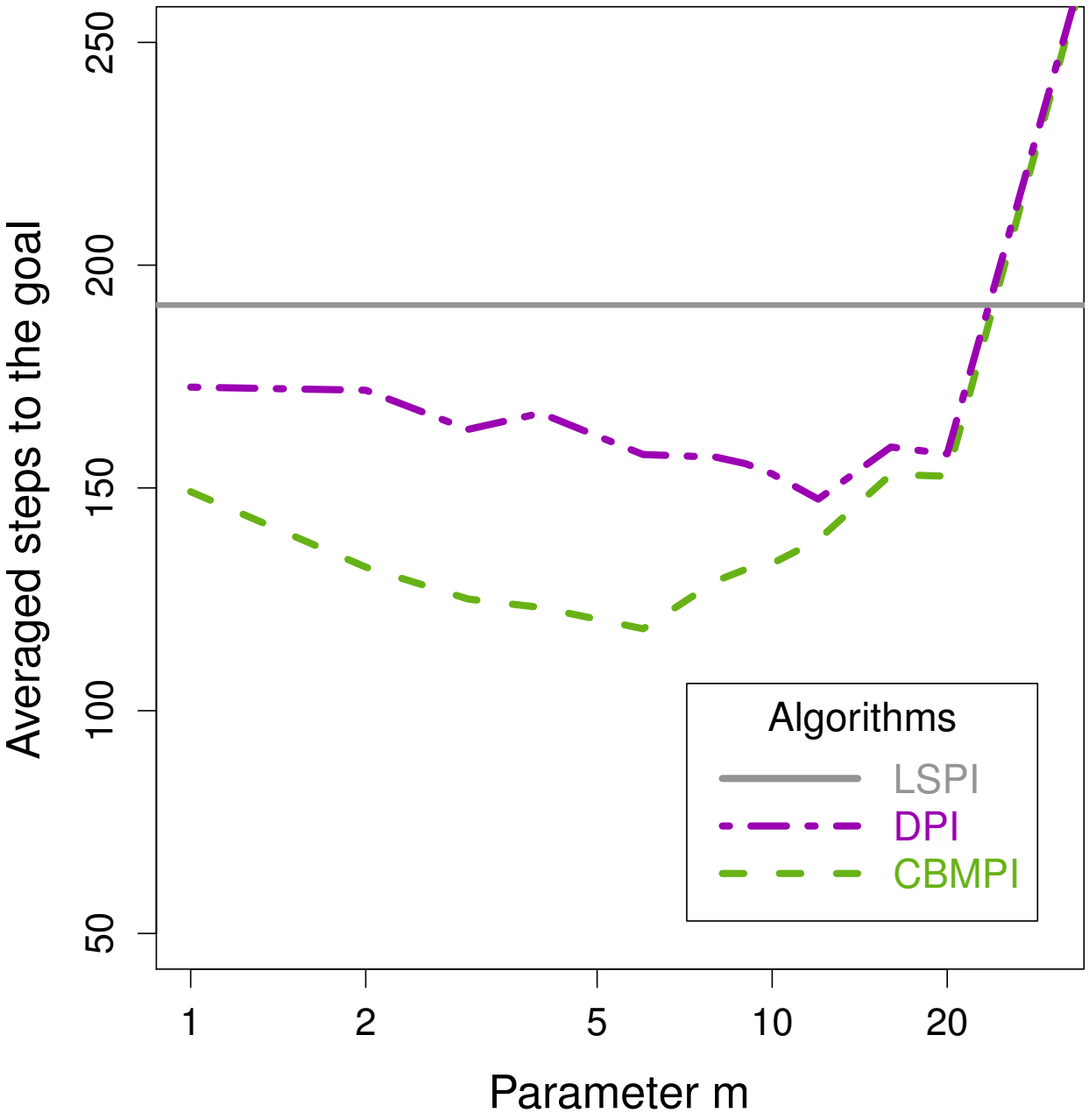
Outline

Outline

Good value function approximator



Bad value function approximator



Conclusions and Future Work

This talk

- We studied modified policy iteration (MPI), that contains both policy and value iteration algorithms
- Three approximate implementations of MPI that extend well-known ADP algorithms : fitted-value iteration, fitted-Q iteration, and classification-based policy iteration
- An error propagation analysis that unifies those of AVI and API
- A detailed finite-sample analysis for the classification-based implementation of AMPI (CBMPI)
- Our results indicate that, in CBMPI, the parameter m of MPI allows us to balance between the errors of critic and actor

Future work

- A detailed analysis of AMPI-V/Q
- More challenging empirical evaluation (Tetris).

References I

Antos, A., Munos, R., & Szepesvári, Cs. 2007.
Fitted Q-iteration in Continuous Action-space MDPs.
Pages 9–16 of : Proceedings of NIPS.

Bertsekas, D.P., & Ioffe, S. 1996.
Temporal differences-based policy iteration and applications in neuro-dynamic programming.
Tech. rept. LIDS-P-2349. MIT.

Bertsekas, D.P., & Tsitsiklis, J.N. 1996.
Neurodynamic Programming.
Athena Scientific.

Ernst, D., Geurts, P., & Wehenkel, L. 2005.
Tree-Based Batch Mode Reinforcement Learning.
Journal of Machine Learning Research, **6**, 503–556.

Fern, A., Yoon, S., & Givan, R. 2006.
Approximate Policy Iteration with a Policy Language Bias : Solving Relational Markov Decision Processes.
Journal of Artificial Intelligence Research, **25**, 75–118.

Gabillon, V., Lazaric, A., Ghavamzadeh, M., & Scherrer, B. 2011.
Classification-based Policy Iteration with a Critic.
Pages 1049–1056 of : Proceedings of ICML.

Lagoudakis, M., & Parr, R. 2003.
Reinforcement Learning as Classification : Leveraging Modern Classifiers.
Pages 424–431 of : Proceedings of ICML.

References II

Lazaric, A., Ghavamzadeh, M., & Munos, R. 2010.
Analysis of a Classification-based Policy Iteration Algorithm.
Pages 607–614 of : Proceedings of ICML.

Munos, R. 2003.
Error Bounds for Approximate Policy Iteration.
Pages 560–567 of : Proceedings of ICML.

Munos, R. 2007.
Performance Bounds in L_p-norm for Approximate Value Iteration.
SIAM J. Control and Optimization, **46**(2), 541–561.

Munos, R., & Szepesvári, Cs. 2008.
Finite-Time Bounds for Fitted Value Iteration.
Journal of Machine Learning Research, **9**, 815–857.

Puterman, M., & Shin, M. 1978.
Modified policy iteration algorithms for discounted Markov decision problems.
Management Science, **24**(11).

Scherrer, Bruno. 2007.
Performance Bounds for λ Policy Iteration.
Tech. rept. INRIA.

Thiery, C., & Scherrer, B. 2010.
Least-Squares λ Policy Iteration : Bias-Variance Trade-off in Control Problems.
In : ICML.

Thiery, Christophe, & Scherrer, Bruno. 2009 (06).
Une approche modifiée de Lambda-Policy Iteration.
In : Journées Francophones Planification Décisio n Apprentissage.
UPMC-Paris 6, Paris France.

Counter example

Proposition

If $m > 1$, there exists no norm for which the operator that MPI uses to update the values from one iteration to the next is a contraction.

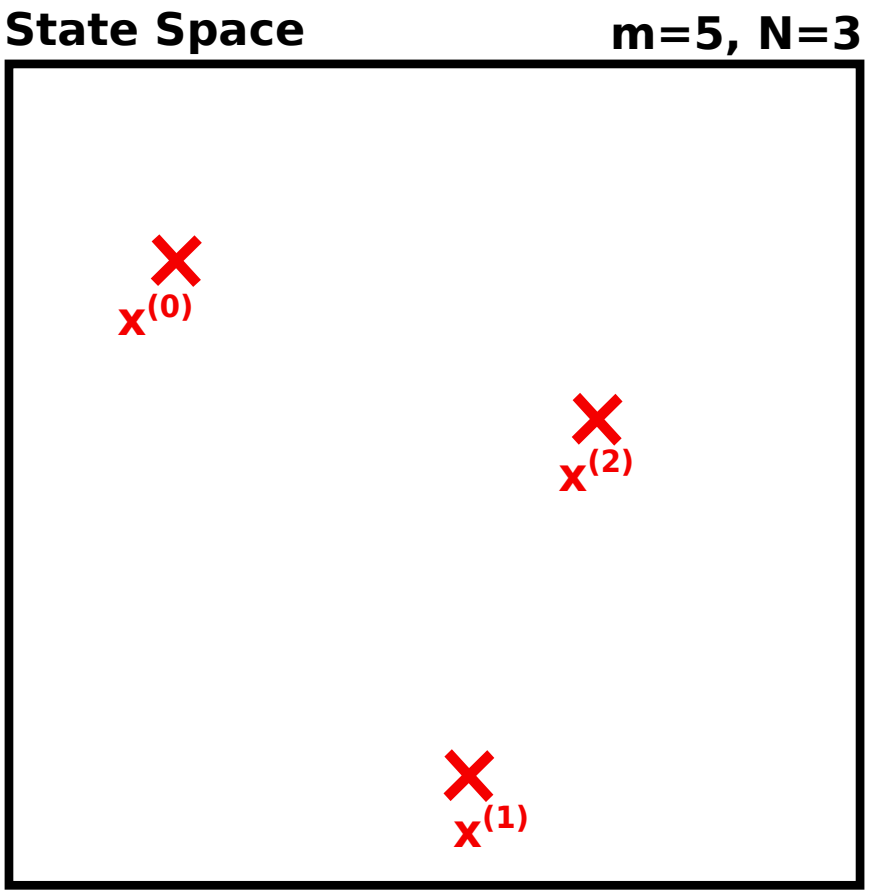
Démonstration.

MDP with 2 states $\{s_1, s_2\}$, 2 actions $\{change, stay\}$, rewards $r(s_1) = 0, r(s_2) = 1$, and deterministic transitions.
$v = (\epsilon, 0)$ and $v' = (0, \epsilon)$ with $\epsilon > 0$.
$\pi = \mathcal{G}v = (st, ch)$ and $\pi' = \mathcal{G}v' = (ch, st)$,

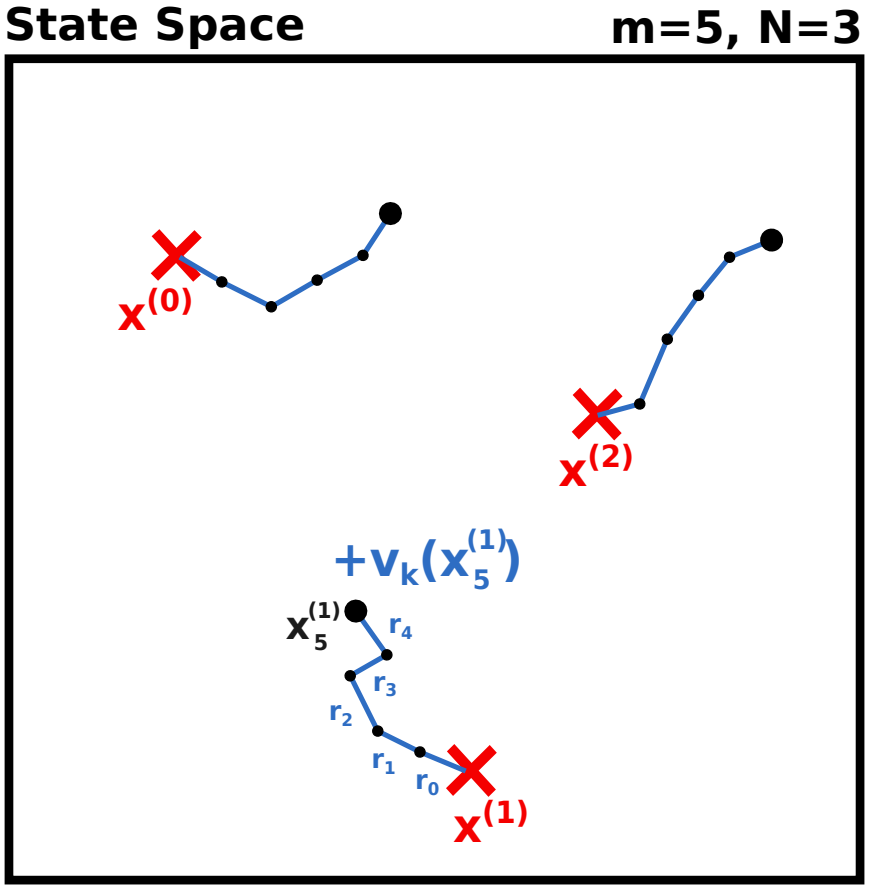
$$(T_\pi)^m v = \begin{pmatrix} \gamma^m \epsilon \\ 1 + \gamma^m \epsilon \end{pmatrix} \text{ and } (T_{\pi'})^m v' = \begin{pmatrix} \frac{\gamma - \gamma^m}{1-\gamma} + \gamma^m \epsilon \\ \frac{1-\gamma^m}{1-\gamma} + \gamma^m \epsilon \end{pmatrix}.$$

$$\Rightarrow (T_{\pi'})^m v' - (T_\pi)^m v = \begin{pmatrix} \frac{\gamma - \gamma^m}{1-\gamma} \\ \frac{\gamma - \gamma^m}{1-\gamma} \end{pmatrix} \text{ while } v' - v = \begin{pmatrix} -\epsilon \\ \epsilon \end{pmatrix}.$$

Since ϵ can be arbitrarily small, the norm of $(T_{\pi'})^m v' - (T_\pi)^m v$ can be arbitrarily larger than the norm of $v - v'$ as long as $m > 1$. □



For $i = 1, \ldots, N$, draw $x^{(i)} \sim \mu$



one generates *a* **rollout** of size m :

$$\left(x^{(i)}, a_0^{(i)}, r_0^{(i)}, x_1^{(i)}, \ldots, a_{m-1}^{(i)}, r_{m-1}^{(i)}, x_m^{(i)}\right) \quad \text{with} \quad a_t^{(i)} = \pi_{k+1}(x_t^{(i)})$$

and estimates $\widehat{v}_{k+1}(x^{(i)}) = \sum_{t=0}^{m-1} \gamma^t r_t^{(i)} + \gamma^m v_k(x_m^{(i)})$ with $E\left[\widehat{v}_{k+1}(x^{(i)})\right] = (T_{\pi_{k+1}})^m v_k$