Results on Fitted Value Iteration

Csaba Szepesvári

Computer and Automation Research Institute of the
Hungarian Academy of Sciences
Kende u. 13-17, Budapest 1111, Hungary
E-mail: **szcsaba@sztaki.hu**

RLAI Papers & Presentations
U. Alberta, 2005

Thanks to: Remi Munos, András Antos

Outline

Outline

Outline

Outline

Problem Setup

Problem Setup:

- Markovian Decision Problems, continuous (or very large) state-spaces
- Generative model ("planning")
- $\Rightarrow$ Value function approximation
- $\Rightarrow$ Approximate Dynamic Programming (ADP)

Main problem:

- **Standard analysis uses L^∞ bounds**
- **Function fitting uses L^2 (L^p) bounds: Hard to get L^∞ guarantees!**

Goal

Finite sample bounds for a practical algorithm (FVI)

Problem Setup

Problem Setup:

- Markovian Decision Problems, continuous (or very large) state-spaces
- Generative model ("planning")
- $\Rightarrow$ Value function approximation
- $\Rightarrow$ Approximate Dynamic Programming (ADP)

Main problem:

- **Standard analysis uses L^∞ bounds**
- **Function fitting uses L^2 (L^p) bounds: Hard to get L^∞ guarantees!**

Goal

Finite sample bounds for a practical algorithm (FVI)

Problem Setup

Problem Setup:

- Markovian Decision Problems, continuous (or very large) state-spaces
- Generative model ("planning")
- $\Rightarrow$ Value function approximation
- $\Rightarrow$ Approximate Dynamic Programming (ADP)

Main problem:

- **Standard analysis uses L^∞ bounds**
- **Function fitting uses L^2 (L^p) bounds: Hard to get L^∞ guarantees!**

Goal

Finite sample bounds for a practical algorithm (FVI)

Problem Setup

Problem Setup:

- Markovian Decision Problems, continuous (or very large) state-spaces
- Generative model ("planning")
- $\Rightarrow$ Value function approximation
- $\Rightarrow$ Approximate Dynamic Programming (ADP)

Main problem:

- **Standard analysis uses L^∞ bounds**
- **Function fitting uses L^2 (L^p) bounds: Hard to get L^∞ guarantees!**

Goal

Finite sample bounds for a practical algorithm (FVI)

Problem Setup

Problem Setup:

- Markovian Decision Problems, continuous (or very large) state-spaces
- Generative model (“planning”)
- $\Rightarrow$ Value function approximation
- $\Rightarrow$ Approximate Dynamic Programming (ADP)

Main problem:

- **Standard analysis uses L^∞ bounds**
- **Function fitting uses L^2 (L^p) bounds: Hard to get L^∞ guarantees!**

Goal

Finite sample bounds for a practical algorithm (FVI)

Problem Setup

Problem Setup:

- Markovian Decision Problems, continuous (or very large) state-spaces
- Generative model ("planning")
- $\Rightarrow$ Value function approximation
- $\Rightarrow$ Approximate Dynamic Programming (ADP)

Main problem:

- **Standard analysis uses L^∞ bounds**
- **Function fitting uses L^2 (L^p) bounds: Hard to get L^∞ guarantees!**

Goal

Finite sample bounds for a practical algorithm (FVI)

Problem Setup

Problem Setup:

- Markovian Decision Problems, continuous (or very large) state-spaces
- Generative model ("planning")
- $\Rightarrow$ Value function approximation
- $\Rightarrow$ Approximate Dynamic Programming (ADP)

Main problem:

- **Standard analysis uses L^∞ bounds**
- **Function fitting uses L^2 (L^p) bounds: Hard to get L^∞ guarantees!**

Goal

Finite sample bounds for a practical algorithm (FVI)

Problem Setup

Problem Setup:

- Markovian Decision Problems, continuous (or very large) state-spaces
- Generative model ("planning")
- $\Rightarrow$ Value function approximation
- $\Rightarrow$ Approximate Dynamic Programming (ADP)

Main problem:

- **Standard analysis uses L^∞ bounds**
- **Function fitting uses L^2 (L^p) bounds: Hard to get L^∞ guarantees!**

Goal

Finite sample bounds for a practical algorithm (FVI)

Outline

Preliminaries – Norms

- Supremum-norm:

$$\|f\|_\infty \stackrel{\text{def}}{=} \sup_{x\in\mathcal{X}} |f(x)|$$

- Space of bounded functions: $B(\mathcal{X})$
- $L^p(\mu)$-norms: μ distribution over $\mathcal{X}$, $p \geq 1$:

$$\|f\|_{p,\mu} \stackrel{\text{def}}{=} \left(\int |f(x)|^p \mu(dx)\right)^{1/p}.$$

- Space of $L^p(\mu)$-norm bounded functions: $L^p(\mathcal{X};\mu)$

Preliminaries – Norms

- Supremum-norm:

$$\|f\|_\infty \stackrel{\text{def}}{=} \sup_{x \in \mathcal{X}} |f(x)|$$

- Space of bounded functions: $B(\mathcal{X})$
- $L^p(\mu)$-norms: μ distribution over $\mathcal{X}$, $p \geq 1$:

$$\|f\|_{p,\mu} \stackrel{\text{def}}{=} \left(\int |f(x)|^p \mu(dx) \right)^{1/p}.$$

- Space of $L^p(\mu)$-norm bounded functions: $L^p(\mathcal{X}; \mu)$

Preliminaries – Norms

- Supremum-norm:

$$\|f\|_\infty \stackrel{\text{def}}{=} \sup_{x \in \mathcal{X}} |f(x)|$$

- Space of bounded functions: $B(\mathcal{X})$
- $L^p(\mu)$-norms: μ distribution over $\mathcal{X}$, $p \geq 1$:

$$\|f\|_{p,\mu} \stackrel{\text{def}}{=} \left(\int |f(x)|^p \mu(dx) \right)^{1/p}.$$

- Space of $L^p(\mu)$-norm bounded functions: $L^p(\mathcal{X};\mu)$

Preliminaries – Norms

- Supremum-norm:

$$\|f\|_\infty \stackrel{\text{def}}{=} \sup_{x\in\mathcal{X}} |f(x)|$$

- Space of bounded functions: $B(\mathcal{X})$
- $L^p(\mu)$-norms: μ distribution over $\mathcal{X}$, $p \geq 1$:

$$\|f\|_{p,\mu} \stackrel{\text{def}}{=} \left(\int |f(x)|^p \mu(dx) \right)^{1/p}.$$

- Space of $L^p(\mu)$-norm bounded functions: $L^p(\mathcal{X};\mu)$

Markovian Decision Problems

$(\mathcal{X}, \mathcal{A}, P, r)$: State space $\mathcal{X}$ ($\subset \mathbb{R}^d$, compact), action space $\mathcal{A}$ (finite), transition probabilities $P(\cdot|x, a)$, reward function $r(x, a)$.

Definitions:

- (stationary) **policy**: a mapping $\pi : \mathcal{X} \to \mathcal{A}$,
- The **value function** V^π defines the performance of a policy π, for example (in the infinite horizon, expected discounted reward case):

$$V^\pi(x) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r(X_t, A_t) | X_0 = x, A_t = \pi(X_t)].$$

Optimal control problem: find (an) optimal policy π^*, i.e., $V^{\pi^*} = \sup_\pi V^\pi$, V^{π^*} written V^*, called the **optimal value function**

Markovian Decision Problems

$(\mathcal{X}, \mathcal{A}, P, r)$: State space $\mathcal{X}$ ($\subset \mathbb{R}^d$, compact), action space $\mathcal{A}$ (finite), transition probabilities $P(\cdot|x,a)$, reward function $r(x,a)$.

Definitions:

- (stationary) **policy**: a mapping $\pi : \mathcal{X} \to \mathcal{A}$,
- The **value function** V^π defines the performance of a policy π, for example (in the infinite horizon, expected discounted reward case):

$$V^\pi(x) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r(X_t, A_t) | X_0 = x, A_t = \pi(X_t)].$$

Optimal control problem: find (an) optimal policy π^*, i.e., $V^{\pi^*} = \sup_\pi V^\pi$, V^{π^*} written V^*, called the **optimal value function**

Markovian Decision Problems

$(\mathcal{X}, \mathcal{A}, P, r)$: State space $\mathcal{X}$ ($\subset \mathbb{R}^d$, compact), action space $\mathcal{A}$ (finite), transition probabilities $P(\cdot|x, a)$, reward function $r(x, a)$.

Definitions:

- (stationary) **policy**: a mapping $\pi : \mathcal{X} \to \mathcal{A}$,
- The **value function** V^π defines the performance of a policy π, for example (in the infinite horizon, expected discounted reward case):

$$V^\pi(x) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r(X_t, A_t) | X_0 = x, A_t = \pi(X_t)].$$

Optimal control problem: find (an) optimal policy π^*, i.e., $V^{\pi^*} = \sup_\pi V^\pi$, V^{π^*} written V^*, called the **optimal value function**

Markovian Decision Problems

$(\mathcal{X}, \mathcal{A}, P, r)$: State space $\mathcal{X}$ ($\subset \mathbb{R}^d$, compact), action space $\mathcal{A}$ (finite), transition probabilities $P(\cdot|x, a)$, reward function $r(x, a)$.

Definitions:

- (stationary) **policy**: a mapping $\pi : \mathcal{X} \to \mathcal{A}$,
- The **value function** V^π defines the performance of a policy π, for example (in the infinite horizon, expected discounted reward case):

$$V^\pi(x) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r(X_t, A_t) | X_0 = x, A_t = \pi(X_t)].$$

Optimal control problem: find (an) optimal policy π^*, i.e., $V^{\pi^*} = \sup_\pi V^\pi$, V^{π^*} written V^*, called the **optimal value function**

Markovian Decision Problems

$(\mathcal{X}, \mathcal{A}, P, r)$: State space $\mathcal{X}$ ($\subset \mathbb{R}^d$, compact), action space $\mathcal{A}$ (finite), transition probabilities $P(\cdot|x, a)$, reward function $r(x, a)$.

Definitions:

- (stationary) **policy**: a mapping $\pi : \mathcal{X} \to \mathcal{A}$,
- The **value function** V^π defines the performance of a policy π, for example (in the infinite horizon, expected discounted reward case):

$$V^\pi(x) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r(X_t, A_t) | X_0 = x, A_t = \pi(X_t)].$$

Optimal control problem: find (an) optimal policy π^*, i.e., $V^{\pi^*} = \sup_\pi V^\pi$, V^{π^*} written V^*, called the **optimal value function**

Dynamic Programming

Proposition: The optimal value function V^* solves the Dynamic Programming (or Bellman) Equation:

$$V^* = TV^*$$

where $T : B(\mathcal{X}) \to B(\mathcal{X})$ is the **Bellman operator**:

$$(TW)(x) \stackrel{\text{def}}{=} \max_{a \in \mathcal{A}} \left\{ r(x,a) + \gamma \int W(y) P(dy|x,a) \right\}.$$

Definition: A policy π is **greedy** w.r.t. $W \in B(\mathcal{X})$ if $\forall x \in \mathcal{X}$,

$$\pi(x) \in \operatorname{argmax}_{a \in \mathcal{A}} \left\{ r(x,a) + \gamma \int W(y) P(dy|x,a) \right\}.$$

Dynamic Programming

Proposition: The optimal value function V^* solves the Dynamic Programming (or Bellman) Equation:

$$V^* = TV^*$$

where $T : B(\mathcal{X}) \to B(\mathcal{X})$ is the **Bellman operator**:

$$(TW)(x) \stackrel{\text{def}}{=} \max_{a \in \mathcal{A}} \left\{ r(x,a) + \gamma \int W(y) P(dy|x,a) \right\}.$$

Definition: A policy π is **greedy** w.r.t. $W \in B(\mathcal{X})$ if $\forall x \in \mathcal{X}$,

$$\pi(x) \in \mathrm{argmax}_{a \in \mathcal{A}} \left\{ r(x,a) + \gamma \int W(y) P(dy|x,a) \right\}.$$

Outline

Value Iteration

- **Property:** T is a contraction mapping in L^∞-norm Banach Fixed Point Theorem $\Rightarrow$ the optimal value function is the unique solution of the DP equation and may be computed by **value iteration**:

$$V_{k+1} = TV_k$$

  with any initial V_0. Then $V_k \to V^*$.
- Problem when $\mathcal{X}$ is large or infinite (e.g. continuous state-space)!
- **Fitted Value Iteration** (Boyan & Moore (1995), Gordon (1995), Tsitsiklis & Van Roy (1996))

$$V_{k+1} = \Pi_{\mathcal{F}} TV_k$$

  where $\Pi_{\mathcal{F}}$ projects iterates into an appropriate function-space.

Value Iteration

- **Property:** T is a contraction mapping in L^∞-norm Banach Fixed Point Theorem $\Rightarrow$ the optimal value function is the unique solution of the DP equation and may be computed by **value iteration**:

$$V_{k+1} = TV_k$$

with any initial V_0. Then $V_k \to V^*$.
- Problem when $\mathcal{X}$ is large or infinite (e.g. continuous state-space)!
- **Fitted Value Iteration** (Boyan & Moore (1995), Gordon (1995), Tsitsiklis & Van Roy (1996))

$$V_{k+1} = \Pi_{\mathcal{F}} TV_k$$

where $\Pi_{\mathcal{F}}$ projects iterates into an appropriate function-space.

Value Iteration

- **Property:** T is a contraction mapping in L^∞-norm Banach Fixed Point Theorem $\Rightarrow$ the optimal value function is the unique solution of the DP equation and may be computed by **value iteration**:

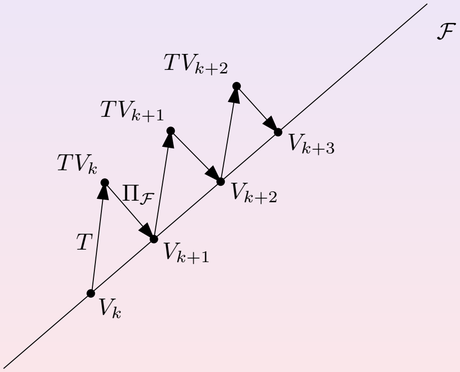
$$V_{k+1} = TV_k$$

with any initial V_0. Then $V_k \to V^*$.
- Problem when $\mathcal{X}$ is large or infinite (e.g. continuous state-space)!
- **Fitted Value Iteration** (Boyan & Moore (1995), Gordon (1995), Tsitsiklis & Van Roy (1996))

$$V_{k+1} = \Pi_{\mathcal{F}} TV_k$$

where $\Pi_{\mathcal{F}}$ projects iterates into an appropriate function-space.

A Graphical View



Sampling Based Fitted Value Iteration (Boyan & Moore (1995))

Input: $\mathcal{F}$ – function space, N, M, K integers, μ – distribution over the state space.

Algorithm (stage k):

1. Sample "basis points": $X_1, \ldots, X_N \in \mathcal{X}$, $X_i \sim \mu$
2. For each action $a \in \mathcal{A}$ and state X_i, sample next states and rewards: $Y_j^{X_i,a} \sim P(\cdot|X_i, a)$, $R_j^{X_i,a} \sim S(\cdot|X_i, a)$, $j = 1, \ldots, M$
3. Calculate the Monte-Carlo approximation of backed up values:
$$v_i = \max_{a \in \mathcal{A}} \frac{1}{M} \sum_{j=1}^{M} \left[R_j^{X_i,a} + \gamma V_k(Y_j^{X_i,a}) \right], \quad i = 1, 2, \ldots, N.$$
4. Solve the least-squares problem: $V_{k+1} = \operatorname{argmin}_{f \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} (f(x_i) - v_i)^2$

Sampling Based Fitted Value Iteration (Boyan & Moore (1995))

Input: $\mathcal{F}$ – function space, N, M, K integers, μ – distribution over the state space.

Algorithm (stage k):

1. Sample "basis points": $X_1, \ldots, X_N \in \mathcal{X}$, $X_i \sim \mu$
2. For each action $a \in \mathcal{A}$ and state X_i, sample next states and rewards: $Y_j^{X_i,a} \sim P(\cdot|X_i, a)$, $R_j^{X_i,a} \sim S(\cdot|X_i, a)$, $j = 1, \ldots, M$
3. Calculate the Monte-Carlo approximation of backed up values:

$$v_i = \max_{a \in \mathcal{A}} \frac{1}{M} \sum_{j=1}^{M} \left[R_j^{X_i,a} + \gamma V_k(Y_j^{X_i,a}) \right], \quad i = 1, 2, \ldots, N.$$

4. Solve the least-squares problem: $V_{k+1} = \mathrm{argmin}_{f \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} (f(x_i) - v_i)^2$

Sampling Based Fitted Value Iteration (Boyan & Moore (1995))

Input: $\mathcal{F}$ – function space, N, M, K integers, μ – distribution over the state space.

Algorithm (stage k):

1. Sample "basis points": $X_1, \ldots, X_N \in \mathcal{X}$, $X_i \sim \mu$
2. For each action $a \in \mathcal{A}$ and state X_i, sample next states and rewards: $Y_j^{X_i,a} \sim P(\cdot|X_i, a)$, $R_j^{X_i,a} \sim S(\cdot|X_i, a)$, $j = 1, \ldots, M$
3. Calculate the Monte-Carlo approximation of backed up values:

$$v_i = \max_{a \in \mathcal{A}} \frac{1}{M} \sum_{j=1}^{M} \left[R_j^{X_i,a} + \gamma V_k(Y_j^{X_i,a}) \right], \quad i = 1, 2, \ldots, N.$$

4. Solve the least-squares problem: $V_{k+1} = \mathrm{argmin}_{f \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} (f(x_i) - v_i)^2$

Sampling Based Fitted Value Iteration (Boyan & Moore (1995))

Input: $\mathcal{F}$ – function space, N, M, K integers, μ – distribution over the state space.

Algorithm (stage k):

1. Sample "basis points": $X_1, \ldots, X_N \in \mathcal{X}$, $X_i \sim \mu$
2. For each action $a \in \mathcal{A}$ and state X_i, sample next states and rewards: $Y_j^{X_i,a} \sim P(\cdot|X_i, a)$, $R_j^{X_i,a} \sim S(\cdot|X_i, a)$, $j = 1, \ldots, M$
3. Calculate the Monte-Carlo approximation of backed up values:
$$v_i = \max_{a\in\mathcal{A}} \frac{1}{M} \sum_{j=1}^{M} \left[R_j^{X_i,a} + \gamma V_k(Y_j^{X_i,a}) \right], \quad i = 1, 2, \ldots, N.$$
4. Solve the least-squares problem: $V_{k+1} = \mathrm{argmin}_{f\in\mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} (f(x_i) - v_i)^2$

Sampling Based Fitted Value Iteration (Boyan & Moore (1995))

Input: $\mathcal{F}$ – function space, N, M, K integers, μ – distribution over the state space.

Algorithm (stage k):

1. Sample "basis points": $X_1, \ldots, X_N \in \mathcal{X}$, $X_i \sim \mu$
2. For each action $a \in \mathcal{A}$ and state X_i, sample next states and rewards: $Y_j^{X_i,a} \sim P(\cdot|X_i, a)$, $R_j^{X_i,a} \sim S(\cdot|X_i, a)$, $j = 1, \ldots, M$
3. Calculate the Monte-Carlo approximation of backed up values:

$$v_i = \max_{a \in \mathcal{A}} \frac{1}{M} \sum_{j=1}^{M} \left[R_j^{X_i,a} + \gamma V_k(Y_j^{X_i,a}) \right], \quad i = 1, 2, \ldots, N.$$

4. Solve the least-squares problem: $V_{k+1} = \mathrm{argmin}_{f \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} (f(x_i) - v_i)^2$

Sampling Based Fitted Value Iteration (Boyan & Moore (1995))

Input: $\mathcal{F}$ – function space, N, M, K integers, μ – distribution over the state space.

Algorithm (stage k):

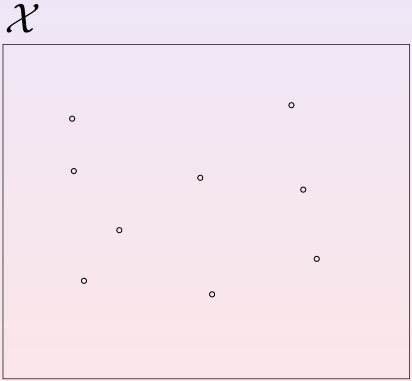
1. Sample "basis points": $X_1, \ldots, X_N \in \mathcal{X}$, $X_i \sim \mu$
2. For each action $a \in \mathcal{A}$ and state X_i, sample next states and rewards: $Y_j^{X_i,a} \sim P(\cdot|X_i, a)$, $R_j^{X_i,a} \sim S(\cdot|X_i, a)$, $j = 1, \ldots, M$
3. Calculate the Monte-Carlo approximation of backed up values:

$$v_i = \max_{a \in \mathcal{A}} \frac{1}{M} \sum_{j=1}^{M} \left[R_j^{X_i,a} + \gamma V_k(Y_j^{X_i,a}) \right], \quad i = 1, 2, \ldots, N.$$

4. Solve the least-squares problem: $V_{k+1} = \mathrm{argmin}_{f \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} (f(x_i) - v_i)^2$

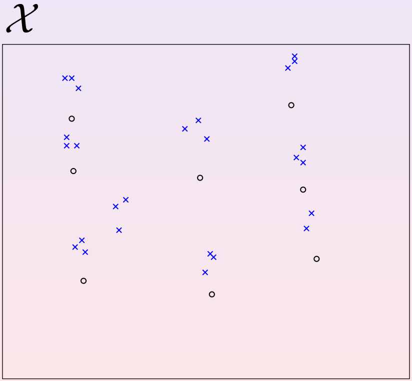
Sampling Based Fitted Value Iteration – Sampling

Navigation problem: $\mathcal{X} = [0,1] \times [0,1]$



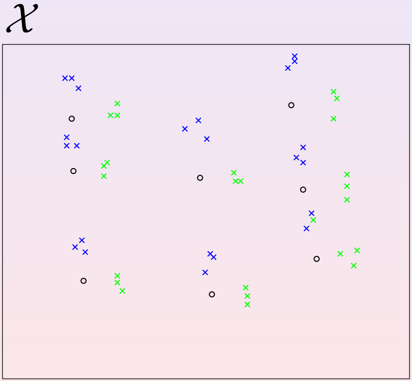
Sampling Based Fitted Value Iteration – Sampling

Navigation problem: $\mathcal{X} = [0,1] \times [0,1]$



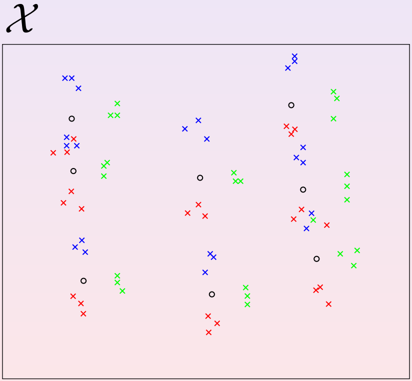
Sampling Based Fitted Value Iteration – Sampling

Navigation problem: $\mathcal{X} = [0,1] \times [0,1]$



Sampling Based Fitted Value Iteration – Sampling

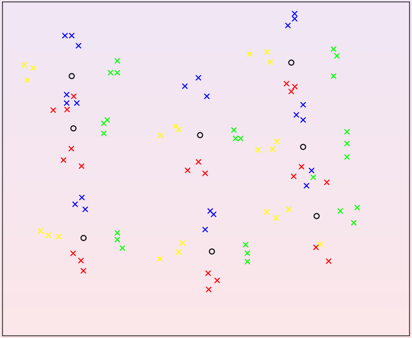
Navigation problem: $\mathcal{X} = [0,1] \times [0,1]$



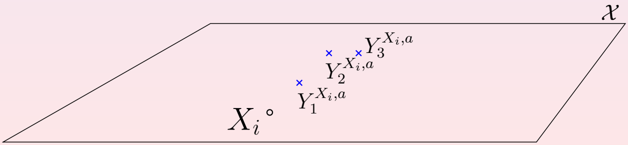
Sampling Based Fitted Value Iteration – Sampling

Navigation problem: $\mathcal{X} = [0,1] \times [0,1]$

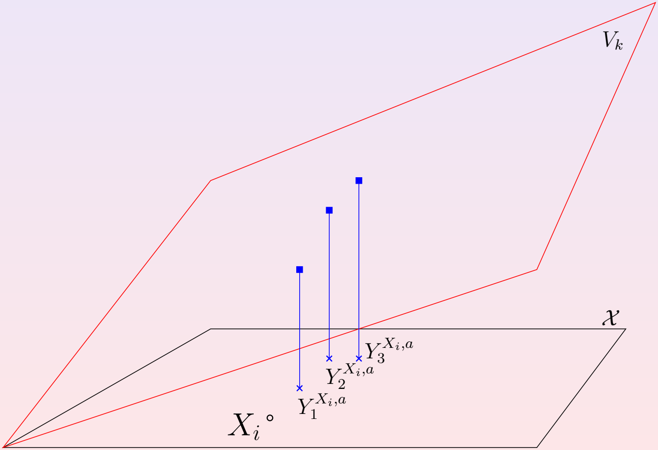
$\mathcal{X}$



Sampling Based Fitted Value Iteration – Computation

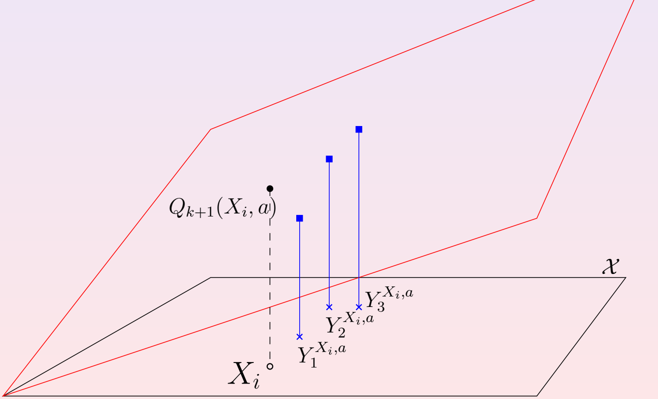


Sampling Based Fitted Value Iteration – Computation



Sampling Based Fitted Value Iteration – Computation

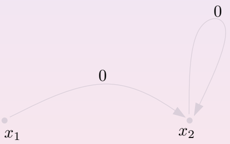
$$\hat{Q}_{k+1}(X_i, a) = R(X_i, a) + \frac{1}{M} \sum_{j=1}^{M} \gamma V_k(Y_j^{X_i,a})$$



Outline

A Minimalist Counterexample (CE – I.)

- Tsitsiklis & Van Roy (1996)
- State space: $\mathcal{X} = \{x_1, x_2\}$
- Dynamics:



- Bellman operator:

$$
\begin{aligned}
(TV)(x_1) &= 0 + \gamma V(x_2) \\
(TV)(x_2) &= 0 + \gamma V(x_2).
\end{aligned}
$$

- Function-space:
$\mathcal{F} = \{\theta\phi \,|\, \theta \in \mathbb{R}\}$,

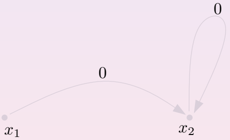
$$\phi(x_1) = 1, \;\; \phi(x_2) = 2.$$

Iteration:

$$
\begin{aligned}
\theta_{t+1} &= \operatorname{argmin}_\theta \|\theta\phi - T(\theta_t\phi)\|_2 \\
&= \operatorname{argmin}_\theta (\theta - \gamma 2\theta_t)^2 + (2\theta - \gamma 2\theta_t)^2 = (6/5\gamma)\theta_t \to +\infty
\end{aligned}
$$

A Minimalist Counterexample (CE – I.)

- Tsitsiklis & Van Roy (1996)
- State space: $\mathcal{X} = \{x_1, x_2\}$
- Dynamics:



- Bellman operator:

$$\begin{aligned}(TV)(x_1) &= 0 + \gamma V(x_2)\\ (TV)(x_2) &= 0 + \gamma V(x_2).\end{aligned}$$

- Function-space: $\mathcal{F} = \{\theta\phi \,|\, \theta \in \mathbb{R}\}$,

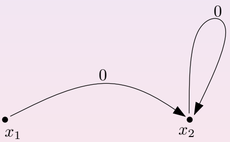
$$\phi(x_1) = 1, \ \phi(x_2) = 2.$$

Iteration:

$$\begin{aligned}\theta_{t+1} &= \mathrm{argmin}_\theta \|\theta\phi - T(\theta_t\phi)\|_2\\ &= \mathrm{argmin}_\theta (\theta - \gamma 2\theta_t)^2 + (2\theta - \gamma 2\theta_t)^2 = (6/5\gamma)\theta_t \to +\infty\end{aligned}$$

A Minimalist Counterexample (CE – I.)

- Tsitsiklis & Van Roy (1996)
- State space: $\mathcal{X} = \{x_1, x_2\}$
- Dynamics:



- Bellman operator:

$$
\begin{aligned}
(TV)(x_1) &= 0 + \gamma V(x_2) \\
(TV)(x_2) &= 0 + \gamma V(x_2).
\end{aligned}
$$

- Function-space:
$\mathcal{F} = \{\theta\phi \,|\, \theta \in \mathbb{R}\}$,

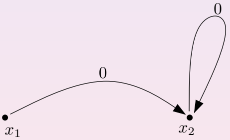
$$\phi(x_1) = 1, \; \phi(x_2) = 2.$$

Iteration:

$$
\begin{aligned}
\theta_{t+1} &= \mathrm{argmin}_\theta \|\theta\phi - T(\theta_t\phi)\|_2 \\
&= \mathrm{argmin}_\theta (\theta - \gamma 2\theta_t)^2 + (2\theta - \gamma 2\theta_t)^2 = (6/5\gamma)\theta_t \to +\infty
\end{aligned}
$$

A Minimalist Counterexample (CE – I.)

- Tsitsiklis & Van Roy (1996)
- State space: $\mathcal{X} = \{x_1, x_2\}$
- Dynamics:



- Bellman operator:

$$\begin{aligned}(TV)(x_1) &= 0 + \gamma V(x_2)\\(TV)(x_2) &= 0 + \gamma V(x_2).\end{aligned}$$

- Function-space:
$\mathcal{F} = \{\theta\phi \,|\, \theta \in \mathbb{R}\}$,

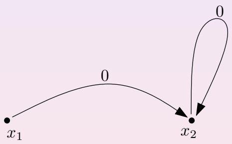
$$\phi(x_1) = 1,\ \phi(x_2) = 2.$$

Iteration:

$$\begin{aligned}\theta_{t+1} &= \mathrm{argmin}_\theta \|\theta\phi - T(\theta_t\phi)\|_2\\&= \mathrm{argmin}_\theta (\theta - \gamma 2\theta_t)^2 + (2\theta - \gamma 2\theta_t)^2 = (6/5\gamma)\theta_t \to +\infty\end{aligned}$$

A Minimalist Counterexample (CE – I.)

- Tsitsiklis & Van Roy (1996)
- State space: $\mathcal{X} = \{x_1, x_2\}$
- Dynamics:



- Bellman operator:

$$\begin{aligned}(TV)(x_1) &= 0 + \gamma V(x_2)\\ (TV)(x_2) &= 0 + \gamma V(x_2).\end{aligned}$$

- Function-space: $\mathcal{F} = \{\theta\phi \,|\, \theta \in \mathbb{R}\}$,

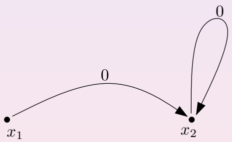
$$\phi(x_1) = 1, \;\; \phi(x_2) = 2.$$

Iteration:

$$\begin{aligned}\theta_{t+1} &= \operatorname{argmin}_\theta \|\theta\phi - T(\theta_t\phi)\|_2\\ &= \operatorname{argmin}_\theta (\theta - \gamma 2\theta_t)^2 + (2\theta - \gamma 2\theta_t)^2 = (6/5\gamma)\theta_t \to +\infty\end{aligned}$$

A Minimalist Counterexample (CE – I.)

- Tsitsiklis & Van Roy (1996)
- State space: $\mathcal{X} = \{x_1, x_2\}$
- Dynamics:



- Bellman operator:

$$
\begin{aligned}
(TV)(x_1) &= 0 + \gamma V(x_2) \\
(TV)(x_2) &= 0 + \gamma V(x_2).
\end{aligned}
$$

- Function-space: $\mathcal{F} = \{\theta\phi \,|\, \theta \in \mathbb{R}\}$,

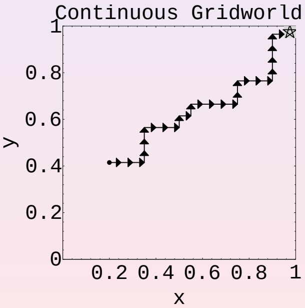
$$\phi(x_1) = 1, \;\; \phi(x_2) = 2.$$

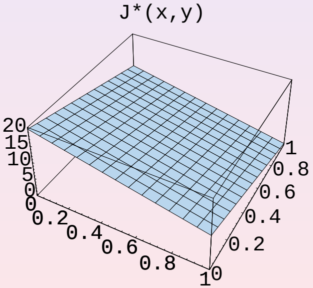
Iteration:

$$
\begin{aligned}
\theta_{t+1} &= \text{argmin}_\theta \|\theta\phi - T(\theta_t\phi)\|_2 \\
&= \text{argmin}_\theta (\theta - \gamma 2\theta_t)^2 + (2\theta - \gamma 2\theta_t)^2 = (6/5\gamma)\theta_t \to +\infty
\end{aligned}
$$

Counterexamples – II/1.[1]

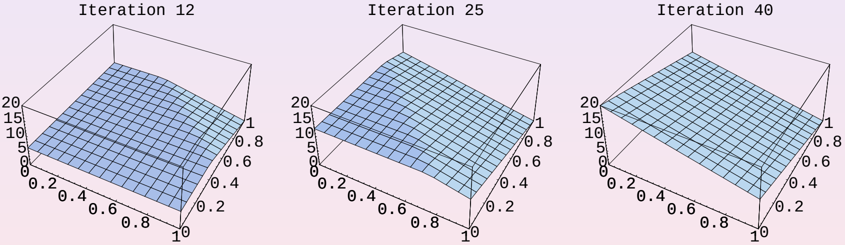
From: Boyan & Moore: "Generalization in Reinforcement Learning: Safely Approximating the Value Function", *NIPS-7*, 1995.





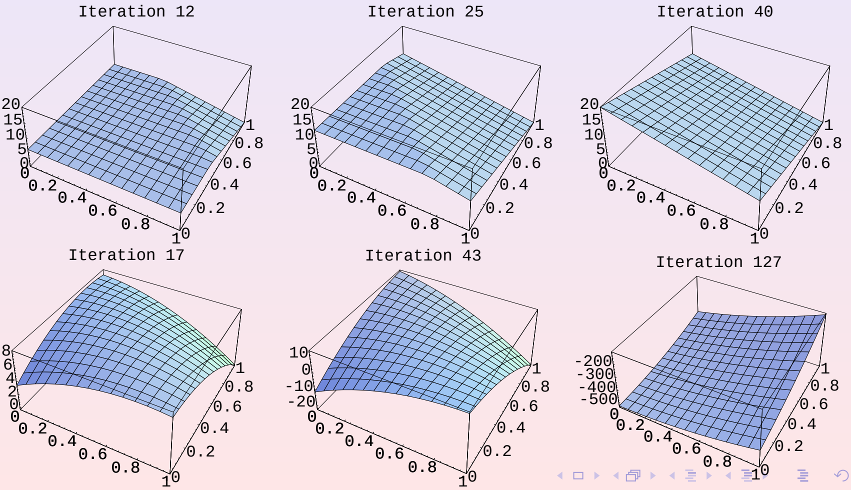
[1] With thanks to Justin Boyan

Counterexamples – II/2.

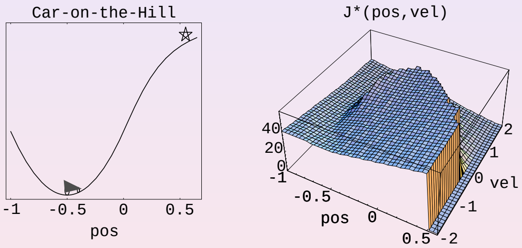


Value Iteration at Work

Counterexamples – II/2.

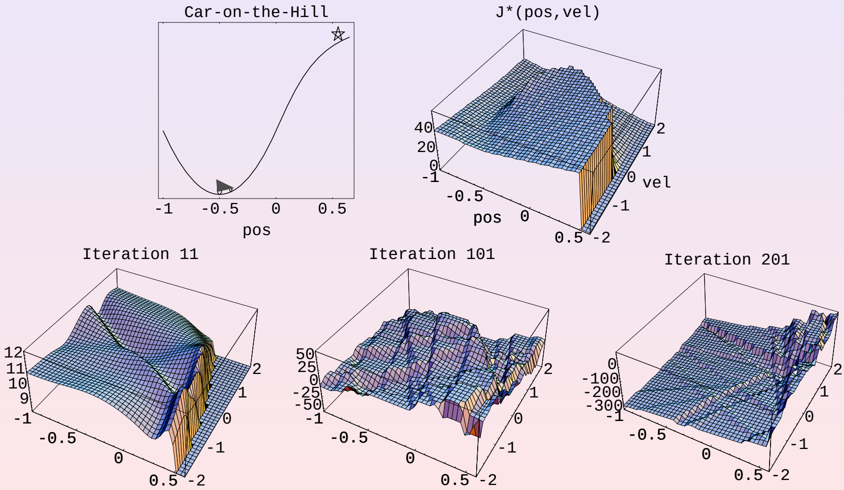


Counterexamples – III.



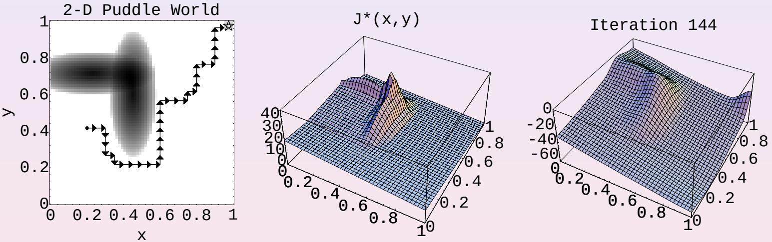
"Car-on-the-hill"

Counterexamples – III.



“Car-on-the-hill”

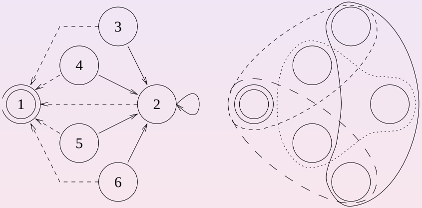
Counterexamples – IV.



“Puddle World”

Counterexamples – V.

G. Gordon: "Stable Function Approximation in Dynamic Programming", *ICML*, 1995.



(a) (b)

Goal state: 1; Markov-process, solid: prob=0.95, dashed=0.05; zero rewards

Summary

- *"In light of these experiments, we conclude that the straightforward combination of DP and function approximation is not robust."* (Boyan & Moore, NIPS-7, 1995)
- *Unfortunately, many popular functions approximators, such as neural nets and linear regression, do not fall in this[2] class (and in fact can diverge).* (G. Gordon, ICML, 1995).

[2] see the slides on "Averagers" below

Summary

- *"In light of these experiments, we conclude that the straightforward combination of DP and function approximation is not robust."* (Boyan & Moore, NIPS-7, 1995)
- *Unfortunately, many popular functions approximators, such as neural nets and linear regression, do not fall in this[2] class (and in fact can diverge).* (G. Gordon, ICML, 1995).

[2] see the slides on "Averagers" below

Outline

Averagers[3]

- Lipschitz-operators:

$$\|Rf - Rg\| \leq \lambda \|f - g\|$$

  - $\lambda = 1$: Non-expansion
  - $\lambda < 1$: Contraction
- Remember: T is a contraction with coefficient γ $\Rightarrow$ Banach-fixed point theorem ensures convergence of $V_{k+1} = TV_k$
- $V_{k+1} = \Pi_{\mathcal{F}} T V_k$: Can $\Pi_{\mathcal{F}} T$ be shown to be a contraction?
- Fact: R λ_R-Lipschitz, S λ_S-Lipschitz $\Rightarrow$ RS $\lambda_R \lambda_S$-Lipschitz.
- Requirement: $\Pi_{\mathcal{F}}$ is a non-expansion ($\Pi_{\mathcal{F}}$ is λ-Lipschitz with $\lambda\gamma < 1$)
- Consequence: V_k is convergent.

[3] G. Gordon, ICML, 1995.

Averagers[3]

- Lipschitz-operators:

$$\|Rf - Rg\| \leq \lambda \|f - g\|$$

  - $\lambda = 1$: Non-expansion
  - $\lambda < 1$: Contraction
- Remember: T is a contraction with coefficient γ $\Rightarrow$ Banach-fixed point theorem ensures convergence of $V_{k+1} = TV_k$
- $V_{k+1} = \Pi_{\mathcal{F}} T V_k$: Can $\Pi_{\mathcal{F}} T$ be shown to be a contraction?
- Fact: R λ_R-Lipschitz, S λ_S-Lipschitz $\Rightarrow$ RS $\lambda_R \lambda_S$-Lipschitz.
- Requirement: $\Pi_{\mathcal{F}}$ is a non-expansion ($\Pi_{\mathcal{F}}$ is λ-Lipschitz with $\lambda\gamma < 1$)
- Consequence: V_k is convergent.

[3] G. Gordon, ICML, 1995.

Averagers[3]

- Lipschitz-operators:

$$\|Rf - Rg\| \leq \lambda \|f - g\|$$

  - $\lambda = 1$: Non-expansion
  - $\lambda < 1$: Contraction
- Remember: T is a contraction with coefficient γ
  $\Rightarrow$ Banach-fixed point theorem ensures convergence of $V_{k+1} = TV_k$
- $V_{k+1} = \Pi_{\mathcal{F}} T V_k$: Can $\Pi_{\mathcal{F}} T$ be shown to be a contraction?
- Fact: R λ_R-Lipschitz, S λ_S-Lipschitz $\Rightarrow$ RS $\lambda_R \lambda_S$-Lipschitz.
- Requirement: $\Pi_{\mathcal{F}}$ is a non-expansion ($\Pi_{\mathcal{F}}$ is λ-Lipschitz with $\lambda\gamma < 1$)
- Consequence: V_k is convergent.

[3] G. Gordon, ICML, 1995.

Averagers[3]

- Lipschitz-operators:

$$\|Rf - Rg\| \leq \lambda \|f - g\|$$

  - $\lambda = 1$: Non-expansion
  - $\lambda < 1$: Contraction
- Remember: T is a contraction with coefficient γ
  $\Rightarrow$ Banach-fixed point theorem ensures convergence of $V_{k+1} = TV_k$
- $V_{k+1} = \Pi_{\mathcal{F}} T V_k$: Can $\Pi_{\mathcal{F}} T$ be shown to be a contraction?
- Fact: R λ_R-Lipschitz, S λ_S-Lipschitz $\Rightarrow$ RS $\lambda_R \lambda_S$-Lipschitz.
- Requirement: $\Pi_{\mathcal{F}}$ is a non-expansion ($\Pi_{\mathcal{F}}$ is λ-Lipschitz with $\lambda\gamma < 1$)
- Consequence: V_k is convergent.

[3] G. Gordon, ICML, 1995.

Averagers[3]

- Lipschitz-operators:

$$\|Rf - Rg\| \leq \lambda \|f - g\|$$

  - $\lambda = 1$: Non-expansion
  - $\lambda < 1$: Contraction
- Remember: T is a contraction with coefficient γ $\Rightarrow$ Banach-fixed point theorem ensures convergence of $V_{k+1} = TV_k$
- $V_{k+1} = \Pi_{\mathcal{F}} TV_k$: Can $\Pi_{\mathcal{F}} T$ be shown to be a contraction?
- Fact: R λ_R-Lipschitz, S λ_S-Lipschitz $\Rightarrow$ RS $\lambda_R \lambda_S$-Lipschitz.
- Requirement: $\Pi_{\mathcal{F}}$ is a non-expansion ($\Pi_{\mathcal{F}}$ is λ-Lipschitz with $\lambda\gamma < 1$)
- Consequence: V_k is convergent.

[3] G. Gordon, ICML, 1995.

Averagers[3]

- Lipschitz-operators:

$$\|Rf - Rg\| \leq \lambda \|f - g\|$$

  - $\lambda = 1$: Non-expansion
  - $\lambda < 1$: Contraction
- Remember: T is a contraction with coefficient γ $\Rightarrow$ Banach-fixed point theorem ensures convergence of $V_{k+1} = TV_k$
- $V_{k+1} = \Pi_{\mathcal{F}} T V_k$: Can $\Pi_{\mathcal{F}} T$ be shown to be a contraction?
- Fact: R λ_R-Lipschitz, S λ_S-Lipschitz $\Rightarrow$ RS $\lambda_R \lambda_S$-Lipschitz.
- Requirement: $\Pi_{\mathcal{F}}$ is a non-expansion ($\Pi_{\mathcal{F}}$ is λ-Lipschitz with $\lambda\gamma < 1$)
- Consequence: V_k is convergent.

[3] G. Gordon, ICML, 1995.

Averagers[3]

- Lipschitz-operators:

$$\|Rf - Rg\| \leq \lambda \|f - g\|$$

  - $\lambda = 1$: Non-expansion
  - $\lambda < 1$: Contraction
- Remember: T is a contraction with coefficient γ $\Rightarrow$ Banach-fixed point theorem ensures convergence of $V_{k+1} = TV_k$
- $V_{k+1} = \Pi_{\mathcal{F}} TV_k$: Can $\Pi_{\mathcal{F}} T$ be shown to be a contraction?
- Fact: R λ_R-Lipschitz, S λ_S-Lipschitz $\Rightarrow$ RS $\lambda_R \lambda_S$-Lipschitz.
- Requirement: $\Pi_{\mathcal{F}}$ is a non-expansion ($\Pi_{\mathcal{F}}$ is λ-Lipschitz with $\lambda\gamma < 1$)
- Consequence: V_k is convergent.

[3] G. Gordon, ICML, 1995.

Averagers[3]

- Lipschitz-operators:

$$\|Rf - Rg\| \leq \lambda \|f - g\|$$

  - $\lambda = 1$: Non-expansion
  - $\lambda < 1$: Contraction
- Remember: T is a contraction with coefficient γ
  $\Rightarrow$ Banach-fixed point theorem ensures convergence of $V_{k+1} = TV_k$
- $V_{k+1} = \Pi_{\mathcal{F}} T V_k$: Can $\Pi_{\mathcal{F}} T$ be shown to be a contraction?
- Fact: R λ_R-Lipschitz, S λ_S-Lipschitz $\Rightarrow$ RS $\lambda_R \lambda_S$-Lipschitz.
- Requirement: $\Pi_{\mathcal{F}}$ is a non-expansion ($\Pi_{\mathcal{F}}$ is λ-Lipschitz with $\lambda\gamma < 1$)
- Consequence: V_k is convergent.

[3] G. Gordon, ICML, 1995.

Averagers[3]

- Lipschitz-operators:

$$\|Rf - Rg\| \leq \lambda \|f - g\|$$

  - $\lambda = 1$: Non-expansion
  - $\lambda < 1$: Contraction
- Remember: T is a contraction with coefficient γ
  $\Rightarrow$ Banach-fixed point theorem ensures convergence of $V_{k+1} = TV_k$
- $V_{k+1} = \Pi_{\mathcal{F}} TV_k$: Can $\Pi_{\mathcal{F}} T$ be shown to be a contraction?
- Fact: R λ_R-Lipschitz, S λ_S-Lipschitz $\Rightarrow$ RS $\lambda_R \lambda_S$-Lipschitz.
- Requirement: $\Pi_{\mathcal{F}}$ is a non-expansion ($\Pi_{\mathcal{F}}$ is λ-Lipschitz with $\lambda\gamma < 1$)
- Consequence: V_k is convergent.

[3] G. Gordon, ICML, 1995.

Averagers

- Equations: Given data $\{(x_i, v_i)\}_{i=1}^n$, f is an *averager* if it has the form:

$$f(x; D) = w_0(x; x_1^n) + \sum_{i=1}^n w_i(x; x_1^n) v_i, \quad x_1^n \stackrel{\text{def}}{=} (x_1, \ldots, x_n),$$

where weights $w_i(x; x_1^n)$ are non-negative and sum to one:

$$\sum_{i=0}^n w_i(x; x_1^n) = 1$$

- Also known as "kernel methods".
- Examples: Kernel averaging (fixed kernel), weighted k-nearest neighbors, Bézier patches, linear interpolation on a triangular (or tetrahedral, etc.) mesh, bilinear interpolation on a square (or cubical, etc.), ...

Averagers

- Equations: Given data $\{(x_i, v_i)\}_{i=1}^n$, f is an *averager* if it has the form:

$$f(x; D) = w_0(x; x_1^n) + \sum_{i=1}^n w_i(x; x_1^n) v_i, \quad x_1^n \stackrel{\text{def}}{=} (x_1, \ldots, x_n),$$

where weights $w_i(x; x_1^n)$ are non-negative and sum to one:

$$\sum_{i=0}^n w_i(x; x_1^n) = 1$$

- Also known as "kernel methods".
- Examples: Kernel averaging (fixed kernel), weighted k-nearest neighbors, Bézier patches, linear interpolation on a triangular (or tetrahedral, etc.) mesh, bilinear interpolation on a square (or cubical, etc.), ...

Averagers

- Equations: Given data $\{(x_i, v_i)\}_{i=1}^n$, f is an *averager* if it has the form:

$$f(x; D) = w_0(x; x_1^n) + \sum_{i=1}^n w_i(x; x_1^n) v_i, \quad x_1^n \stackrel{\text{def}}{=} (x_1, \ldots, x_n),$$

  where weights $w_i(x; x_1^n)$ are non-negative and sum to one:

$$\sum_{i=0}^n w_i(x; x_1^n) = 1$$

- Also known as “kernel methods”.
- Examples: Kernel averaging (fixed kernel), weighted k-nearest neighbors, Bézier patches, linear interpolation on a triangular (or tetrahedral, etc.) mesh, bilinear interpolation on a square (or cubical, etc.), ...

Averagers - II.

- Operator-view:

$$\Pi_{\mathcal{F}} : B(\mathcal{X}) \to B(\mathcal{X}), \quad (\Pi_{\mathcal{F}} V)(x) = f(x; \{(x_i, V(x_i))\}_{i=1}^{n}).$$

- Non-expansion? $\|\Pi_{\mathcal{F}} V - \Pi_{\mathcal{F}} U\|_\infty \leq ?$:

$$\begin{aligned}
|(\Pi_{\mathcal{F}} V)(x) - (\Pi_{\mathcal{F}} U)(x)| &= |\sum_{i=1}^{n} w_i(x; x_1^n)(V(x_i) - U(x_i))| \\
&\leq \sum_{i=1}^{n} w_i(x; x_1^n)|V(x_i) - U(x_i)| \\
&\leq \sup_x |V(x) - U(x)| \sum_{i=1}^{n} w_i(x; x_1^n) \leq \|V - U\|_\infty .
\end{aligned}$$

Averagers - II.

- Operator-view:

$$\Pi_{\mathcal{F}} : B(\mathcal{X}) \to B(\mathcal{X}), \quad (\Pi_{\mathcal{F}} V)(x) = f(x; \{(x_i, V(x_i))\}_{i=1}^n).$$

- Non-expansion? $\|\Pi_{\mathcal{F}} V - \Pi_{\mathcal{F}} U\|_\infty \leq ?$:

$$\begin{aligned} |(\Pi_{\mathcal{F}} V)(x) &- (\Pi_{\mathcal{F}} U)(x)| = |\sum_{i=1}^n w_i(x; x_1^n)(V(x_i) - U(x_i))| \\ &\leq \sum_{i=1}^n w_i(x; x_1^n)|V(x_i) - U(x_i)| \\ &\leq \sup_x |V(x) - U(x)| \sum_{i=1}^n w_i(x; x_1^n) \leq \|V - U\|_\infty. \end{aligned}$$

Averagers - II.

- Operator-view:

$$\Pi_{\mathcal{F}} : B(\mathcal{X}) \to B(\mathcal{X}), \quad (\Pi_{\mathcal{F}} V)(x) = f(x; \{(x_i, V(x_i))\}_{i=1}^n).$$

- Non-expansion? $\|\Pi_{\mathcal{F}} V - \Pi_{\mathcal{F}} U\|_\infty \leq ?$:

$$\begin{aligned} |(\Pi_{\mathcal{F}} V)(x) &- (\Pi_{\mathcal{F}} U)(x)| = |\sum_{i=1}^n w_i(x; x_1^n)(V(x_i) - U(x_i))| \\ &\leq \sum_{i=1}^n w_i(x; x_1^n)|V(x_i) - U(x_i)| \\ &\leq \sup_x |V(x) - U(x)| \sum_{i=1}^n w_i(x; x_1^n) \leq \|V - U\|_\infty. \end{aligned}$$

Converging to what?

- $V_{k+1} = \Pi_{\mathcal{F}} T V_k$
- $U_k = TV_k$: $U_{k+1} = TV_{k+1} = T\Pi_{\mathcal{F}} TV_k = T\Pi_{\mathcal{F}} U_k$
- Assume $U_k \to U^*$. Then:

$$\|U^* - V^*\| = \|T\Pi_{\mathcal{F}} U^* - TV^*\| \le \gamma \|\Pi_{\mathcal{F}} U^* - V^*\|$$

$$\|\Pi_{\mathcal{F}} U^* - V^*\| \le \|\Pi_{\mathcal{F}} U^* - \Pi_{\mathcal{F}} V^*\| + \|\Pi_{\mathcal{F}} V^* - V^*\|$$

$$(1-\gamma)\|U^* - V^*\| \le \gamma \|\Pi_{\mathcal{F}} V^* - V^*\|$$

$$\|U^* - V^*\| \le \frac{\gamma}{1-\gamma} \|\Pi_{\mathcal{F}} V^* - V^*\|$$

Converging to what?

- $V_{k+1} = \Pi_{\mathcal{F}} T V_k$
- $U_k = T V_k$: $U_{k+1} = T V_{k+1} = T \Pi_{\mathcal{F}} T V_k = T \Pi_{\mathcal{F}} U_k$
- Assume $U_k \to U^*$. Then:

$$\|U^* - V^*\| = \|T\Pi_{\mathcal{F}} U^* - TV^*\| \leq \gamma \|\Pi_{\mathcal{F}} U^* - V^*\|$$

$$\|\Pi_{\mathcal{F}} U^* - V^*\| \leq \|\Pi_{\mathcal{F}} U^* - \Pi_{\mathcal{F}} V^*\| + \|\Pi_{\mathcal{F}} V^* - V^*\|$$

$$(1-\gamma)\|U^* - V^*\| \leq \gamma \|\Pi_{\mathcal{F}} V^* - V^*\|$$

$$\|U^* - V^*\| \leq \frac{\gamma}{1-\gamma} \|\Pi_{\mathcal{F}} V^* - V^*\|$$

Converging to what?

- $V_{k+1} = \Pi_{\mathcal{F}} T V_k$
- $U_k = T V_k$: $U_{k+1} = T V_{k+1} = T \Pi_{\mathcal{F}} T V_k = T \Pi_{\mathcal{F}} U_k$
- Assume $U_k \to U^*$. Then:

$$\|U^* - V^*\| = \|T\Pi_{\mathcal{F}} U^* - TV^*\| \le \gamma \|\Pi_{\mathcal{F}} U^* - V^*\|$$

$$\|\Pi_{\mathcal{F}} U^* - V^*\| \le \|\Pi_{\mathcal{F}} U^* - \Pi_{\mathcal{F}} V^*\| + \|\Pi_{\mathcal{F}} V^* - V^*\|$$

$$(1-\gamma)\|U^* - V^*\| \le \gamma \|\Pi_{\mathcal{F}} V^* - V^*\|$$

$$\|U^* - V^*\| \le \frac{\gamma}{1-\gamma} \|\Pi_{\mathcal{F}} V^* - V^*\|$$

Converging to what?

- $V_{k+1} = \Pi_{\mathcal{F}} T V_k$
- $U_k = T V_k$: $U_{k+1} = T V_{k+1} = T \Pi_{\mathcal{F}} T V_k = T \Pi_{\mathcal{F}} U_k$
- Assume $U_k \to U^*$. Then:

$$\|U^* - V^*\| = \|T \Pi_{\mathcal{F}} U^* - T V^*\| \le \gamma \|\Pi_{\mathcal{F}} U^* - V^*\|$$

$$\|\Pi_{\mathcal{F}} U^* - V^*\| \le \|\Pi_{\mathcal{F}} U^* - \Pi_{\mathcal{F}} V^*\| + \|\Pi_{\mathcal{F}} V^* - V^*\|$$

$$(1-\gamma)\|U^* - V^*\| \le \gamma \|\Pi_{\mathcal{F}} V^* - V^*\|$$

$$\|U^* - V^*\| \le \frac{\gamma}{1-\gamma} \|\Pi_{\mathcal{F}} V^* - V^*\|$$

Converging to what?

- $V_{k+1} = \Pi_{\mathcal{F}} T V_k$
- $U_k = T V_k$: $U_{k+1} = T V_{k+1} = T \Pi_{\mathcal{F}} T V_k = T \Pi_{\mathcal{F}} U_k$
- Assume $U_k \to U^*$. Then:

$$\|U^* - V^*\| = \|T \Pi_{\mathcal{F}} U^* - T V^*\| \leq \gamma \|\Pi_{\mathcal{F}} U^* - V^*\|$$

$$\|\Pi_{\mathcal{F}} U^* - V^*\| \leq \|\Pi_{\mathcal{F}} U^* - \Pi_{\mathcal{F}} V^*\| + \|\Pi_{\mathcal{F}} V^* - V^*\|$$

$$(1-\gamma)\|U^* - V^*\| \leq \gamma \|\Pi_{\mathcal{F}} V^* - V^*\|$$

$$\|U^* - V^*\| \leq \frac{\gamma}{1-\gamma} \|\Pi_{\mathcal{F}} V^* - V^*\|$$

Converging to what?

- $V_{k+1} = \Pi_{\mathcal{F}} T V_k$
- $U_k = T V_k$: $U_{k+1} = T V_{k+1} = T \Pi_{\mathcal{F}} T V_k = T \Pi_{\mathcal{F}} U_k$
- Assume $U_k \to U^*$. Then:

$$\|U^* - V^*\| = \|T\Pi_{\mathcal{F}} U^* - TV^*\| \le \gamma \|\Pi_{\mathcal{F}} U^* - V^*\|$$

$$\|\Pi_{\mathcal{F}} U^* - V^*\| \le \|\Pi_{\mathcal{F}} U^* - \Pi_{\mathcal{F}} V^*\| + \|\Pi_{\mathcal{F}} V^* - V^*\|$$

$$(1-\gamma)\|U^* - V^*\| \le \gamma \|\Pi_{\mathcal{F}} V^* - V^*\|$$

$$\|U^* - V^*\| \le \frac{\gamma}{1-\gamma} \|\Pi_{\mathcal{F}} V^* - V^*\|$$

Converging to what?

- $V_{k+1} = \Pi_{\mathcal{F}} T V_k$
- $U_k = TV_k$: $U_{k+1} = TV_{k+1} = T\Pi_{\mathcal{F}} TV_k = T\Pi_{\mathcal{F}} U_k$
- Assume $U_k \to U^*$. Then:

$$\|U^* - V^*\| = \|T\Pi_{\mathcal{F}} U^* - TV^*\| \le \gamma \|\Pi_{\mathcal{F}} U^* - V^*\|$$

$$\|\Pi_{\mathcal{F}} U^* - V^*\| \le \|\Pi_{\mathcal{F}} U^* - \Pi_{\mathcal{F}} V^*\| + \|\Pi_{\mathcal{F}} V^* - V^*\|$$

$$(1-\gamma)\|U^* - V^*\| \le \gamma \|\Pi_{\mathcal{F}} V^* - V^*\|$$

$$\|U^* - V^*\| \le \frac{\gamma}{1-\gamma} \|\Pi_{\mathcal{F}} V^* - V^*\|$$

Stability

Fitted value iteration is a special case of *approximate value iteration*:

$$V_{k+1} = TV_k + \epsilon_k.$$

L^∞-stability Theorem [Bertsekas & Tsitsiklis, 1996]: Let π_k be the greedy policy w.r.t. V_k. Then

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

Stability: By making $\sup_k \|\epsilon_k\|_\infty$ small, we can make $\limsup_{k\to\infty} \|V^* - V^{\pi_k}\|_\infty$ small.

Stability

Fitted value iteration is a special case of *approximate value iteration*:

$$V_{k+1} = TV_k + \epsilon_k.$$

L^∞-stability Theorem [Bertsekas & Tsitsiklis, 1996]: Let π_k be the greedy policy w.r.t. V_k. Then

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

Stability: By making $\sup_k \|\epsilon_k\|_\infty$ small, we can make $\limsup_{k\to\infty} \|V^* - V^{\pi_k}\|_\infty$ small.

Stability

Fitted value iteration is a special case of *approximate value iteration*:

$$V_{k+1} = TV_k + \epsilon_k.$$

L^∞-stability Theorem [Bertsekas & Tsitsiklis, 1996]: Let π_k be the greedy policy w.r.t. V_k. Then

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

Stability: By making $\sup_k \|\epsilon_k\|_\infty$ small, we can make $\limsup_{k\to\infty} \|V^* - V^{\pi_k}\|_\infty$ small.

Is there Life After Averagers?

- F. A. Longstaff, E. S. Shwartz: “Valuing american options by simulation: A simple least-squares approach”, *Rev. Financial Studies*, 14(1):113–147, 2001.
- M. Haugh: “Duality theory and simulation in financial engineering”, *Proceedings of the Winter Simulation Conference*, pp. 327–334, 2003.
- T. Jung and T. Uthmann: “Experiments in value function approximation with sparse support vector regression” *ECML-2004*, 2004.
- M. Riedmiller: “Neural fitted Q iteration – first experiences with a data efficient neural reinforcement learning method”, *ECML-2005*, 2005.

Is there Life After Averagers?

- F. A. Longstaff, E. S. Shwartz: “Valuing american options by simulation: A simple least-squares approach”, *Rev. Financial Studies*, 14(1):113–147, 2001.
- M. Haugh: “Duality theory and simulation in financial engineering”, *Proceedings of the Winter Simulation Conference*, pp. 327–334, 2003.
- T. Jung and T. Uthmann: “Experiments in value function approximation with sparse support vector regression” *ECML-2004*, 2004.
- M. Riedmiller: “Neural fitted Q iteration – first experiences with a data efficient neural reinforcement learning method”, *ECML-2005*, 2005.

Is there Life After Averagers?

- F. A. Longstaff, E. S. Shwartz: “Valuing american options by simulation: A simple least-squares approach”, *Rev. Financial Studies*, 14(1):113–147, 2001.
- M. Haugh: “Duality theory and simulation in financial engineering”, *Proceedings of the Winter Simulation Conference*, pp. 327–334, 2003.
- T. Jung and T. Uthmann: “Experiments in value function approximation with sparse support vector regression” *ECML-2004*, 2004.
- M. Riedmiller: “Neural fitted Q iteration – first experiences with a data efficient neural reinforcement learning method”, *ECML-2005*, 2005.

Is there Life After Averagers?

- F. A. Longstaff, E. S. Shwartz: “Valuing american options by simulation: A simple least-squares approach”, *Rev. Financial Studies*, 14(1):113–147, 2001.
- M. Haugh: “Duality theory and simulation in financial engineering”, *Proceedings of the Winter Simulation Conference*, pp. 327–334, 2003.
- T. Jung and T. Uthmann: “Experiments in value function approximation with sparse support vector regression” *ECML-2004*, 2004.
- M. Riedmiller: “Neural fitted Q iteration – first experiences with a data efficient neural reinforcement learning method”, *ECML-2005*, 2005.

Issues

Problem # 1: Non-averagers

Many of the previous papers do not use averagers, but use fitted value iteration with some linear or non-linear function approximator – minimizing least-square error. Can such methods guaranteed to work?

Problem # 2: Sampling

All of the previous papers do some sort of sampling instead of exact computation.

- Can we show that least-square fitting "works"?
- What is the effect of using sampling? Is FVI robust against sampling errors?

Issues

Problem # 1: Non-averagers

Many of the previous papers do not use averagers, but use fitted value iteration with some linear or non-linear function approximator – minimizing least-square error. Can such methods guaranteed to work?

Problem # 2: Sampling

All of the previous papers do some sort of sampling instead of exact computation.

- Can we show that least-square fitting "works"?
- What is the effect of using sampling? Is FVI robust against sampling errors?

Issues

Problem # 1: Non-averagers

Many of the previous papers do not use averagers, but use fitted value iteration with some linear or non-linear function approximator – minimizing least-square error. Can such methods guaranteed to work?

Problem # 2: Sampling

All of the previous papers do some sort of sampling instead of exact computation.

- Can we show that least-square fitting "works"?
- What is the effect of using sampling? Is FVI robust against sampling errors?

Issues

Problem # 1: Non-averagers

Many of the previous papers do not use averagers, but use fitted value iteration with some linear or non-linear function approximator – minimizing least-square error. Can such methods guaranteed to work?

Problem # 2: Sampling

All of the previous papers do some sort of sampling instead of exact computation.

- Can we show that least-square fitting "works"?
- What is the effect of using sampling? Is FVI robust against sampling errors?

Why the Restriction to Using Averagers is not Entirely Satisfactory?

- Sometimes non-averagers are used in practice with success
- Sup-norm stability is not satisfactory:
  - Value-function iterators might have high derivatives (or even be discontinuous)
  - Guaranteing uniformly small errors over the state-space is
    - Cannot control spatial distribution of errors
    - Computationally challenging
    - Too demanding
    - Might not work for algorithms that optimize for least-square error

Why the Restriction to Using Averagers is not Entirely Satisfactory?

- Sometimes non-averagers are used in practice with success
- Sup-norm stability is not satisfactory:
  - Value-function iterators might have high derivatives (or even be discontinuous)
  - Guaranteing uniformly small errors over the state-space is
    - Cannot control spatial distribution of errors
    - Computationally challenging
    - Too demanding
    - Might not work for algorithms that optimize for least-square error

Why the Restriction to Using Averagers is not Entirely Satisfactory?

- Sometimes non-averagers are used in practice with success
- Sup-norm stability is not satisfactory:
  - Value-function iterators might have high derivatives (or even be discontinuous)
  - Guaranteing uniformly small errors over the state-space is
    - Cannot control spatial distribution of errors
    - Computationally challenging
    - Too demanding
    - Might not work for algorithms that optimize for least-square error

Why the Restriction to Using Averagers is not Entirely Satisfactory?

- Sometimes non-averagers are used in practice with success
- Sup-norm stability is not satisfactory:
  - Value-function iterators might have high derivatives (or even be discontinuous)
  - Guaranteing uniformly small errors over the state-space is
    - Cannot control spatial distribution of errors
    - Computationally challenging
    - Too demanding
    - Might not work for algorithms that optimize for least-square error

Why the Restriction to Using Averagers is not Entirely Satisfactory?

- Sometimes non-averagers are used in practice with success
- Sup-norm stability is not satisfactory:
  - Value-function iterators might have high derivatives (or even be discontinuous)
  - Guaranteing uniformly small errors over the state-space is
    - Cannot control spatial distribution of errors
    - Computationally challenging
    - Too demanding
    - Might not work for algorithms that optimize for least-square error

Why the Restriction to Using Averagers is not Entirely Satisfactory?

- Sometimes non-averagers are used in practice with success
- Sup-norm stability is not satisfactory:
  - Value-function iterators might have high derivatives (or even be discontinuous)
  - Guaranteing uniformly small errors over the state-space is
    - Cannot control spatial distribution of errors
    - Computationally challenging
    - Too demanding
    - Might not work for algorithms that optimize for least-square error

Why the Restriction to Using Averagers is not Entirely Satisfactory?

- Sometimes non-averagers are used in practice with success
- Sup-norm stability is not satisfactory:
  - Value-function iterators might have high derivatives (or even be discontinuous)
  - Guaranteing uniformly small errors over the state-space is
    - Cannot control spatial distribution of errors
    - Computationally challenging
    - Too demanding
    - Might not work for algorithms that optimize for least-square error

Why the Restriction to Using Averagers is not Entirely Satisfactory?

- Sometimes non-averagers are used in practice with success
- Sup-norm stability is not satisfactory:
  - Value-function iterators might have high derivatives (or even be discontinuous)
  - Guaranteing uniformly small errors over the state-space is
    - Cannot control spatial distribution of errors
    - Computationally challenging
    - Too demanding
    - Might not work for algorithms that optimize for least-square error

Sampling Based Fitted Value Iteration

Input: $\mathcal{F}$ – function space, N, M, K integers, μ – distribution over the state space.

Algorithm (stage k):

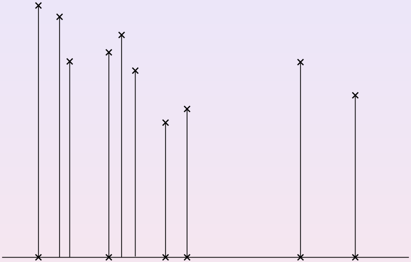
1. Sample "basis points": $X_1, \ldots, X_N \in \mathcal{X}$, $X_i \sim \mu$
2. For each action $a \in \mathcal{A}$ and state X_i, sample next states and rewards: $Y_j^{X_i,a} \sim P(\cdot|X_i, a)$, $R_j^{X_i,a} \sim S(\cdot|X_i, a)$, $j = 1, \ldots, M$
3. Calculate the Monte-Carlo approximation of backed up values:

$$v_i = \max_{a \in \mathcal{A}} \frac{1}{M} \sum_{j=1}^{M} \left[R_j^{X_i,a} + \gamma V_k(Y_j^{X_i,a}) \right], \quad i = 1, 2, \ldots, N.$$

4. Solve the least-squares problem: $V_{k+1} = \mathrm{argmin}_{f \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} (f(x_i) - v_i)^2$

Outline

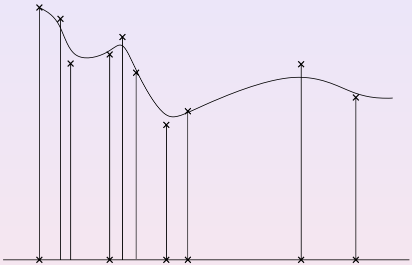
Regression



A Question of Capacity

- If $\mathcal{F}$ has high-capacity: It can fit nearly everything – suspicious to “overtraining”
- If $\mathcal{F}$ has low-capacity: It does more smoothing of data – cannot fit well all kind of data
- Dilemma: How much capacity to allow?

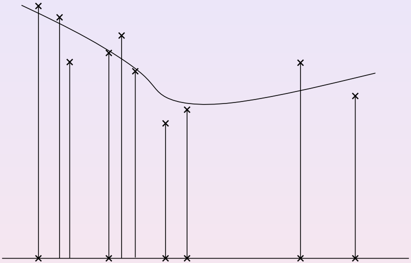
Regression



A Question of Capacity

- If $\mathcal{F}$ has high-capacity: It can fit nearly everything – suspicious to “overtraining”
- If $\mathcal{F}$ has low-capacity: It does more smoothing of data – cannot fit well all kind of data
- Dilemma: How much capacity to allow?

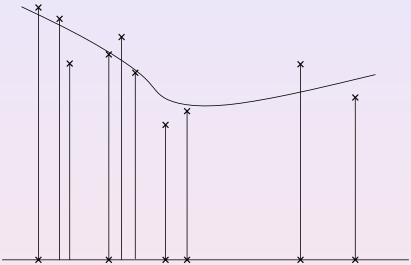
Regression



A Question of Capacity

- If $\mathcal{F}$ has high-capacity: It can fit nearly everything – suspicious to "overtraining"
- If $\mathcal{F}$ has low-capacity: It does more smoothing of data – cannot fit well all kind of data
- Dilemma: How much capacity to allow?

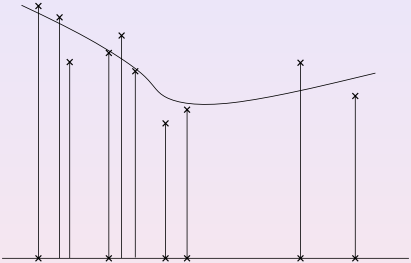
Regression



A Question of Capacity

- If $\mathcal{F}$ has high-capacity: It can fit nearly everything – suspicious to “overtraining”
- If $\mathcal{F}$ has low-capacity: It does more smoothing of data – cannot fit well all kind of data
- Dilemma: How much capacity to allow?

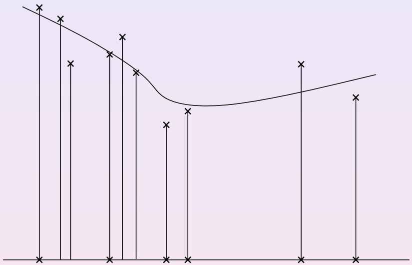
Regression



A Question of Capacity

- If $\mathcal{F}$ has high-capacity: It can fit nearly everything – suspicious to “overtraining”
- If $\mathcal{F}$ has low-capacity: It does more smoothing of data – cannot fit well all kind of data
- Dilemma: How much capacity to allow?

Regression



A Question of Capacity

- If $\mathcal{F}$ has high-capacity: It can fit nearly everything – suspicious to "overtraining"
- If $\mathcal{F}$ has low-capacity: It does more smoothing of data – cannot fit well all kind of data
- Dilemma: How much capacity to allow?

Approximating Expected Values by Averages

Theorem (Pollard, 1984)

Let X_i, $i = 1, \ldots, n$ be i.i.d., $\mathcal{F}$ a space of uniformly bounded measurable functions, with common bound K. Then

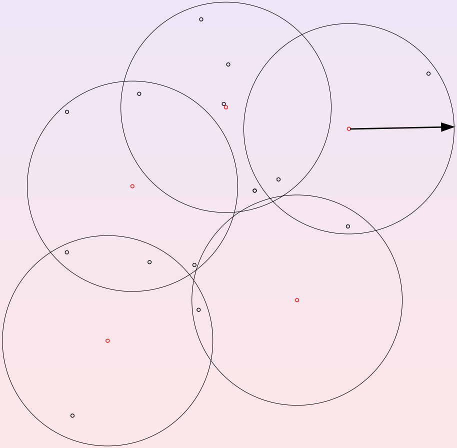
$$\mathbb{P}\left(\sup_{f\in\mathcal{F}}\left|\frac{1}{n}\sum_{i=1}^{n} f(X_i) - \mathbb{E}f(X_1)\right| > \epsilon\right) \le 8e^{-\frac{n\epsilon^2}{128K^2}}\mathbb{E}\mathcal{N}(\epsilon/8, \mathcal{F}(X^{1:n})),$$

where $\mathcal{N}(\epsilon, \mathcal{F}(X^{1:n}))$ is the smallest natural number m such that

$$\mathcal{F}(x^{1:n}) = \{(f(x_1), \ldots, f(x_n)) \,|\, f \in \mathcal{F}\},$$

can be covered in $(\mathbb{R}^n, \ell^1)$ by m spheres, centered at $\mathcal{F}(x^{1:n})$ and with a radius of at most $r = n\epsilon$.

Covering Numbers



$$\mathcal{F}(x^{1:n}) = \{(f(x_1),\ldots,f(x_n)) \mid f \in \mathcal{F}\}$$

Covering Numbers – Examples

- Non-parametric regression:[4] $\mathcal{N}(\epsilon, \mathcal{F}(X^{1:n})) \sim n$.
- Finitely parameterized function classes:
  - Linear fapps: $\mathcal{F} = \{\theta^T \phi \mid \|\theta\| \le B\}$
  - Neural nets: $\mathcal{F} = \{s(As(B \cdot + b) + a) \mid \|A\|, \|B\|, \|a\|, \|b\| \le B\}$
- For these, $\mathcal{N}(\epsilon, \mathcal{F}(X^{1:n}))$ scales with $O(d[\log d])$[5] [6]

[4] L. Devroye and L. Györfi and G. Lugosi: *“A Probabilistic Theory of Pattern Recognition”*, Springer, 1996.

[5] T. Zhang: “Covering Number Bounds of Certain Regularized Linear Function Classes”, *JMLR* 2:527–550, 2002

[6] M. Anthony and P.L. Bartlett: *“Neural Network Learning: Theoretical Foundations”*, Cambridge Univ. Press, UK 1999.

Covering Numbers – Examples

- Non-parametric regression:[4] $\mathcal{N}(\epsilon, \mathcal{F}(X^{1:n})) \sim n$.
- Finitely parameterized function classes:
  - Linear fapps: $\mathcal{F} = \{\theta^T \phi \,|\, \|\theta\| \le B\}$
  - Neural nets: $\mathcal{F} = \{s(As(B \cdot + b) + a) \,|\, \|A\|, \|B\|, \|a\|, \|b\| \le B\}$
- For these, $\mathcal{N}(\epsilon, \mathcal{F}(X^{1:n}))$ scales with $O(d[\log d])$[5] [6]

[4] L. Devroye and L. Györfi and G. Lugosi: *“A Probabilistic Theory of Pattern Recognition”*, Springer, 1996.

[5] T. Zhang: “Covering Number Bounds of Certain Regularized Linear Function Classes”, *JMLR* 2:527–550, 2002

[6] M. Anthony and P.L. Bartlett: *“Neural Network Learning: Theoretical Foundations”*, Cambridge Univ. Press, UK 1999.

Covering Numbers – Examples

- Non-parametric regression:[4] $\mathcal{N}(\epsilon, \mathcal{F}(X^{1:n})) \sim n$.
- Finitely parameterized function classes:
  - Linear fapps: $\mathcal{F} = \{\theta^T \phi \mid \|\theta\| \le B\}$
  - Neural nets: $\mathcal{F} = \{s(As(B \cdot + b) + a) \mid \|A\|, \|B\|, \|a\|, \|b\| \le B\}$
- For these, $\mathcal{N}(\epsilon, \mathcal{F}(X^{1:n}))$ scales with $O(d[\log d])$[5] [6]

---

[4] L. Devroye and L. Györfi and G. Lugosi: *"A Probabilistic Theory of Pattern Recognition"*, Springer, 1996.

[5] T. Zhang: "Covering Number Bounds of Certain Regularized Linear Function Classes", *JMLR* 2:527–550, 2002

[6] M. Anthony and P.L. Bartlett: *"Neural Network Learning: Theoretical Foundations"*, Cambridge Univ. Press, UK 1999.

Covering Numbers – Examples

- Non-parametric regression:[4] $\mathcal{N}(\epsilon, \mathcal{F}(X^{1:n})) \sim n$.
- Finitely parameterized function classes:
  - Linear fapps: $\mathcal{F} = \{\theta^T \phi \mid \|\theta\| \leq B\}$
  - Neural nets: $\mathcal{F} = \{s(As(B \cdot + b) + a) \mid \|A\|, \|B\|, \|a\|, \|b\| \leq B\}$
- For these, $\mathcal{N}(\epsilon, \mathcal{F}(X^{1:n}))$ scales with $O(d[\log d])$[5] [6]

[4] L. Devroye and L. Györfi and G. Lugosi: *"A Probabilistic Theory of Pattern Recognition"*, Springer, 1996.

[5] T. Zhang: "Covering Number Bounds of Certain Regularized Linear Function Classes", *JMLR* 2:527–550, 2002

[6] M. Anthony and P.L. Bartlett: *"Neural Network Learning: Theoretical Foundations"*, Cambridge Univ. Press, UK 1999.

Covering Numbers – Examples

- Non-parametric regression:[4] $\mathcal{N}(\epsilon, \mathcal{F}(X^{1:n})) \sim n$.
- Finitely parameterized function classes:
  - Linear fapps: $\mathcal{F} = \{\theta^T \phi \mid \|\theta\| \le B\}$
  - Neural nets:
    $\mathcal{F} = \{s(As(B\cdot + b) + a) \mid \|A\|, \|B\|, \|a\|, \|b\| \le B\}$
- For these, $\mathcal{N}(\epsilon, \mathcal{F}(X^{1:n}))$ scales with $O(d[\log d])$[5] [6]

[4] L. Devroye and L. Györfi and G. Lugosi: *"A Probabilistic Theory of Pattern Recognition"*, Springer, 1996.

[5] T. Zhang: "Covering Number Bounds of Certain Regularized Linear Function Classes", *JMLR* 2:527–550, 2002

[6] M. Anthony and P.L. Bartlett: *"Neural Network Learning: Theoretical Foundations"*, Cambridge Univ. Press, UK 1999.

Covering Numbers – Linear FAPPs

- $H : (\Theta, \|\cdot\|_2) \to (B(\mathcal{X}), L^\infty)$, $H : \theta \mapsto f_\theta$
- H Lipschitz with coefficient L
  - Example: $f_\theta = \theta^T \phi$ with some basis function $\phi : \mathcal{X} \to \mathbb{R}^d$.
  - $\|\theta_1^T \phi - \theta_2^T \phi\|_\infty = \sup_{x \in \mathcal{X}} |\langle \theta_1 - \theta_2, \phi(x) \rangle| \le \|\theta_1 - \theta_2\|_2 \sup_{x \in \mathcal{X}} \|\phi(x)\|_2$
- If $\mathcal{F} = \{ \theta^T \phi \,|\, \|\theta\| \le B \}$, then an ϵ/L covering of $\mathcal{F}(X^{1:n})$ can be constructed from an ϵ-covering of $\Theta = \{ \theta \,|\, \|\theta\| \le B \}$. [7]
- $\mathcal{N}(\epsilon, \mathcal{F}(X^{1:n})) \sim (\epsilon/L)^d$
- $\log \mathcal{N}(\epsilon, \mathcal{F}(X^{1:n})) \sim d \log(\epsilon/L)$.

---

[7] $\|(f_\theta)(x^{1:n}) - (f_{\theta_i})(x^{1:n})\|_1 \le n \|f_\theta - f_{\theta_i}\|_\infty \le nL \|\theta - \theta_i\|$

Covering Numbers – Linear FAPPs

- $H : (\Theta, \|\cdot\|_2) \to (B(\mathcal{X}), L^\infty)$, $H : \theta \mapsto f_\theta$
- H Lipschitz with coefficient L
  - Example: $f_\theta = \theta^T \phi$ with some basis function $\phi : \mathcal{X} \to \mathbb{R}^d$.
  - $\left\|\theta_1^T \phi - \theta_2^T \phi\right\|_\infty = \sup_{x \in \mathcal{X}} |\langle \theta_1 - \theta_2, \phi(x) \rangle| \le \|\theta_1 - \theta_2\|_2 \sup_{x \in \mathcal{X}} \|\phi(x)\|_2$
- If $\mathcal{F} = \{\theta^T \phi \,|\, \|\theta\| \le B\}$, then an ϵ/L covering of $\mathcal{F}(X^{1:n})$ can be constructed from an ϵ-covering of $\Theta = \{\theta \,|\, \|\theta\| \le B\}$. [7]
- $\mathcal{N}(\epsilon, \mathcal{F}(X^{1:n})) \sim (\epsilon/L)^d$
- $\log \mathcal{N}(\epsilon, \mathcal{F}(X^{1:n})) \sim d \log(\epsilon/L)$.

[7] $\left\|(f_\theta)(x^{1:n}) - (f_{\theta_i})(x^{1:n})\right\|_1 \le n \left\|f_\theta - f_{\theta_i}\right\|_\infty \le nL \left\|\theta - \theta_i\right\|$

Covering Numbers – Linear FAPPs

- $H : (\Theta, \|\cdot\|_2) \to (B(\mathcal{X}), L^\infty)$, $H : \theta \mapsto f_\theta$
- H Lipschitz with coefficient L
  - Example: $f_\theta = \theta^T \phi$ with some basis function $\phi : \mathcal{X} \to \mathbb{R}^d$.
  - $\left\|\theta_1^T\phi - \theta_2^T\phi\right\|_\infty = \sup_{x\in\mathcal{X}} |\langle \theta_1 - \theta_2, \phi(x)\rangle| \le \|\theta_1 - \theta_2\|_2 \sup_{x\in\mathcal{X}} \|\phi(x)\|_2$
- If $\mathcal{F} = \{\theta^T\phi \,|\, \|\theta\| \le B\}$, then an ϵ/L covering of $\mathcal{F}(X^{1:n})$ can be constructed from an ϵ-covering of $\Theta = \{\theta \,|\, \|\theta\| \le B\}$. [7]
- $\mathcal{N}(\epsilon, \mathcal{F}(X^{1:n})) \sim (\epsilon/L)^d$
- $\log \mathcal{N}(\epsilon, \mathcal{F}(X^{1:n})) \sim d \log(\epsilon/L)$.

[7] $\left\|(f_\theta)(x^{1:n}) - (f_{\theta_i})(x^{1:n})\right\|_1 \le n \left\|f_\theta - f_{\theta_i}\right\|_\infty \le nL \left\|\theta - \theta_i\right\|$

Covering Numbers – Linear FAPPs

- $H : (\Theta, \|\cdot\|_2) \to (B(\mathcal{X}), L^\infty)$, $H : \theta \mapsto f_\theta$
- H Lipschitz with coefficient L
  - Example: $f_\theta = \theta^T \phi$ with some basis function $\phi : \mathcal{X} \to \mathbb{R}^d$.
  - $\left\|\theta_1^T \phi - \theta_2^T \phi\right\|_\infty = \sup_{x \in \mathcal{X}} |\langle \theta_1 - \theta_2, \phi(x) \rangle| \leq \|\theta_1 - \theta_2\|_2 \sup_{x \in \mathcal{X}} \|\phi(x)\|_2$
- If $\mathcal{F} = \{\theta^T \phi \,|\, \|\theta\| \leq B\}$, then an ϵ/L covering of $\mathcal{F}(X^{1:n})$ can be constructed from an ϵ-covering of $\Theta = \{\theta \,|\, \|\theta\| \leq B\}$. [7]
- $\mathcal{N}(\epsilon, \mathcal{F}(X^{1:n})) \sim (\epsilon/L)^d$
- $\log \mathcal{N}(\epsilon, \mathcal{F}(X^{1:n})) \sim d \log(\epsilon/L)$.

[7] $\left\|(f_\theta)(x^{1:n}) - (f_{\theta_i})(x^{1:n})\right\|_1 \leq n \left\|f_\theta - f_{\theta_i}\right\|_\infty \leq nL \left\|\theta - \theta_i\right\|$

Covering Numbers – Linear FAPPs

- $H : (\Theta, \|\cdot\|_2) \to (B(\mathcal{X}), L^\infty)$, $H : \theta \mapsto f_\theta$
- H Lipschitz with coefficient L
  - Example: $f_\theta = \theta^T \phi$ with some basis function $\phi : \mathcal{X} \to \mathbb{R}^d$.
  - $\left\|\theta_1^T\phi - \theta_2^T\phi\right\|_\infty = \sup_{x\in\mathcal{X}} |\langle\theta_1 - \theta_2, \phi(x)\rangle| \leq \|\theta_1 - \theta_2\|_2 \sup_{x\in\mathcal{X}} \|\phi(x)\|_2$
- If $\mathcal{F} = \{\theta^T\phi \,|\, \|\theta\| \leq B\}$, then an ϵ/L covering of $\mathcal{F}(X^{1:n})$ can be constructed from an ϵ-covering of $\Theta = \{\theta \,|\, \|\theta\| \leq B\}$. [7]
- $\mathcal{N}(\epsilon, \mathcal{F}(X^{1:n})) \sim (\epsilon/L)^d$
- $\log \mathcal{N}(\epsilon, \mathcal{F}(X^{1:n})) \sim d \log(\epsilon/L)$.

[7] $\left\|(f_\theta)(x^{1:n}) - (f_{\theta_i})(x^{1:n})\right\|_1 \leq n \|f_\theta - f_{\theta_i}\|_\infty \leq nL \|\theta - \theta_i\|$

Covering Numbers – Linear FAPPs

- $H : (\Theta, \|\cdot\|_2) \to (B(\mathcal{X}), L^\infty)$, $H : \theta \mapsto f_\theta$
- H Lipschitz with coefficient L
  - Example: $f_\theta = \theta^T \phi$ with some basis function $\phi : \mathcal{X} \to \mathbb{R}^d$.
  - $\left\|\theta_1^T\phi - \theta_2^T\phi\right\|_\infty = \sup_{x\in\mathcal{X}} |\langle\theta_1 - \theta_2, \phi(x)\rangle| \le \|\theta_1 - \theta_2\|_2 \sup_{x\in\mathcal{X}} \|\phi(x)\|_2$
- If $\mathcal{F} = \{\theta^T\phi \,|\, \|\theta\| \le B\}$, then an ϵ/L covering of $\mathcal{F}(X^{1:n})$ can be constructed from an ϵ-covering of $\Theta = \{\theta \,|\, \|\theta\| \le B\}$. [7]
- $\mathcal{N}(\epsilon, \mathcal{F}(X^{1:n})) \sim (\epsilon/L)^d$
- $\log \mathcal{N}(\epsilon, \mathcal{F}(X^{1:n})) \sim d \log(\epsilon/L)$.

[7] $\left\|(f_\theta)(x^{1:n}) - (f_{\theta_i})(x^{1:n})\right\|_1 \le n \|f_\theta - f_{\theta_i}\|_\infty \le nL \|\theta - \theta_i\|$

Covering Numbers – Linear FAPPs

- $H : (\Theta, \|\cdot\|_2) \to (B(\mathcal{X}), L^\infty)$, $H : \theta \mapsto f_\theta$
- H Lipschitz with coefficient L
  - Example: $f_\theta = \theta^T \phi$ with some basis function $\phi : \mathcal{X} \to \mathbb{R}^d$.
  - $\left\|\theta_1^T \phi - \theta_2^T \phi\right\|_\infty = \sup_{x \in \mathcal{X}} |\langle \theta_1 - \theta_2, \phi(x) \rangle| \le \|\theta_1 - \theta_2\|_2 \sup_{x\in\mathcal{X}} \|\phi(x)\|_2$
- If $\mathcal{F} = \{\theta^T \phi \,|\, \|\theta\| \le B\}$, then an ϵ/L covering of $\mathcal{F}(X^{1:n})$ can be constructed from an ϵ-covering of $\Theta = \{\theta \,|\, \|\theta\| \le B\}$. [7]
- $\mathcal{N}(\epsilon, \mathcal{F}(X^{1:n})) \sim (\epsilon/L)^d$
- $\log \mathcal{N}(\epsilon, \mathcal{F}(X^{1:n})) \sim d \log(\epsilon/L)$.

[7] $\left\|(f_\theta)(x^{1:n}) - (f_{\theta_i})(x^{1:n})\right\|_1 \le n \|f_\theta - f_{\theta_i}\|_\infty \le nL \|\theta - \theta_i\|$

Outline

Finite-time Bounds

Theorem[8]: Assume MDP is regular. Fix $\delta > 0$, $\epsilon > 0$, $\mathcal{F}$, ρ, μ. Assume that $\mathcal{V}$, the "capacity" of $\mathcal{F}$ is finite. Assume that Bellman-errors for functions in $\mathcal{F}$ can be uniformly bounded:

$$\sup_{g \in \mathcal{F}} \inf_{f \in \mathcal{F}} \|f - Tg\|_{p,\mu} \leq \epsilon.$$

Then, it is possible to select N, M, K such that after K iterations of the sampling based FVI algorithm run with (μ, N, M)

$$\|V^* - V^{\pi_K}\|_{p,\rho} \leq \frac{4C^{1/p}}{(1-\gamma)^2}\, \epsilon$$

with probability at least $1-\delta$. Further, N, M, K are polynomial in $\mathcal{V}$, $R_{\max}$, $1/\epsilon$, $\log|\mathcal{A}|$, $\log(1/\delta)$, $1/(1-\gamma)$.
Here C is a constant related to how quickly future state distributions can concentrate away from ρ relative to μ.

[8] Munos & Szepesvári, ICML-2005

Relation to L^∞-error

$$\|V^* - V^{\pi_K}\|_{p,\rho} \leq \frac{4C^{1/p}}{(1-\gamma)^2}\,\epsilon$$

For $p \to \infty$ we get[9]

$$\|V^* - V^{\pi_K}\|_{\infty} \leq \frac{4}{(1-\gamma)^2}\,\epsilon$$

Previous L^∞-bound:

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

[9] The constant 4 can be replaced by 2 if sampling-effects are excluded.

Relation to L^∞-error

$$\|V^* - V^{\pi_K}\|_{p,\rho} \le \frac{4C^{1/p}}{(1-\gamma)^2}\,\epsilon$$

For $p \to \infty$ we get[9]

$$\|V^* - V^{\pi_K}\|_\infty \le \frac{4}{(1-\gamma)^2}\,\epsilon$$

Previous L^∞-bound:

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

[9] The constant 4 can be replaced by 2 if sampling-effects are excluded.

Relation to L^∞-error

$$\|V^* - V^{\pi_K}\|_{p,\rho} \le \frac{4C^{1/p}}{(1-\gamma)^2}\,\epsilon$$

For $p \to \infty$ we get[9]

$$\|V^* - V^{\pi_K}\|_{\infty} \le \frac{4}{(1-\gamma)^2}\,\epsilon$$

Previous L^∞-bound:

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

[9] The constant 4 can be replaced by 2 if sampling-effects are excluded.

MDP Regularity

- **MDP regularity**:
  - $\mathcal{X} \subseteq \mathbb{R}^d$ is compact
  - Sampled immediate rewards are bounded by $R_{\max}$

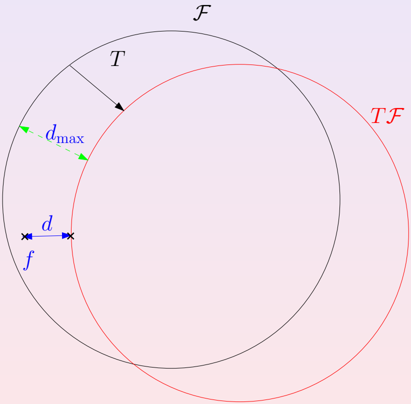
MDP Regularity

- **MDP regularity**:
  - $\mathcal{X} \subseteq \mathbb{R}^d$ is compact
  - Sampled immediate rewards are bounded by $R_{\max}$

MDP Regularity

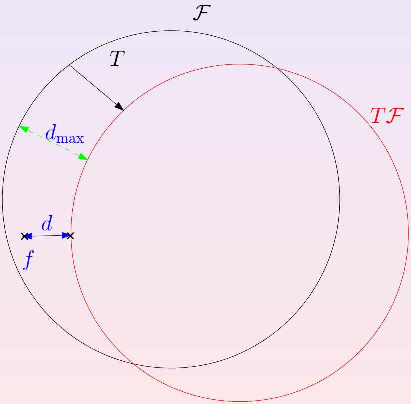
- **MDP regularity**:
  - $\mathcal{X} \subseteq \mathbb{R}^d$ is compact
  - Sampled immediate rewards are bounded by $R_{\max}$

Bellman-errors



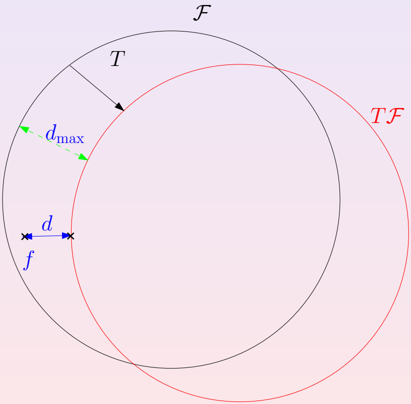
- When can we expect $d_{\max}$ to be small?
- $\mathcal{F}$ – class of smooth functions
- T does not decrease smoothness

Bellman-errors



- When can we expect $d_{\max}$ to be small?
- $\mathcal{F}$ – class of smooth functions
- T does not decrease smoothness

Bellman-errors



- When can we expect $d_{\max}$ to be small?
- $\mathcal{F}$ – class of smooth functions
- T does not decrease smoothness

Outline

Definition of constant C

- μ – distribution used in the optimization step of the algorithm
- ρ – distribution appearing in performance bounds
- m-step (worst-case) concentration of future state distribution:

$$c(m) = \sup_{\pi_1,\ldots,\pi_m,\|V\|=1} \frac{\rho P^{\pi_1} P^{\pi_2} \ldots P^{\pi_m} V}{\mu V}$$

- Average (discounted) concentration:

$$C = (1-\gamma)^2 \sum_{m \geq 1} m \gamma^{m-1} c(m).$$

Definition of constant C

- μ – distribution used in the optimization step of the algorithm
- ρ – distribution appearing in performance bounds
- m-step (worst-case) concentration of future state distribution:

$$c(m) = \sup_{\pi_1,\dots,\pi_m,\|V\|=1} \frac{\rho P^{\pi_1} P^{\pi_2} \dots P^{\pi_m} V}{\mu V}$$

- Average (discounted) concentration:

$$C = (1-\gamma)^2 \sum_{m \ge 1} m \gamma^{m-1} c(m).$$

Definition of constant C

- μ – distribution used in the optimization step of the algorithm
- ρ – distribution appearing in performance bounds
- m-step (worst-case) concentration of future state distribution:

$$c(m) = \sup_{\pi_1,\ldots,\pi_m,\|V\|=1} \frac{\rho P^{\pi_1} P^{\pi_2} \ldots P^{\pi_m} V}{\mu V}$$

- Average (discounted) concentration:

$$C = (1-\gamma)^2 \sum_{m\ge 1} m\gamma^{m-1} c(m).$$

Definition of constant C

- μ – distribution used in the optimization step of the algorithm
- ρ – distribution appearing in performance bounds
- m-step (worst-case) concentration of future state distribution:

$$c(m) = \sup_{\pi_1,\ldots,\pi_m,\|V\|=1} \frac{\rho P^{\pi_1} P^{\pi_2} \ldots P^{\pi_m} V}{\mu V}$$

- Average (discounted) concentration:

$$C = (1-\gamma)^2 \sum_{m \geq 1} m \gamma^{m-1} c(m).$$

Relation to Lyapunov exponents

Top Lyapunov exponent:

$$L^+ = \sup_\pi \limsup_{m\to\infty} \frac{1}{m} \log^+ \|\rho P^{\pi_1} P^{\pi_2} \dots P^{\pi_m}\|.$$

Statement: If $L^+ \le 0$ holds then the growth rate of $c(m)$ is polynomial. Hence,

$$C = (1-\gamma)^2 \sum_{m\ge 1} m\gamma^{m-1} c(m).$$

is finite.

Relation to Lyapunov exponents

Top Lyapunov exponent:

$$L^+ = \sup_{\pi} \limsup_{m\to\infty} \frac{1}{m} \log^+ \| \rho P^{\pi_1} P^{\pi_2} \dots P^{\pi_m} \|.$$

Statement: If $L^+ \leq 0$ holds then the growth rate of $c(m)$ is polynomial. Hence,

$$C = (1-\gamma)^2 \sum_{m\geq 1} m\gamma^{m-1} c(m).$$

is finite.

Why C? – Pointwise Analysis of Error

Let

$$
\begin{aligned}
V_{k+1} &= TV_k - \epsilon_k, \quad \epsilon_k = TV_k - V_{k+1}, \\
\pi_k & \qquad \text{greedy w.r.t. } V_k.
\end{aligned}
$$

Then

$$
\begin{aligned}
V^* - V^{\pi_K} \le \; & (I - \gamma P^{\pi_K})^{-1} \Big\{ \sum_{k=0}^{K-1} \gamma^{K-k} \big[(P^{\pi^*})^{K-k} + P^{\pi_K} P^{\pi_{K-1}} \dots P^{\pi_{k+1}} \big] |\epsilon_k| \\
& + \gamma^{K+1} \big[(P^{\pi^*})^{K+1} + (P^{\pi_K} P^{\pi_K} P^{\pi_{K-1}} \dots P^{\pi_1}) \big] |V^* - V_0| \Big\}.
\end{aligned}
$$

Why C? – Pointwise Analysis of Error

Let

$$\begin{aligned} V_{k+1} &= TV_k - \epsilon_k, \quad \epsilon_k = TV_k - V_{k+1}, \\ \pi_k & \quad \text{greedy w.r.t. } V_k. \end{aligned}$$

Then

$$\begin{aligned} V^* - V^{\pi_K} \leq \\ & (I - \gamma P^{\pi_K})^{-1} \Big\{ \textstyle\sum_{k=0}^{K-1} \gamma^{K-k} \left[(P^{\pi^*})^{K-k} + P^{\pi_K} P^{\pi_{K-1}} \dots P^{\pi_{k+1}} \right] |\epsilon_k| \\ & + \gamma^{K+1} \left[(P^{\pi^*})^{K+1} + (P^{\pi_K} P^{\pi_K} P^{\pi_{K-1}} \dots P^{\pi_1}) \right] |V^* - V_0| \Big\}. \end{aligned}$$

Why C?

C relates ρ and μ: Optimization w.r.t. μ is not a good idea if future state distributions (starting from ρ) can concentrate “away from μ”.

Open question: How to select μ?

When will C be finite?

State-space form:

$$X_{t+1} = f(X_t, A_t) + W_t, \quad W_t \text{ random}$$

If W_t assumes a density p then C can be bounded in terms of $\sup_w p(w)$.
Note: This is not the only way to make C finite: C can be finite even for deterministic systems as well.

When will C be finite?

State-space form:

$$X_{t+1} = f(X_t, A_t) + W_t, \quad W_t \text{ random}$$

If W_t assumes a density p then C can be bounded in terms of $\sup_w p(w)$.

Note: This is not the only way to make C finite: C can be finite even for deterministic systems as well.

When will C be finite?

State-space form:

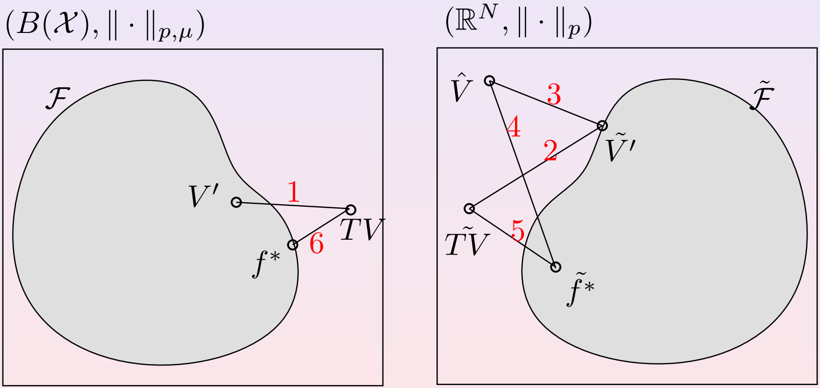
$$X_{t+1} = f(X_t, A_t) + W_t, \quad W_t \text{ random}$$

If W_t assumes a density p then C can be bounded in terms of $\sup_w p(w)$.
Note: This is not the only way to make C finite: C can be finite even for deterministic systems as well.

Proof – Main Steps

- Single-iteration PAC Bound (needs covering numbers)
- L^p bounds for AVI (see above)
- Putting it all together: Union error bounds

Single-iteration PAC Bounds



Outline

The Single-sample Variant

- Why not use a single set of samples throughout all the iterations?
- **Technical problem:** In the previous result one bounds

$$\mathbb{P}(\|V_{k+1} - TV_k\|_{p,\mu} > \epsilon | D_k),$$

  where D_k is the sample used up to iteration k. If $D_k = D$, V_{k+1} becomes measurable w.r.t. D_k and $\mathbb{P}(\|V_{k+1} - TV_k\|_{p,\mu} > \epsilon | D)$ degenerates.
- **Question:** Will this method still work?
- **Answer:** Yes!

Idea: Strengthen single-iteration PAC bound:

$$\sup_{g\in\mathcal{F}} \sup_{f\in\mathcal{F}} \left| \|f - Tg\|_{p,\mu} - \|f - Tg\|_{p,\hat{\mu}} \right| \geq$$

$$\sup_{f\in\mathcal{F}} \left| \|f - TV\|_{p,\mu} - \|f - TV\|_{p,\hat{\mu}} \right|$$

The Single-sample Variant

- Why not use a single set of samples throughout all the iterations?
- **Technical problem:** In the previous result one bounds

$$\mathbb{P}(\|V_{k+1} - TV_k\|_{p,\mu} > \epsilon | D_k),$$

  where D_k is the sample used up to iteration k. If $D_k = D$, V_{k+1} becomes measurable w.r.t. D_k and $\mathbb{P}(\|V_{k+1} - TV_k\|_{p,\mu} > \epsilon | D)$ degenerates.
- **Question:** Will this method still work?
- **Answer:** Yes!

Idea: Strengthen single-iteration PAC bound:

$$\sup_{g\in\mathcal{F}} \sup_{f\in\mathcal{F}} \left| \|f - Tg\|_{p,\mu} - \|f - Tg\|_{p,\hat{\mu}} \right| \geq \sup_{f\in\mathcal{F}} \left| \|f - TV\|_{p,\mu} - \|f - TV\|_{p,\hat{\mu}} \right|$$

The Single-sample Variant

- Why not use a single set of samples throughout all the iterations?
- **Technical problem:** In the previous result one bounds

$$\mathbb{P}(\|V_{k+1} - TV_k\|_{p,\mu} > \epsilon | D_k),$$

where D_k is the sample used up to iteration k. If $D_k = D$, V_{k+1} becomes measurable w.r.t. D_k and $\mathbb{P}(\|V_{k+1} - TV_k\|_{p,\mu} > \epsilon | D)$ degenerates.
- **Question:** Will this method still work?
- **Answer:** Yes!

Idea: Strengthen single-iteration PAC bound:

$$\sup_{g \in \mathcal{F}} \sup_{f \in \mathcal{F}} \left| \|f - Tg\|_{p,\mu} - \|f - Tg\|_{p,\hat{\mu}} \right| \geq$$

$$\sup_{f \in \mathcal{F}} \left| \|f - TV\|_{p,\mu} - \|f - TV\|_{p,\hat{\mu}} \right|$$

The Single-sample Variant

- Why not use a single set of samples throughout all the iterations?
- **Technical problem:** In the previous result one bounds

$$\mathbb{P}(\|V_{k+1} - TV_k\|_{p,\mu} > \epsilon | D_k),$$

  where D_k is the sample used up to iteration k. If $D_k = D$, V_{k+1} becomes measurable w.r.t. D_k and $\mathbb{P}(\|V_{k+1} - TV_k\|_{p,\mu} > \epsilon | D)$ degenerates.
- **Question:** Will this method still work?
- **Answer:** Yes!

Idea: Strengthen single-iteration PAC bound:

$$\sup_{g\in\mathcal{F}} \sup_{f\in\mathcal{F}} \left| \|f - Tg\|_{p,\mu} - \|f - Tg\|_{p,\hat{\mu}} \right| \geq$$

$$\sup_{f\in\mathcal{F}} \left| \|f - TV\|_{p,\mu} - \|f - TV\|_{p,\hat{\mu}} \right|$$

The Single-sample Variant

- Why not use a single set of samples throughout all the iterations?
- **Technical problem:** In the previous result one bounds

$$\mathbb{P}(\|V_{k+1} - TV_k\|_{p,\mu} > \epsilon | D_k),$$

where D_k is the sample used up to iteration k. If $D_k = D$, V_{k+1} becomes measurable w.r.t. D_k and $\mathbb{P}(\|V_{k+1} - TV_k\|_{p,\mu} > \epsilon | D)$ degenerates.
- **Question:** Will this method still work?
- **Answer:** Yes!

Idea: Strengthen single-iteration PAC bound:

$$\sup_{g\in\mathcal{F}} \sup_{f\in\mathcal{F}} \left| \|f - Tg\|_{p,\mu} - \|f - Tg\|_{p,\hat{\mu}} \right| \geq$$

$$\sup_{f\in\mathcal{F}} \left| \|f - TV\|_{p,\mu} - \|f - TV\|_{p,\hat{\mu}} \right|$$

Outline

How to Use the Result?

- **Problem 1:** V_K does not itself lead to a policy: Given V_K, we still need to compute a greedy policy w.r.t. V_K. How?
- **Good news:** Can do it using Monte-Carlo.
- This works (similar bounds to the previous ones).
- **Problem 2: How to select $\mathcal{F}$?** E.g. $\mathcal{F}_1 \subset \mathcal{F}_2 \subset \ldots \mathcal{F}_s \ldots$, increasingly richer parameterizations. Capacity and approximation power both grow. Procedure:
  1. Select target precision
  2. Select s large enough as a function of the target precision[10]
  3. Select N, M, K as in the theorem
- **Note:** Given a finite amount of data, the capacity and approximation power of $\mathcal{F}$ needs to be traded off.

---

[10] Use Jackson's theorem

How to Use the Result?

- **Problem 1:** V_K does not itself lead to a policy: Given V_K, we still need to compute a greedy policy w.r.t. V_K. How?
- **Good news:** Can do it using Monte-Carlo.
- This works (similar bounds to the previous ones).
- **Problem 2: How to select $\mathcal{F}$?** E.g. $\mathcal{F}_1 \subset \mathcal{F}_2 \subset \ldots \mathcal{F}_s \ldots$, increasingly richer parameterizations. Capacity and approximation power both grow. Procedure:
  1. Select target precision
  2. Select s large enough as a function of the target precision[10]
  3. Select N, M, K as in the theorem.
- **Note:** Given a finite amount of data, the capacity and approximation power of $\mathcal{F}$ needs to be traded off.

[10] Use Jackson's theorem

How to Use the Result?

- **Problem 1:** V_K does not itself lead to a policy: Given V_K, we still need to compute a greedy policy w.r.t. V_K. How?
- **Good news:** Can do it using Monte-Carlo.
- This works (similar bounds to the previous ones).
- **Problem 2: How to select $\mathcal{F}$?** E.g. $\mathcal{F}_1 \subset \mathcal{F}_2 \subset \ldots \mathcal{F}_s \ldots$, increasingly richer parameterizations. Capacity and approximation power both grow. Procedure:
  1. Select target precision
  2. Select s large enough as a function of the target precision[10]
  3. Select N, M, K as in the theorem.
- **Note:** Given a finite amount of data, the capacity and approximation power of $\mathcal{F}$ needs to be traded off.

[10] Use Jackson's theorem

How to Use the Result?

- **Problem 1:** V_K does not itself lead to a policy: Given V_K, we still need to compute a greedy policy w.r.t. V_K. How?
- **Good news:** Can do it using Monte-Carlo.
- This works (similar bounds to the previous ones).
- **Problem 2: How to select $\mathcal{F}$?** E.g. $\mathcal{F}_1 \subset \mathcal{F}_2 \subset \ldots \mathcal{F}_s \ldots$, increasingly richer parameterizations. Capacity and approximation power both grow. Procedure:
  1. Select target precision
  2. Select s large enough as a function of the target precision[10]
  3. Select N, M, K as in the theorem.
- **Note:** Given a finite amount of data, the capacity and approximation power of $\mathcal{F}$ needs to be traded off.

[10] Use Jackson's theorem

How to Use the Result?

- **Problem 1:** V_K does not itself lead to a policy: Given V_K, we still need to compute a greedy policy w.r.t. V_K. How?
- **Good news:** Can do it using Monte-Carlo.
- This works (similar bounds to the previous ones).
- **Problem 2: How to select $\mathcal{F}$?** E.g. $\mathcal{F}_1 \subset \mathcal{F}_2 \subset \ldots \mathcal{F}_s \ldots$, increasingly richer parameterizations. Capacity and approximation power both grow. Procedure:
  1. Select target precision
  2. Select s large enough as a function of the target precision[10]
  3. Select N, M, K as in the theorem.
- **Note:** Given a finite amount of data, the capacity and approximation power of $\mathcal{F}$ needs to be traded off.

[10] Use Jackson's theorem

How to Use the Result?

- **Problem 1:** V_K does not itself lead to a policy: Given V_K, we still need to compute a greedy policy w.r.t. V_K. How?
- **Good news:** Can do it using Monte-Carlo.
- This works (similar bounds to the previous ones).
- **Problem 2: How to select $\mathcal{F}$?** E.g. $\mathcal{F}_1 \subset \mathcal{F}_2 \subset \ldots \mathcal{F}_s \ldots$, increasingly richer parameterizations. Capacity and approximation power both grow. Procedure:
  1. Select target precision
  2. Select s large enough as a function of the target precision[10]
  3. Select N, M, K as in the theorem.
- **Note:** Given a finite amount of data, the capacity and approximation power of $\mathcal{F}$ needs to be traded off.

[10] Use Jackson's theorem

How to Use the Result?

- **Problem 1:** V_K does not itself lead to a policy: Given V_K, we still need to compute a greedy policy w.r.t. V_K. How?
- **Good news:** Can do it using Monte-Carlo.
- This works (similar bounds to the previous ones).
- **Problem 2: How to select $\mathcal{F}$?** E.g. $\mathcal{F}_1 \subset \mathcal{F}_2 \subset \ldots \mathcal{F}_s \ldots$, increasingly richer parameterizations. Capacity and approximation power both grow. Procedure:
  1. Select target precision
  2. Select s large enough as a function of the target precision[10]
  3. Select N, M, K as in the theorem.
- **Note:** Given a finite amount of data, the capacity and approximation power of $\mathcal{F}$ needs to be traded off.

[10] Use Jackson's theorem

How to Use the Result?

- **Problem 1:** V_K does not itself lead to a policy: Given V_K, we still need to compute a greedy policy w.r.t. V_K. How?
- **Good news:** Can do it using Monte-Carlo.
- This works (similar bounds to the previous ones).
- **Problem 2: How to select $\mathcal{F}$?** E.g. $\mathcal{F}_1 \subset \mathcal{F}_2 \subset \ldots \mathcal{F}_s \ldots$, increasingly richer parameterizations. Capacity and approximation power both grow. Procedure:
  1. Select target precision
  2. Select s large enough as a function of the target precision[10]
  3. Select N, M, K as in the theorem.
- **Note:** Given a finite amount of data, the capacity and approximation power of $\mathcal{F}$ needs to be traded off.

[10] Use Jackson's theorem

Illustration

- Optimal replacement problem (e.g. Rust, 1996)
- X_t – accumulated utilization of a durable ($X_t = 0$: new)
  - 'keep': $X_{t+1} - X_t \sim \exp(-\beta(X_{t+1} - X_t))$, $X_{t+1} - X_t \geq 0$
  - 'replace': $X_{t+1} \sim \exp(-\beta X_{t+1})$, $X_{t+1} \geq 0$
- $r(x, \text{'keep'}) = -4x$, $r(x, \text{'replace'}) = -30$

Illustration

- Optimal replacement problem (e.g. Rust, 1996)
- X_t – accumulated utilization of a durable ($X_t = 0$: new)
  - 'keep': $X_{t+1} - X_t \sim \exp(-\beta(X_{t+1} - X_t))$, $X_{t+1} - X_t \geq 0$
  - 'replace': $X_{t+1} \sim \exp(-\beta X_{t+1})$, $X_{t+1} \geq 0$
- $r(x, \text{'keep'}) = -4x$, $r(x, \text{'replace'}) = -30$

Illustration

- Optimal replacement problem (e.g. Rust, 1996)
- X_t – accumulated utilization of a durable ($X_t = 0$: new)
  - 'keep': $X_{t+1} - X_t \sim \exp(-\beta(X_{t+1} - X_t))$, $X_{t+1} - X_t \geq 0$
  - 'replace': $X_{t+1} \sim \exp(-\beta X_{t+1})$, $X_{t+1} \geq 0$
- $r(x, \text{'keep'}) = -4x$, $r(x, \text{'replace'}) = -30$

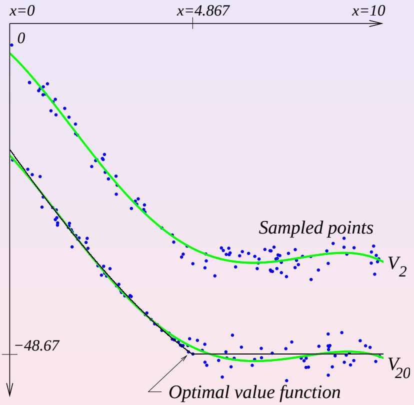
Illustration

- Optimal replacement problem (e.g. Rust, 1996)
- X_t – accumulated utilization of a durable ($X_t = 0$: new)
    - 'keep': $X_{t+1} - X_t \sim \exp(-\beta(X_{t+1} - X_t))$, $X_{t+1} - X_t \geq 0$
    - 'replace': $X_{t+1} \sim \exp(-\beta X_{t+1})$, $X_{t+1} \geq 0$
- $r(x, \text{'keep'}) = -4x$, $r(x, \text{'replace'}) = -30$

Illustration

- Optimal replacement problem (e.g. Rust, 1996)
- X_t – accumulated utilization of a durable ($X_t = 0$: new)
  - 'keep': $X_{t+1} - X_t \sim \exp(-\beta(X_{t+1} - X_t))$, $X_{t+1} - X_t \geq 0$
  - 'replace': $X_{t+1} \sim \exp(-\beta X_{t+1})$, $X_{t+1} \geq 0$
- $r(x, '\text{keep}') = -4x$, $r(x, '\text{replace}') = -30$

Iterates



Two iterates: $k = 2$ and $k = 20$. $N = 100$, $M = 10$
$\mathcal{F}$: Chebyshev-polynomials of degree 4.

Single-sample vs. multi-sample

- $\mathcal{F}$: Chebyshev-polynomials with $d = 5$.
- $K = 10$
- $N = 100$
- #runs$= 50$
- Total number of samples: 10000.

Single-sample vs. multi-sample

- $\mathcal{F}$: Chebyshev-polynomials with $d = 5$.
- $K = 10$
- $N = 100$
- #runs$= 50$
- Total number of samples: 10000.

Single-sample vs. multi-sample

- $\mathcal{F}$: Chebyshev-polynomials with $d = 5$.
- $K = 10$
- $N = 100$
- #runs$= 50$
- Total number of samples: 10000.

Single-sample vs. multi-sample

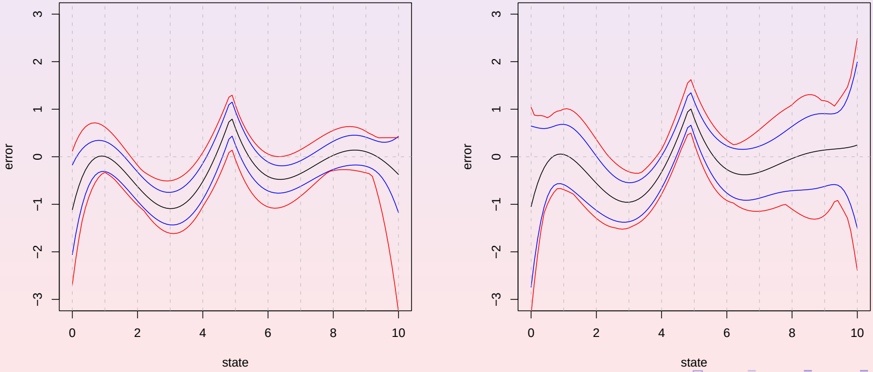
- $\mathcal{F}$: Chebyshev-polynomials with $d = 5$.
- $K = 10$
- $N = 100$
- #runs$=$ 50
- Total number of samples: 10000.

Single-sample vs. multi-sample

- $\mathcal{F}$: Chebyshev-polynomials with $d = 5$.
- $K = 10$
- $N = 100$
- #runs$= 50$
- Total number of samples: 10000.

Single-sample vs. multi-sample

- Single-sample: $M = 100$; $N \times M = 10,000$ (left)
- Multi-sample: $M = 10$; $N \times M \times K = 10,000$ (right)



Conclusions

- Model: Continuous (or infinite, or very large) state space, generative model of the environment
- Main condition: Future state distributions do not concentrate fast
- Result: Error of multi/single-sample FVI bounded with high prob, in terms of approximation power and capacity of underlying function space, and the so-called concentration coefficient
- Choose $\mathcal{F}$ to be sufficiently rich; then as the number of samples and iterations growth to infinity, FVI will ultimately yield arbitrarily good estimates of V^*

Conclusions

- Model: Continuous (or infinite, or very large) state space, generative model of the environment
- Main condition: Future state distributions do not concentrate fast
- Result: Error of multi/single-sample FVI bounded with high prob, in terms of approximation power and capacity of underlying function space, and the so-called concentration coefficient
- Choose $\mathcal{F}$ to be sufficiently rich; then as the number of samples and iterations growth to infinity, FVI will ultimately yield arbitrarily good estimates of V^*

Conclusions

- Model: Continuous (or infinite, or very large) state space, generative model of the environment
- Main condition: Future state distributions do not concentrate fast
- Result: Error of multi/single-sample FVI bounded with high prob, in terms of approximation power and capacity of underlying function space, and the so-called concentration coefficient
- Choose $\mathcal{F}$ to be sufficiently rich; then as the number of samples and iterations growth to infinity, FVI will ultimately yield arbitrarily good estimates of V^*

Conclusions

- Model: Continuous (or infinite, or very large) state space, generative model of the environment
- Main condition: Future state distributions do not concentrate fast
- Result: Error of multi/single-sample FVI bounded with high prob, in terms of approximation power and capacity of underlying function space, and the so-called concentration coefficient
- Choose $\mathcal{F}$ to be sufficiently rich; then as the number of samples and iterations growth to infinity, FVI will ultimately yield arbitrarily good estimates of V^*

Take-home Message

- Simple ideas might sometimes work
- Sometimes noise helps

Take-home Message

- Simple ideas might sometimes work
- Sometimes noise helps

Future work

- Initially (in the iteration) use less samples
- Multi-sample variant: Exploit conditional independence of errors
- Single-sample vs. multi-sample
- Sharpen C
- Estimate C from data $\Rightarrow$ adaptive versions (data-dependent bounds, optimal stopping)
- Regularization, imperfect optimization (neural nets)
- Single trajectory learning: Policy iteration (mostly done)
- $M = 1$ – at least for L^2-errors
- Average cost
- Bellman-residual minimization

Future work

- Initially (in the iteration) use less samples
- Multi-sample variant: Exploit conditional independence of errors
- Single-sample vs. multi-sample
- Sharpen C
- Estimate C from data $\Rightarrow$ adaptive versions (data-dependent bounds, optimal stopping)
- Regularization, imperfect optimization (neural nets)
- Single trajectory learning: Policy iteration (mostly done)
- $M = 1$ – at least for L^2-errors
- Average cost
- Bellman-residual minimization

Future work

- Initially (in the iteration) use less samples
- Multi-sample variant: Exploit conditional independence of errors
- Single-sample vs. multi-sample
- Sharpen C
- Estimate C from data $\Rightarrow$ adaptive versions (data-dependent bounds, optimal stopping)
- Regularization, imperfect optimization (neural nets)
- Single trajectory learning: Policy iteration (mostly done)
- $M = 1$ – at least for L^2-errors
- Average cost
- Bellman-residual minimization

Future work

- Initially (in the iteration) use less samples
- Multi-sample variant: Exploit conditional independence of errors
- Single-sample vs. multi-sample
- Sharpen C
- Estimate C from data $\Rightarrow$ adaptive versions (data-dependent bounds, optimal stopping)
- Regularization, imperfect optimization (neural nets)
- Single trajectory learning: Policy iteration (mostly done)
- $M = 1$ – at least for L^2-errors
- Average cost
- Bellman-residual minimization

Future work

- Initially (in the iteration) use less samples
- Multi-sample variant: Exploit conditional independence of errors
- Single-sample vs. multi-sample
- Sharpen C
- Estimate C from data $\Rightarrow$ adaptive versions (data-dependent bounds, optimal stopping)
- Regularization, imperfect optimization (neural nets)
- Single trajectory learning: Policy iteration (mostly done)
- $M = 1$ – at least for L^2-errors
- Average cost
- Bellman-residual minimization

Future work

- Initially (in the iteration) use less samples
- Multi-sample variant: Exploit conditional independence of errors
- Single-sample vs. multi-sample
- Sharpen C
- Estimate C from data $\Rightarrow$ adaptive versions (data-dependent bounds, optimal stopping)
- Regularization, imperfect optimization (neural nets)
- Single trajectory learning: Policy iteration (mostly done)
- $M = 1$ – at least for L^2-errors
- Average cost
- Bellman-residual minimization

Future work

- Initially (in the iteration) use less samples
- Multi-sample variant: Exploit conditional independence of errors
- Single-sample vs. multi-sample
- Sharpen C
- Estimate C from data $\Rightarrow$ adaptive versions (data-dependent bounds, optimal stopping)
- Regularization, imperfect optimization (neural nets)
- Single trajectory learning: Policy iteration (mostly done)
- $M = 1$ – at least for L^2-errors
- Average cost
- Bellman-residual minimization

Future work

- Initially (in the iteration) use less samples
- Multi-sample variant: Exploit conditional independence of errors
- Single-sample vs. multi-sample
- Sharpen C
- Estimate C from data $\Rightarrow$ adaptive versions (data-dependent bounds, optimal stopping)
- Regularization, imperfect optimization (neural nets)
- Single trajectory learning: Policy iteration (mostly done)
- $M = 1$ – at least for L^2-errors
- Average cost
- Bellman-residual minimization

Future work

- Initially (in the iteration) use less samples
- Multi-sample variant: Exploit conditional independence of errors
- Single-sample vs. multi-sample
- Sharpen C
- Estimate C from data $\Rightarrow$ adaptive versions (data-dependent bounds, optimal stopping)
- Regularization, imperfect optimization (neural nets)
- Single trajectory learning: Policy iteration (mostly done)
- $M = 1$ – at least for L^2-errors
- Average cost
- Bellman-residual minimization

Future work

- Initially (in the iteration) use less samples
- Multi-sample variant: Exploit conditional independence of errors
- Single-sample vs. multi-sample
- Sharpen C
- Estimate C from data $\Rightarrow$ adaptive versions (data-dependent bounds, optimal stopping)
- Regularization, imperfect optimization (neural nets)
- Single trajectory learning: Policy iteration (mostly done)
- $M = 1$ – at least for L^2-errors
- Average cost
- Bellman-residual minimization

Questions?

???