Introduction to Reinforcement Learning and multi-armed bandits

Rémi Munos

INRIA Lille - Nord Europe
Currently on leave at MSR-NE
http://researchers.lille.inria.fr/∼munos/

NETADIS Summer School 2013, Hillerod, Denmark

Outline of the course

- Part 1: Introduction to Reinforcement Learning and Dynamic Programming
  - Dynamic programming: value iteration, policy iteration
  - Q-learning.
- Part 2: Approximate DP and RL
  - L_∞-norm performance bounds
  - Sample-based algorithms.
  - Links with statistical learning
- Part 3: Intro to multi-armed bandits
  - The stochastic bandit: UCB
  - The adversarial bandit: EXP3
  - Approximation of Nash equilibrium
  - Monte-Carlo Tree Search

Part 1: Introduction to Reinforcement Learning and Dynamic Programming

A few general references:

- *Neuro Dynamic Programming*, Bertsekas et Tsitsiklis, 1996.
- *Introduction to Reinforcement Learning*, Sutton and Barto, 1998.
- *Markov Decision Problems*, Puterman, 1994.
- *Algorithms for Reinforcement Learning*, Szepesvári, 2009.

Introduction to Reinforcement Learning (RL)

- Learn to make good decisions in unknown environments
- Learning from experience: success or failures
- Examples: learning to ride a bicycle, play chess, autonomous robotics, operation research, playing in stochastic market, ...

I learned to ride with RL...

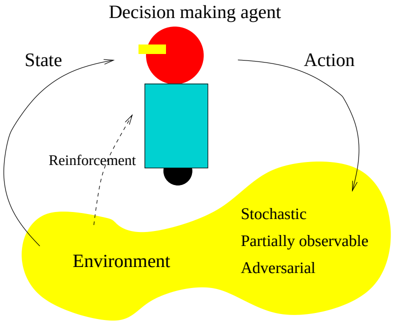
A few applications

- TD-Gammon. [Tesauro 1992-1995]: Backgammon.
- KnightCap [Baxter et al. 1998]: chess ($\simeq$2500 ELO)
- Robotics: juggling, acrobots [Schaal and Atkeson, 1994]
- Mobile robot navigation [Thrun et al., 1999]
- Elevator controller [Crites et Barto, 1996],
- Packet Routing [Boyan et Littman, 1993],
- Job-Shop Scheduling [Zhang et Dietterich, 1995],
- Production manufacturing optimization[Mahadevan et al., 1998],
- Game of poker (Bandit algo for Nash computation)
- Game of go (hierarchical bandits, UCT)

http://www.ualberta.ca/∼szepesva/RESEARCH/RLApplications.html

INFINITE MARIO BROS
(NOT Sequential Learning)
http://www

Reinforcement Learning

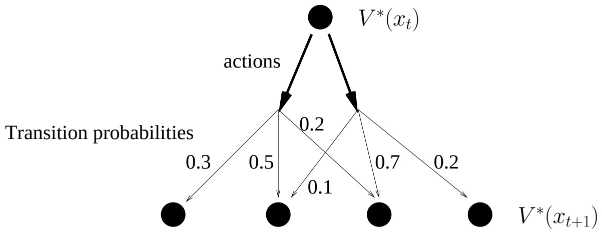


- **Environment:** can be stochastic (Tetris), adversarial (Chess), partially unknown (bicycle), partially observable (robot)
- **Available information:** the reinforcement (may be delayed)
- **Goal:** maximize the expected sum of future rewards.

Problem: How to sacrify a short term small reward to priviledge larger rewards in the long term?

Optimal value function

- Gives an evaluation of each state if the agent plays optimally.
- Ex: in a stochastic environment:



- Bellman equation:
  $V^*(x_t) = \max_{a \in A} \left[r(x_t, a) + \sum_y p(y|x_t, a) V^*(y) \right]$
- Temporal difference: $\delta_t = V^*(x_{t+1}) + r(x_t, a_t) - V^*(x_t)$
- If V^* is known, then when choosing the optimal action a_t, $\mathbb{E}[\delta_t] = 0$ (i.e., in average there is no surprise)

Challenges of RL

- Environment may be stochastic, adversarial, partially observable...
- The state-dynamics and reward functions are unknown: we need to combine
  - Learning
  - Planning
- The curse of dimensionality: We need to rely on *approximations* for representing the value function and the optimal policy.

Introduction to Dynamic Programming

A **Markov Decision Process** (X, A, p, r) defines a discrete-time process $(x_t) \in X$ where:

- X: **state space**
- A: **action space** (or decisions)
- **State dynamics**: All relevant information about future is included in the current state and action (Markov property)

  $$\mathbb{P}(x_{t+1} \mid x_t, x_{t-1}, \dots, x_0, a_t, a_{t-1}, \dots, a_0) = \mathbb{P}(x_{t+1} \mid x_t, a_t)$$

  Thus we define the **transition probabilities** $p(y|x, a)$
- **Reinforcement** (or **reward**): $r(x, a)$ is obtained when choosing action a in state x.

Definition of policy

Policy $\pi = (\pi_1, \pi_2, \dots)$, where at time t,

$$\pi_t : X \to A$$

maps an action $\pi_t(x)$ to any possible state x.

Given a policy π the process $(x_t)_{t \geq 0}$ is a Markov chain with transition probabilities

$$p(x_{t+1}|x_t) = p(x_{t+1}|x_t, \pi_t(x_t)).$$

When the policy is independent of time, $\pi = (\pi, \pi, \dots, \pi)$, the policy is called *stationary* (or *Markovian*).

Performance of a policy

For any policy π, define the **value function** V^π:

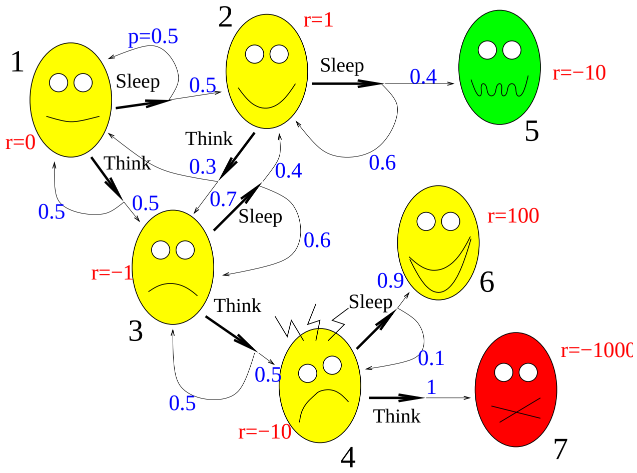
Infinite horizon:

- Discounted: $V^\pi(x) = \mathbb{E}\big[\sum_{t=0}^{\infty} \gamma^t r(x_t, a_t) \,|\, x_0 = x; \pi\big]$,

  where $0 \le \gamma < 1$ is the discount factor
- Undiscounted: $V^\pi(x) = \mathbb{E}\big[\sum_{t=0}^{\infty} r(x_t, a_t) \,|\, x_0 = x; \pi\big]$
- Average: $V^\pi(x) = \lim_{T\to\infty} \frac{1}{T}\mathbb{E}\big[\sum_{t=0}^{T-1} r(x_t, a_t) \,|\, x_0 = x; \pi\big]$

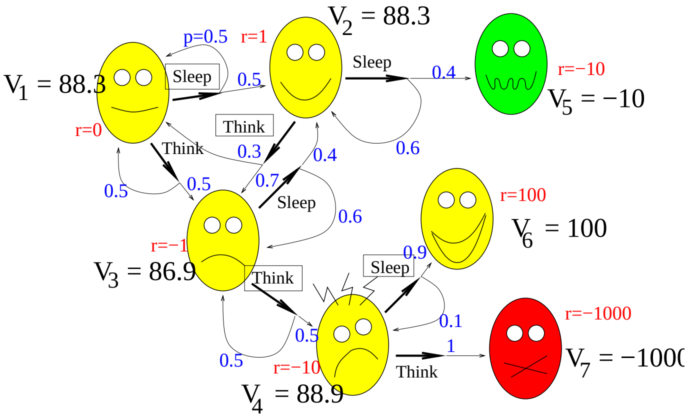
Finite horizon: $V^\pi(x, t) = \mathbb{E}\big[\sum_{s=t}^{T-1} r(x_s, a_s) + R(x_T) \,|\, x_t = x; \pi\big]$

The dilemma of the Netadis SS student



You try to maximize the sum of rewards!

Solution of the Netadis SS student



$V_5 = -10,\ V_6 = 100,\ V_7 = -1000,$
$V_4 = -10 + 0.9 V_6 + 0.1 V_4 \simeq 88.9.$
$V_3 = -1 + 0.5 V_4 + 0.5 V_3 \simeq 86.9.$ $V_2 = 1 + 0.7 V_3 + 0.3 V_1$ and
$V_1 = \max\{0.5 V_2 + 0.5 V_1, 0.5 V_3 + 0.5 V_1\}$, thus: $V_1 = V_2 = 88.3.$

Infinite horizon, discounted problems

For any stationary policy π, define the **value function** V^π as:

$$V^\pi(x) = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t r(x_t, \pi(x_t)) \,\Big|\, x_0 = x; \pi\Big],$$

where $0 \leq \gamma < 1$ a discount factor (which relates rewards in the future compared to current rewards).

Bellman equation for V^π

Proposition 1 (Bellman equation).

For any policy π, V^π satisfies:

$$V^\pi(x) = r(x, \pi(x)) + \gamma \sum_{y \in X} p(y|x, \pi(x)) V^\pi(y),$$

Thus V^π is the fixed point of the **Bellman operator** $\mathcal{T}^\pi$ (i.e., $V^\pi = \mathcal{T}^\pi V^\pi$) where $\mathcal{T}^\pi W$ is defined as

$$\mathcal{T}^\pi W(x) = r(x, \pi(x)) + \gamma \sum_{y} p(y|x, \pi(x)) W(y)$$

Using matrix notations, $\mathcal{T}^\pi W = r^\pi + \gamma P^\pi W$, where $r^\pi(x) = r(x, \pi(x))$ and $P^\pi(x, y) = p(y|x, \pi(x))$.

Proof of Proposition 1

$$
\begin{aligned}
V^\pi(x) &= \mathbb{E}\Big[\sum_{t\geq 0}\gamma^t r(x_t,\pi(x_t))\,|\,x_0=x;\pi\Big] \\
&= r(x,\pi(x)) + \mathbb{E}\Big[\sum_{t\geq 1}\gamma^t r(x_t,\pi(x_t))\,|\,x_0=x;\pi\Big] \\
&= r(x,\pi(x)) + \gamma\sum_y P(x_1=y\,|\,x_0=x;\pi) \\
&\qquad\qquad \mathbb{E}\Big[\sum_{t\geq 1}\gamma^{t-1} r(x_t,\pi(x_t))\,|\,x_1=y;\pi\Big] \\
&= r(x,\pi(x)) + \gamma\sum_y p(y|x,\pi(x))V^\pi(y).
\end{aligned}
$$

Bellman equation for V^*

Define the **optimal value function**: $V^* = \sup_\pi V^\pi$.

Proposition 2 (Dynamic programming equation).

V^* satisfies:

$$V^*(x) = \max_{a \in A} \Big[r(x,a) + \gamma \sum_{y \in X} p(y|x,a) V^*(y) \Big].$$

Thus V^* is the fixed point of the **Dynamic programming operator** $\mathcal{T}$ (i.e., $V^* = \mathcal{T} V^*$) where $\mathcal{T} W$ is defined as

$$\mathcal{T} W(x) = \max_{a \in A} \Big[r(x,a) + \gamma \sum_{y \in X} p(y|x,a) W(y) \Big].$$

Proof of Proposition 2

And for all policy $\pi = (a, \pi')$ (not necessarily stationary),

$$
\begin{aligned}
V^*(x) &= \max_\pi \mathbb{E}\Big[\sum_{t\geq 0} \gamma^t r(x_t, \pi(x_t)) \,|\, x_0 = x; \pi \Big] \\
&= \max_{(a,\pi')} \Big[r(x,a) + \gamma \sum_y p(y|x,a) V^{\pi'}(y) \Big] \\
&= \max_a \Big[r(x,a) + \gamma \sum_y p(y|x,a) \max_{\pi'} V^{\pi'}(y) \Big] \qquad (1) \\
&= \max_a \Big[r(x,a) + \gamma \sum_y p(y|x,a) V^*(y) \Big].
\end{aligned}
$$

where (1) holds since:

- $\max_{\pi'} \sum_y p(y|x,a) V^{\pi'}(y) \leq \sum_y p(y|x,a) \max_{\pi'} V^{\pi'}(y)$
- Let $\bar{\pi}$ be the policy defined by $\bar{\pi}(y) = \arg\max_{\pi'} V^{\pi'}(y)$. Thus $\sum_y p(y|x,a) \max_{\pi'} V^{\pi'}(y) = \sum_y p(y|x,a) V^{\bar{\pi}}(y) \leq \max_{\pi'} \sum_y p(y|x,a) V^{\pi'}(y)$.

Properties of the Bellman operators

- **Monotonicity**: If $W_1 \le W_2$ (componentwise) then

$$\mathcal{T}^\pi W_1 \le \mathcal{T}^\pi W_2, \text{ and } \mathcal{T} W_1 \le \mathcal{T} W_2.$$

- **Contraction in max-norm**: For any vectors W_1 and W_2,

$$\begin{aligned} ||\mathcal{T}^\pi W_1 - \mathcal{T}^\pi W_2||_\infty &\le \gamma ||W_1 - W_2||_\infty, \\ ||\mathcal{T} W_1 - \mathcal{T} W_2||_\infty &\le \gamma ||W_1 - W_2||_\infty. \end{aligned}$$

Indeed, for all $x \in X$,

$$\begin{aligned} |\mathcal{T} W_1(x) - \mathcal{T} W_2(x)| &= \Big| \max_a \big[r(x,a) + \gamma \sum_y p(y|x,a) W_1(y) \big] \\ &\quad - \max_a \big[r(x,a) + \gamma \sum_y p(y|x,a) W_2(y) \big] \Big| \\ &\le \gamma \max_a \sum_y p(y|x,a) |W_1(y) - W_2(y)| \\ &\le \gamma ||W_1 - W_2||_\infty \end{aligned}$$

Properties of the value functions

Proposition 3.

1. *V^π is the unique fixed-point of $\mathcal{T}^\pi$*

$$V^\pi = \mathcal{T}^\pi V^\pi.$$

2. *V^* is the unique fixed-point of $\mathcal{T}$:*

$$V^* = \mathcal{T} V^*.$$

3. *For any policy π, we have $V^\pi = (I - \gamma P^\pi)^{-1} r^\pi$*
4. *The policy defined by*

$$\pi^*(x) \in \arg\max_{a \in A} \Big[r(x,a) + \gamma \sum_y p(y|x,a) V^*(y) \Big]$$

is optimal (and stationary)

Proof of Proposition 3

1. From Proposition 1, V^π is a fixed point of $\mathcal{T}^\pi$. Uniqueness comes from the contraction property of $\mathcal{T}^\pi$.
2. Idem for V^*.
3. $V^\pi = \mathcal{T}^\pi V^\pi = r^\pi + \gamma P^\pi V^\pi$. Thus $(I - \gamma P^\pi)V^\pi = r^\pi$. Now P^π is a stochastic matrix (whose eingenvalues have a modulus ≤ 1), thus the eing. of $(I - \gamma P^\pi)$ have a modulus $\geq 1 - \gamma > 0$, thus is invertible.
4. From the definition of π^*, we have

$$\mathcal{T}^{\pi^*} V^* = \mathcal{T} V^* = V^*$$

   Thus V^* is the fixed-point of $\mathcal{T}^{\pi^*}$. But, by definition, V^{π^*} is the fixed-point of $\mathcal{T}^{\pi^*}$ and since there is uniqueness of the fixed-point, $V^{\pi^*} = V^*$ and π^* is optimal.

Value Iteration

Proposition 4.

- *For any bounded* π *and* V_0*, define* $V_{k+1} = \mathcal{T}^\pi V_k$*. Then* $V_k \to V^\pi$*.*
- *For any bounded* V_0*, define* $V_{k+1} = \mathcal{T} V_k$*. Then* $V_k \to V^*$*.*

Proof.

$$||V_{k+1} - V^*|| = ||\mathcal{T} V_k - \mathcal{T} V^*|| \leq \gamma ||V_k - V^*|| \leq \gamma^{k+1} ||V_0 - V^*|| \to 0$$

(idem for V^π) $\square$

Variant: asynchronous iterations

Policy Iteration

Choose any initial policy π_0. Iterate:

1. **Policy evaluation**: compute V^{π_k}.
2. **Policy improvement**: π_{k+1} greedy w.r.t. V^{π_k}:

$$\pi_{k+1}(x) \in \arg\max_{a \in A} \Big[r(x,a) + \gamma \sum_y p(y|x,a) V^{\pi_k}(y) \Big],$$

(i.e. $\pi_{k+1} \in \arg\max_\pi \mathcal{T}^\pi V^{\pi_k}$)

Stop when $V^{\pi_k} = V^{\pi_{k+1}}$.

Proposition 5.

Policy iteration generates a sequence of policies with increasing performance ($V^{\pi_{k+1}} \geq V^{\pi_k}$) and (in the case of finite state and action spaces) terminates in a finite number of steps with the optimal policy π^*.

Proof of Proposition 5

From the definition of the operators $\mathcal{T}$, $\mathcal{T}^{\pi_k}$, $\mathcal{T}^{\pi_{k+1}}$ and from π_{k+1},

$$V^{\pi_k} = \mathcal{T}^{\pi_k} V^{\pi_k} \leq \mathcal{T} V^{\pi_k} = \mathcal{T}^{\pi_{k+1}} V^{\pi_k}, \tag{2}$$

and from the monotonicity of $\mathcal{T}^{\pi_{k+1}}$, we have

$$V^{\pi_k} \leq \lim_{n\to\infty} (\mathcal{T}^{\pi_{k+1}})^n V^{\pi_k} = V^{\pi_{k+1}}.$$

Thus $(V^{\pi_k})_k$ is a non-decreasing sequence. Since there is a finite number of possible policies (finite state and action spaces), the stopping criterion holds for a finite k; We thus have equality in (2), thus

$$V^{\pi_k} = \mathcal{T} V^{\pi_k}$$

so $V^{\pi_k} = V^*$ and π_k is an optimal policy.

Back to Reinforcement Learning

What if the transition probabilities $p(y|x,a)$ and the reward functions $r(x,a)$ are unknown?
In DP, we used their knowledge

- in value iteration:

$$V_{k+1}(x) = \mathcal{T} V_k(x) = \max_a \left[r(x,a) + \gamma \sum_y p(y|x,a) V_k(y) \right].$$

- in policy iteration:
  - when computing V^{π_k} (which requires iterating $\mathcal{T}^{\pi_k}$)
  - when computing the greedy policy:

$$\pi_{k+1}(x) \in \arg\max_{a \in A} \left[r(x,a) + \gamma \sum_y p(y|x,a) V^{\pi_k}(y) \right],$$

RL $=$ introduction of 2 ideas: **Q-functions** and **sampling**.

Definition of the Q-value function

Define the Q-value function $Q^\pi : X \times A \to \boldsymbol{R}$: for a policy π,

$$Q^\pi(x,a) = \mathbb{E}\big[\sum_{t\geq 0} \gamma^t r(x_t,a_t) | x_0 = x, a_0 = a, a_t = \pi(x_t), t \geq 1\big]$$

and the optimal Q-value function $Q^*(x,a) = \max_\pi Q^\pi(x,a)$.

Proposition 6.

Q^π and Q^* satisfy the Bellman equations:

$$\begin{aligned} Q^\pi(x,a) &= r(x,a) + \gamma \sum_{y\in X} p(y|x,a) Q^\pi(y,\pi(y)) \\ Q^*(x,a) &= r(x,a) + \gamma \sum_{y\in X} p(y|x,a) \max_{b\in A} Q^\pi(y,b) \end{aligned}$$

Idea: compute Q^* and then $\pi^*(x) \in \arg\max_a Q^*(x,a)$.

Q-learning algorithm [Watkins, 1989]

Builds a sequence of Q-value functions Q_k.
Whenever a transition $x_t, a_t \xrightarrow{r_t} x_{t+1}$ occurs, update the Q-value:

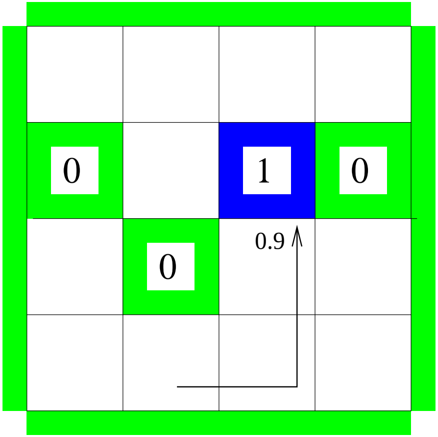
$$Q_{k+1}(x_t, a_t) = Q_k(x_t, a_t) + \eta_k(x_t, a_t) \underbrace{\Big[r_t + \gamma \max_{b \in A} Q_k(x_{t+1}, b) - Q_k(x_t, a_t) \Big]}_{\text{temporal difference}}.$$

Proposition 7 (Watkins et Dayan, 1992).
Assume that all state-action pairs (x, a) *are visited infinitely often and that the learning steps satisfy for all* x, a,
$\sum_{k \geq 0} \eta_k(x, a) = \infty$, $\sum_{k \geq 0} \eta_k^2(x, a) < \infty$, *then* $Q_k \xrightarrow{a.s.} Q^*$.

The proof relies on Stochastic Approximation for estimating the fixed-point of a contraction mapping.

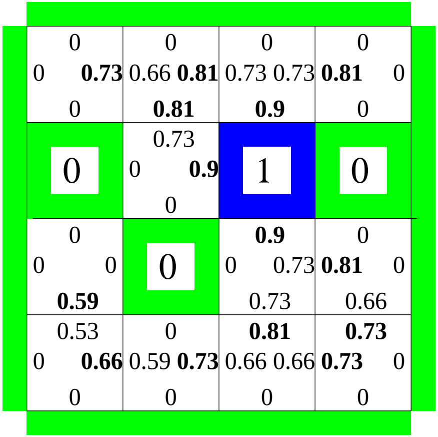
Q-learning algorithm

Deterministic case, discount factor $\gamma = 0.9$. Take steps $\eta = 1$.



After transition $x, a \xrightarrow{r} y$ update $Q_{k+1}(x, a) = r + \gamma \max_{b \in A} Q_k(y, b)$

Optimal Q-values



Bellman's equation: $Q^*(x,a) = \gamma \max_{b \in A} Q^*(\text{next-state}(x,a), b)$.

First conclusions

When the state-space is finite and "small":

- If transition probabilities and rewards are known, then DP algorithms (value iteration, policy iteration) compute the optimal solution
- Otherwise, use sampling techniques and RL algorithms (Q-learning, TD(λ)) apply

2 main issues:

- Usually state-space is large (infinite)! We need to build **approximate solutions**.
- We need to design clever **exploration strategies**.

Introduction to Reinforcement Learning and multi-armed bandits

Rémi Munos

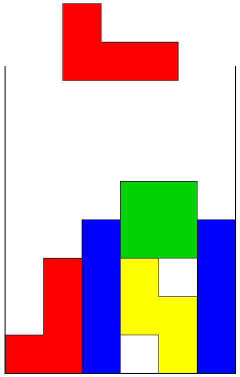
INRIA Lille - Nord Europe
Currently on leave at MSR-NE
http://researchers.lille.inria.fr/∼munos/

NETADIS Summer School 2013, Hillerod, Denmark

Part 2: Reinforcement Learning and dynamic programming with function approximation

- Approximate policy iteration
- Approximate value iteration
- Analysis of sample-based algorithms

Example: Tetris



- **State**: wall configuration + new piece
- **Action**: posible positions of the new piece on the wall,
- **Reward**: number of lines removed
- **Next state**: Resulting configuration of the wall + random new piece.

Size state space: $\approx 10^{61}$ states!

Approximate methods

When the state space is finite and small, use DP or RL techniques. However in most interesting problems, the state-space X is huge, possibly infinite:

- Tetris, Backgammon, ...
- Control problems often consider continuous spaces

We need to use function approximation:

- Linear approximation $\mathcal{F} = \{f_\alpha = \sum_{i=1}^{d} \alpha_i \phi_i, \alpha \in \boldsymbol{R}^d\}$
- Neural networks: $\mathcal{F} = \{f_\alpha\}$, where α is the weight vector
- Non-parametric: k-nearest neighboors, Kernel methods, SVM, ...

Write $\mathcal{F}$ the set of representable functions.

Approximate dynamic programming

General approach: build an approximation $V \in \mathcal{F}$ of the optimal value function V^* (which may not belong to $\mathcal{F}$), and then consider the policy π greedy policy w.r.t. V, i.e.,

$$\pi(x) \in \arg\max_{a \in A} \big[r(x,a) + \gamma \sum_y p(y|x,a) V(y) \big].$$

(for the case of *infinite horizon with discounted rewards.*)

We expect that if $V \in \mathcal{F}$ is close to V^* then the policy π will be close-to-optimal.

Bound on the performance loss

Proposition 1.

Let V be an approximation of V^*, and write π the policy greedy w.r.t. V. Then

$$||V^* - V^\pi||_\infty \leq \frac{2\gamma}{1-\gamma}||V^* - V||_\infty.$$

Proof.

From the contraction properties of the operators $\mathcal{T}$ and $\mathcal{T}^\pi$ and that by definition of π we have $\mathcal{T}V = \mathcal{T}^\pi V$, we deduce

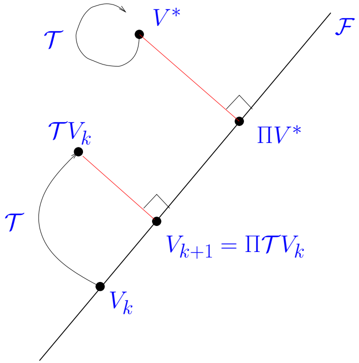
$$\begin{aligned}
\|V^* - V^\pi\|_\infty &\leq \|V^* - \mathcal{T}^\pi V\|_\infty + \|\mathcal{T}^\pi V - \mathcal{T}^\pi V^\pi\|_\infty \\
&\leq \|\mathcal{T}V^* - \mathcal{T}V\|_\infty + \gamma\|V - V^\pi\|_\infty \\
&\leq \gamma\|V^* - V\|_\infty + \gamma(\|V - V^*\|_\infty + \|V^* - V^\pi\|_\infty) \\
&\leq \frac{2\gamma}{1-\gamma}\|V^* - V\|_\infty.
\end{aligned}$$

Approximate Value Iteration

Approximate Value Iteration: builds a sequence of $V_k \in \mathcal{F}$:

$$V_{k+1} = \Pi \mathcal{T} V_k,$$

where Π is a projection operator onto $\mathcal{F}$ (under some norm $\|\cdot\|$).



Property: the algorithm may not converge.

Performance bound for AVI

Apply AVI for K iterations.

Proposition 2 (Bertsekas & Tsitsiklis, 1996).

The performance loss $\|V^* - V^{\pi_K}\|_\infty$ resulting from using the policy π_K greedy w.r.t. V_K is bounded as:

$$\|V^* - V^{\pi_K}\|_\infty \le \frac{2\gamma}{(1-\gamma)^2} \max_{0\le k<K} \underbrace{\|\mathcal{T} V_k - V_{k+1}\|_\infty}_{\textit{projection error}} + \frac{2\gamma^{K+1}}{1-\gamma}\|V^* - V_0\|_\infty .$$

Proof of Proposition 2

Write $\varepsilon = \max_{0 \leq k < K} \|\mathcal{T} V_k - V_{k+1}\|_\infty$. For all $0 \leq k < K$, we have

$$\begin{aligned} \|V^* - V_{k+1}\|_\infty &\leq \|\mathcal{T} V^* - \mathcal{T} V_k\|_\infty + \|\mathcal{T} V_k - V_{k+1}\|_\infty \\ &\leq \gamma \|V^* - V_k\|_\infty + \varepsilon, \end{aligned}$$

thus, $$\begin{aligned} \|V^* - V_K\|_\infty &\leq (1 + \gamma + \cdots + \gamma^{K-1})\varepsilon + \gamma^K \|V^* - V_0\|_\infty \\ &\leq \frac{1}{1-\gamma}\varepsilon + \gamma^K \|V^* - V_0\|_\infty \end{aligned}$$

and we conclude by using Proposition 1.

A possible numerical implementation

Makes use of a generative model. At each round k,

1. Sample n states $(x_i)_{1\leq i\leq n}$
2. From each state x_i, for each action $a \in A$, use the model to generate a reward $r(x_i, a)$ and m next-state samples $(y^j_{i,a})_{1\leq j\leq m} \sim p(\cdot|x_i, a)$
3. Define

$$V_{k+1} = \arg\min_{V\in\mathcal{F}} \max_{1\leq i\leq n} \Big| V(x_i) - \underbrace{\max_{a\in A}\Big[r(x_i,a) + \gamma\frac{1}{m}\sum_{j=1}^{m} V_k(y^j_{i,a})\Big]}_{\text{sample estimate of } \mathcal{T}V_k(x_i)} \Big|$$

This is still a numerically hard problem.

Approximate Policy Iteration

Choose an initial policy π_0 and iterate:

1. **Approximate policy evaluation** of π_k:
   compute an approximation V_k of V^{π_k}.
2. **Policy improvement**: π_{k+1} is greedy w.r.t. V_k:

$$\pi_{k+1}(x) \in \arg\max_{a \in A} \Big[r(x,a) + \gamma \sum_{y \in X} p(y|x,a) V_k(y) \Big].$$

Property: the algorithm may not converge.

Performance bound for API

Proposition 3 (Bertsekas & Tsitsiklis, 1996).

We have

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

Thus if we are able to compute a good approximation of the value function V^{π_k} at each iteration then the performance of the resulting policies will be good.

Proof of Proposition 3 [part 1]

Write $e_k = V_k - V^{\pi_k}$ the *approximation error*, $g_k = V^{\pi_{k+1}} - V^{\pi_k}$ the *performance gain* between iterations k and $k+1$, and $l_k = V^* - V^{\pi_k}$ the loss of using policy π_k instead of π^*.
The next policy cannot be much worst that the current one:

$$g_k \geq -\gamma(I - \gamma P^{\pi_{k+1}})^{-1}(P^{\pi_{k+1}} - P^{\pi_k})\, e_k \qquad (1)$$

Indeed, since $T^{\pi_{k+1}} V_k \geq T^{\pi_k} V_k$ (as π_{k+1} is greedy w.r.t. V_k), we have:

$$\begin{aligned} g_k &= T^{\pi_{k+1}} V^{\pi_{k+1}} - T^{\pi_{k+1}} V^{\pi_k} + T^{\pi_{k+1}} V^{\pi_k} - T^{\pi_{k+1}} V_k \\ &\quad + T^{\pi_{k+1}} V_k - T^{\pi_k} V_k + T^{\pi_k} V_k - T^{\pi_k} V^{\pi_k} \\ &\geq \gamma P^{\pi_{k+1}} g_k - \gamma(P^{\pi_{k+1}} - P^{\pi_k})\, e_k \\ &\geq -\gamma(I - \gamma P^{\pi_{k+1}})^{-1}(P^{\pi_{k+1}} - P^{\pi_k})\, e_k \end{aligned}$$

Proof of Proposition 3 [part 2]

The loss at the next iteration is bounded by the current loss as:

$$l_{k+1} \leq \gamma P^{\pi^*} l_k + \gamma[P^{\pi_{k+1}}(I - \gamma P^{\pi_{k+1}})^{-1}(I - \gamma P^{\pi_k}) - P^{\pi^*}]e_k$$

Indeed, since $T^{\pi^*} V_k \leq T^{\pi_{k+1}} V_k$,

$$\begin{aligned} l_{k+1} &= T^{\pi^*} V^* - T^{\pi^*} V^{\pi_k} + T^{\pi^*} V^{\pi_k} - T^{\pi^*} V_k \\ &\quad + T^{\pi^*} V_k - T^{\pi_{k+1}} V_k + T^{\pi_{k+1}} V_k - T^{\pi_{k+1}} V^{\pi_k} \\ &\quad + T^{\pi_{k+1}} V^{\pi_k} - T^{\pi_{k+1}} V^{\pi_{k+1}} \\ &\leq \gamma[P^{\pi^*} l_k - P^{\pi_{k+1}} g_k + (P^{\pi_{k+1}} - P^{\pi^*}) e_k] \end{aligned}$$

and by using (1),

$$\begin{aligned} l_{k+1} &\leq \gamma P^{\pi^*} l_k + \gamma[P^{\pi_{k+1}}(I - \gamma P^{\pi_{k+1}})^{-1}(P^{\pi_{k+1}} - P^{\pi_k}) + P^{\pi_{k+1}} - P^{\pi^*}]e_k \\ &\leq \gamma P^{\pi^*} l_k + \gamma[P^{\pi_{k+1}}(I - \gamma P^{\pi_{k+1}})^{-1}(I - \gamma P^{\pi_k}) - P^{\pi^*}]e_k. \end{aligned}$$

Proof of Proposition 3 [part 3]

Writing $f_k = \gamma[P^{\pi_{k+1}}(I - \gamma P^{\pi_{k+1}})^{-1}(I - \gamma P^{\pi_k}) - P^{\pi^*}]e_k$, we have:

$$l_{k+1} \leq \gamma P^{\pi^*} l_k + f_k.$$

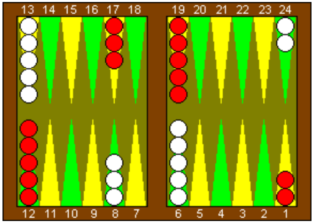
Thus, by taking the limit sup.,

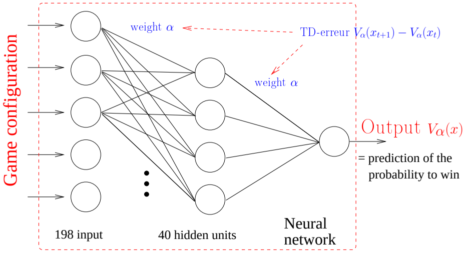
$$\begin{aligned}(I - \gamma P^{\pi^*}) \limsup_{k\to\infty} l_k &\leq \limsup_{k\to\infty} f_k \\ \limsup_{k\to\infty} l_k &\leq (I - \gamma P^{\pi^*})^{-1} \limsup_{k\to\infty} f_k,\end{aligned}$$

since $I - \gamma P^{\pi^*}$ is invertible. In L_∞-norm, we have

$$\begin{aligned}\limsup_{k\to\infty} ||l_k|| &\leq \frac{\gamma}{1-\gamma} \limsup_{k\to\infty} ||P^{\pi_{k+1}}(I - \gamma P^{\pi_{k+1}})^{-1}(I + \gamma P^{\pi_k}) + P^{\pi^*}|| \, ||e_k|| \\ &\leq \frac{\gamma}{1-\gamma}(\frac{1+\gamma}{1-\gamma} + 1) \limsup_{k\to\infty} ||e_k|| = \frac{2\gamma}{(1-\gamma)^2} \limsup_{k\to\infty} ||e_k||.\end{aligned}$$

Case study: TD-Gammon [Tesauro, 1994]





State = game configuration x + player $j \to N \simeq 10^{20}$.
Reward 1 or 0 at the end of the game.

The neural network returns an approximation of $V^*(x, j)$: probability that player j wins from position x, assuming that both players play optimally.

TD-Gammon algorithm

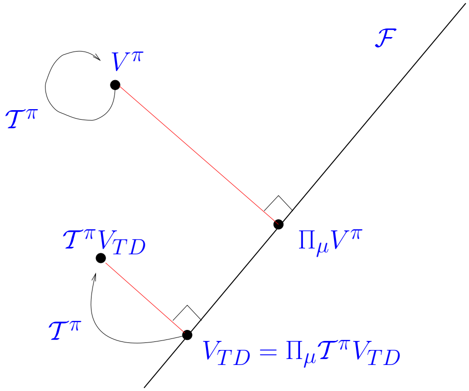
- At time t, the current game configuration is x_t
- Roll dices and select the action that maximizes the value V_α of the resulting state x_{t+1}
- Set the temporal difference $d_t = V_\alpha(x_{t+1}, j_{t+1}) - V_\alpha(x_t, j_t)$ (if this is a final position, replace $V_\alpha(x_{t+1}, j_{t+1})$ by $+1$ or 0)
- Update α_t according to a gradient descent

$$\alpha_{t+1} = \alpha_t + \eta_t d_t \sum_{0 \le s \le t} \lambda^{t-s} \nabla_\alpha V_\alpha(x_s).$$

After several weeks of self playing $\rightarrow$ **world best player.**
According to human experts it developed new strategies, specially in openings.

Least Squares Temporal Difference (LSTD)

[Bradtke & Barto, 1996] Consider a linear space $\mathcal{F}$.
Let Π_μ be the projection onto $\mathcal{F}$ defined by a weighted norm $L_2(\mu)$.
The **Least Squares Temporal Difference** solution V_{TD} is the fixed-point of $\Pi_\mu T^\pi$.



Performance bound for LSTD

In general, no guarantee that there exists a fixed-point to $\Pi_\mu \mathcal{T}^\pi$ (since $\mathcal{T}^\pi$ is not a contraction in $L_2(\mu)$-norm).
However, when μ is the stationary distribution associated to π (i.e., such that $\mu P^\pi = \mu$), then there exists a unique LSTD solution.

Proposition 4.

Consider μ to be the stationary distribution associated to π. Then $\mathcal{T}^\pi$ is a contraction mapping in $L_2(\mu)$-norm, thus $\Pi_\mu \mathcal{T}^\pi$ is also a contraction, and there exists a unique LSTD solution V_{TD}. In addition, we have the approximation error:

$$\|V^\pi - V_{TD}\|_\mu \leq \frac{1}{\sqrt{1-\gamma^2}} \inf_{V \in \mathcal{F}} \|V^\pi - V\|_\mu. \qquad (2)$$

Proof of Proposition 4 [part 1]

First let us prove that $\|P_\pi\|_\mu = 1$. We have:

$$\begin{aligned} \|P^\pi V\|_\mu^2 &= \sum_x \mu(x)\big(\sum_y p(y|x,\pi(x))V(y)\big)^2 \\ &\leq \sum_x \sum_y \mu(x)p(y|x,\pi(x))V(y)^2 \\ &= \sum_y \mu(y)V(y)^2 = \|V\|_\mu^2. \end{aligned}$$

We deduce that $\mathcal{T}^\pi$ is a contraction mapping in $L_2(\mu)$:

$$\|\mathcal{T}^\pi V_1 - \mathcal{T}^\pi V_2\|_\mu = \gamma\|P^\pi(V_1 - V_2)\|_\mu \leq \gamma\|V_1 - V_2\|_\mu,$$

and since Π_μ is a non-expansion in $L_2(\mu)$, then $\Pi_\mu\mathcal{T}^\pi$ is a contraction in $L_2(\mu)$. Write V_{TD} its (unique) fixed-point.

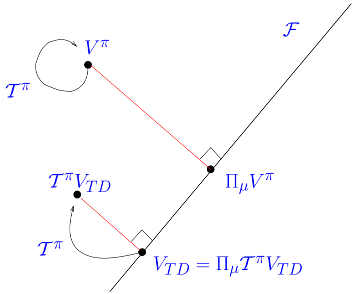
Proof of Proposition 4 [part 2]

We have $\|V^\pi - V_{TD}\|_\mu^2 = \|V^\pi - \Pi_\mu V^\pi\|_\mu^2 + \|\Pi_\mu V^\pi - V_{TD}\|_\mu^2$,

$$\begin{aligned}\text{but } \|\Pi_\mu V^\pi - V_{TD}\|_\mu^2 &= \|\Pi_\mu V^\pi - \Pi_\mu \mathcal{T}^\pi V_{TD}\|_\mu^2 \\ &\le \|\mathcal{T}^\pi V^\pi - \mathcal{T} V_{TD}\|_\mu^2 \le \gamma^2 \|V^\pi - V_{TD}\|_\mu^2.\end{aligned}$$

$$\text{Thus } \|V^\pi - V_{TD}\|_\mu^2 \le \|V^\pi - \Pi_\mu V^\pi\|_\mu^2 + \gamma^2 \|V^\pi - V_{TD}\|_\mu^2,$$

from which the result follows.



Characterization of the LSTD solution

The Bellman residual $\mathcal{T}^\pi V_{TD} - V_{TD}$ is orthogonal to the space $\mathcal{F}$, thus for all $1 \leq i \leq d$,

$$\begin{aligned} \langle r^\pi + \gamma P^\pi V_{TD} - V_{TD}, \phi_i \rangle_\mu &= 0 \\ \langle r^\pi, \phi_i \rangle_\mu + \sum_{j=1}^{d} \langle \gamma P^\pi \phi_j - \phi_j, \phi_i \rangle_\mu \alpha_{TD,j} &= 0, \end{aligned}$$

where α_{TD} is the parameter of V_{TD}. We deduce that α_{TD} is solution to the linear system (of size d):

$$A\alpha = b, \text{ with} \left\{ \begin{array}{lcl} A_{i,j} &=& \langle \phi_i, \phi_j - \gamma P^\pi \phi_j \rangle_\mu \\ b_i &=& \langle \phi_i, r^\pi \rangle_\mu \end{array} \right.$$

Empirical LSTD

Consider a trajectory $(x_1, x_2, \ldots, x_n)$ generated by following π
Build the matrix $\hat{A}$ and the vector $\hat{b}$ as

$$\begin{aligned}\hat{A}_{ij} &= \frac{1}{n}\sum_{t=1}^{n} \phi_i(x_t)[\phi_j(x_t) - \gamma\phi_j(x_{t+1})],\\ \hat{b}_i &= \frac{1}{n}\sum_{t=1}^{n} \phi_i(x_t) r_{x_t}.\end{aligned}$$

and compute the empirical LSTD solution $\hat{V}_{TD}$ whose parameter is the solution to $\hat{A}\alpha = \hat{b}$.

We have $\hat{V}_{TD} \overset{a.s.}{\to} V_{TD}$ when $n \to \infty$, since $\hat{A} \overset{a.s.}{\to} A$ and $\hat{b} \overset{a.s.}{\to} b$.

Finite-time analysis of LSTD

Define the empirical norm $\|f\|_n = \sqrt{\frac{1}{n}\sum_{t=1}^n f(x_t)^2}$.

Theorem 1 (Lazaric et al., 2010).
With probability $1-\delta$ *(w.r.t. the trajectory),*

$$||V^\pi - \hat{V}_{TD}||_n \leq \frac{1}{\sqrt{1-\gamma^2}} \underbrace{\inf_{V\in\mathcal{F}} ||V^\pi - V||_n}_{\textit{Approximation error}} + \frac{c}{1-\gamma} \underbrace{\sqrt{\frac{d\log(1/\delta)}{n}}}_{\textit{Estimation error}}$$

This type of bounds is similar to results in Statistical Learning.

Least-Squares Policy Iteration

[Lagoudakis & Parr, 2003] Consider $Q(x,a) = \sum_{i=1}^{d} \alpha_i \phi_i(x,a)$

- **Policy evaluation**: At round k, run a trajectory $(x_t)_{1\le t\le n}$ by following policy π_k. Build $\hat{A}$ and $\hat{b}$ as

$$\begin{aligned}\hat{A}_{ij} &= \frac{1}{n}\sum_{t=1}^{n} \phi_i(x_t,a_t)[\phi_j(x_t,a_t) - \gamma\phi_j(x_{t+1},a_{t+1})],\\ \hat{b}_i &= \frac{1}{n}\sum_{t=1}^{n} \phi_i(x_t,a_t) r(x_t,a_t).\end{aligned}$$

  and $\hat{Q}_k$ is the Q-function defined by the solution to $\hat{A}\alpha = \hat{b}$.
- **Policy improvement**:$\pi_{k+1}(x) \in \arg\max_{a\in A} \hat{Q}_k(x,a)$.

We would like guarantees on $\|Q^* - Q^{\pi_K}\|$

Theoretical guarantees so far

Approximate Value Iteration:

$$\|V^* - V^{\pi_K}\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2} \max_{0\leq k<K} \underbrace{\|\mathcal{T} V_k - V_{k+1}\|_\infty}_{\text{projection error}} + O(\gamma^K).$$

Approximate Policy Iteration:

$$\|V^* - V^{\pi_K}\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2} \max_{0\leq k<K} \underbrace{\|V^{\pi_k} - V_k\|_\infty}_{\text{approximation error}} + O(\gamma^K).$$

Problem: hard to control L_∞-norm using samples. We could minimize an empirical L_∞-norm, but

- Numerically intractable
- Hard to relate L_∞-norm to empirical L_∞-norm.

Instead use empirical L_2-norm

- For AVI this is just a linear regression problem:

$$V_{k+1} = \arg\min_{V \in \mathcal{F}} \sum_{i=1}^{n} \left| \widehat{\mathcal{T} V}_k(x_i) - V(x_i) \right|^2,$$

- For API this is just LSTD: fixed-point of an empirical Bellman operator projected onto $\mathcal{F}$ using an empirical norm.

In both cases, V_k is solution to a linear problem, which is

- Numerically tractable
- For which generalization bounds exits (using VC theory):

$$\|\mathcal{T} V_k - V_{k+1}\|_2^2 \leq \frac{1}{n} \sum_{i=1}^{n} \left| \widehat{\mathcal{T} V}_k(x_i) - V(x_i) \right|^2 + c\sqrt{\frac{VC(\mathcal{F})}{n}}$$

L_p-norm analysis of ADP

Under smoothness assumptions on the MDP, the propagation error of all usual ADP algorithms can be analyzed in L_p-norm ($p \geq 1$).

Proposition 5 (Munos, 2003, 2007).

- **Approximate Value Iteration:** *Assume there is a constant $C \geq 1$ and a distribution μ such that $\forall x \in X$, $\forall a \in A$,*

$$p(\cdot|x,a) \leq C\mu(\cdot).$$

$$\|V^* - V^{\pi_K}\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2} C^{1/p} \max_{0 \leq k < K} \|\mathcal{T}V_k - V_{k+1}\|_{p,\mu} + O(\gamma^K).$$

- **Approximate Policy Iteration:** *Assume $p(\cdot|x,a) \leq C\mu_\pi(\cdot)$, for any policy π*

$$\|V^* - V^{\pi_K}\|_\infty \leq \frac{2\gamma}{(1-\gamma)^2} C^{1/p} \max_{0 \leq k < K} \|V_k - V^{\pi_k}\|_{p,\mu_\pi} + O(\gamma^K).$$

We have all ingredients for a finite-sample analysis of RL/ADP.

Finite-sample analysis of LSPI

Perform K policy iterations steps. At stage k, run one trajectory of length n following π_k and compute the LSTD solution $\hat{V}_k$ (by solving a linear system).

Proposition 6 (Lazaric et al., 2010).

For any $\delta > 0$, with probability at least $1 - \delta$, we have:

$$\begin{aligned}||V^* - V^{\pi_K}||_\infty \quad \leq \quad & \frac{2\gamma}{(1-\gamma)^3} C^{1/2} \sup_k \inf_{V \in \mathcal{F}} \|V^{\pi_k} - V\|_{2,\mu_k} \\ & + O\Big(\frac{d \log(1/\delta)}{n}\Big)^{1/2} + O(\gamma^K)\end{aligned}$$

Finite-sample analysis of AVI

K iterations of AVI with n samples $x_i \sim \mu$. From each state x_i, each $a \in A$, generate m next state samples $y^j_{i,a} \sim p(\cdot|x_i, a)$.

Proposition 7 (Munos and Szepesvári, 2007).

For any $\delta > 0$, with probability at least $1 - \delta$, we have:

$$\begin{aligned} ||V^* - V^{\pi_K}||_\infty \quad \leq \quad & \frac{2\gamma}{(1-\gamma)^2} C^{1/p} \, d(\mathcal{T}\mathcal{F}, \mathcal{F}) + O(\gamma^K) \\ & + O\Big(\frac{V(\mathcal{F}) \log(1/\delta)}{n}\Big)^{1/4} + O\Big(\frac{\log(1/\delta)}{m}\Big)^{1/2}, \end{aligned}$$

where $d(\mathcal{T}\mathcal{F}, \mathcal{F}) \stackrel{\text{def}}{=} \sup_{g \in \mathcal{F}} \inf_{f \in \mathcal{F}} ||\mathcal{T}g - f||_{2,\mu}$ is the Bellman residual of the space $\mathcal{F}$, and $V(\mathcal{F})$ the pseudo-dimension of $\mathcal{F}$.

More works on finite-sample analysis of ADP/RL

This is important to know how many samples n are required to build an ϵ-approximation of the optimal policy.

- Policy iteration using a single trajectory [Antos et al., 2008]
- BRM [Maillard et al., 2010]
- LSTD with random projections [Ghavamzadeh et al., 2010]
- Lasso-TD [Ghavamzadeh et al., 2011]

Active research topic which links RL and statistical learning theory.

Introduction to Reinforcement Learning and multi-armed bandits

Rémi Munos

INRIA Lille - Nord Europe
Currently on leave at MSR-NE
http://researchers.lille.inria.fr/∼munos/

NETADIS Summer School 2013, Hillerod, Denmark

Outline of Part 3

Exploration for sequential decision making:
Application to games, optimization, and planning

- The stochastic bandit: UCB
- The adversarial bandit: EXP3
- Populations of bandits
  - Computation of equilibrium in games. Application to Poker
  - Hierarchical bandits. MCTS and application to Go.
- Optimism for decision making
  - Lipschitz optimization
  - Lipschitz bandits
  - Optimistic planning in MDPs

The stochastic multi-armed bandit problem

Setting:

- Set of K arms, defined by distributions ν_k (with support in $[0,1]$), whose law is unknown,
- At each time t, choose an arm k_t and receive reward $x_t \overset{i.i.d.}{\sim} \nu_{k_t}$.
- **Goal**: find an arm selection policy such as to maximize the expected sum of rewards.

Exploration-exploitation tradeoff:

- **Explore**: learn about the environment
- **Exploit**: act optimally according to our current beliefs

The regret

Definitions:

- Let $\mu_k = \mathbb{E}[\nu_k]$ be the expected value of arm k,
- Let $\mu^* = \max_k \mu_k$ the best expected value,
- The cumulative expected **regret**:

$$R_n \stackrel{\text{def}}{=} \sum_{t=1}^{n} \mu^* - \mu_{k_t} = \sum_{k=1}^{K} (\mu^* - \mu_k) \sum_{t=1}^{n} \mathbf{1}\{k_t = k\} = \sum_{k=1}^{K} \Delta_k n_k,$$

where $\Delta_k \stackrel{\text{def}}{=} \mu^* - \mu_k$, and n_k the number of times arm k has been pulled up to time n.

Goal: Find an arm selection policy such as to minimize R_n.

Proposed solutions

This is an old problem! [Robbins, 1952] Maybe surprisingly, not fully solved yet!
Many proposed strategies:

- ϵ-**greedy exploration**: choose apparent best action with proba $1 - \epsilon$, or random action with proba ϵ,
- **Bayesian exploration**: assign prior to the arm distributions and select arm according to the posterior distributions (Gittins index, Thompson strategy, ...)
- **Softmax exploration**: choose arm k with proba $\propto \exp(\beta \widehat{X}_k)$ (ex: EXP3 algo)
- **Follow the perturbed leader**: choose best perturbed arm
- **Optimistic exploration**: select arm with highest upper bound

The UCB algorithm

Upper Confidence Bound algorithm [Auer, Cesa-Bianchi, Fischer, 2002]: at each time n, select the arm k with highest $B_{k,n_k,n}$ value:

$$B_{k,n_k,n} \stackrel{\text{def}}{=} \underbrace{\frac{1}{n_k}\sum_{s=1}^{n_k} x_{k,s}}_{\widehat{X}_{k,n_k}} + \underbrace{\sqrt{\frac{3\log(n)}{2n_k}}}_{c_{n_k,n}},$$

with:

- n_k is the number of times arm k has been pulled up to time n
- $x_{k,s}$ is the s-th reward received when pulling arm k.

Note that

- Sum of an *exploitation term* and an *exploration term*.
- $c_{n_k,n}$ is a confidence interval term, so $B_{k,n_k,n}$ is a UCB.

Intuition of the UCB algorithm

Idea:

- **"Optimism in the face of uncertainty"** principle
- Select the arm with highest upper bound (on the true value of the arm, given what has been observed so far).
- The B-values $B_{k,s,t}$ are UCBs on μ_k. Indeed:

$$\begin{aligned}
\mathbb{P}\Big(\widehat{X}_{k,s} - \mu_k \geq \sqrt{\frac{3\log(t)}{2s}}\Big) &\leq \frac{1}{t^3}, \\
\mathbb{P}\Big(\widehat{X}_{k,s} - \mu_k \leq -\sqrt{\frac{3\log(t)}{2s}}\Big) &\leq \frac{1}{t^3}
\end{aligned}$$

Reminder of Chernoff-Hoeffding inequality:

$$\begin{aligned}
\mathbb{P}(\widehat{X}_{k,s} - \mu_k \geq \epsilon) &\leq e^{-2s\epsilon^2} \\
\mathbb{P}(\widehat{X}_{k,s} - \mu_k \leq -\epsilon) &\leq e^{-2s\epsilon^2}
\end{aligned}$$

Regret bound for UCB

Proposition 1.

Each sub-optimal arm k is visited in average, at most:

$$\mathbb{E}n_k(n) \leq 6\frac{\log n}{\Delta_k^2} + 1 + \frac{\pi^2}{3}$$

times (where $\Delta_k \stackrel{\text{def}}{=} \mu^* - \mu_k > 0$).

Thus the expected regret is bounded by:

$$\mathbb{E}R_n = \sum_k \mathbb{E}[n_k]\Delta_k \leq 6 \sum_{k:\Delta_k>0} \frac{\log n}{\Delta_k} + K(1 + \frac{\pi^2}{3}).$$

Intuition of the proof

Let k be a sub-optimal arm, and k^* be an optimal arm. At time n, if arm k is selected, this means that

$$\begin{aligned}
B_{k,n_k,n} &\geq B_{k^*,n_{k^*},n} \\
\widehat{X}_{k,n_k} + \sqrt{\frac{3\log(n)}{2n_k}} &\geq \widehat{X}_{k^*,n_{k^*}} + \sqrt{\frac{3\log(n)}{2n_{k^*}}} \\
\mu_k + 2\sqrt{\frac{3\log(n)}{2n_k}} &\geq \mu^*, \text{ with high proba} \\
n_k &\leq \frac{6\log(n)}{\Delta_k^2}
\end{aligned}$$

Thus, if $n_k > \frac{6\log(n)}{\Delta_k^2}$, then there is only a small probability that arm k be selected.

Proof of Proposition 1

Write $u = \frac{6\log(n)}{\Delta_k^2} + 1$. We have:

$$\begin{aligned} n_k(n) &\le u + \sum_{t=u+1}^{n} \mathbf{1}\{k_t = k; n_k(t) > u\} \\ &\le u + \sum_{t=u+1}^{n} \Big[\sum_{s=u+1}^{t} \mathbf{1}\{\hat{X}_{k,s} - \mu_k \ge c_{t,s}\} + \sum_{s=1}^{t} \mathbf{1}\{\hat{X}_{k^*,s^*} - \mu_k \le -c_{t,s^*}\} \Big] \end{aligned}$$

Now, taking the expectation of both sides,

$$\begin{aligned} \mathbb{E}[n_k(n)] &\le u + \sum_{t=u+1}^{n} \Big[\sum_{s=u+1}^{t} \mathbb{P}\big(\hat{X}_{k,s} - \mu_k \ge c_{t,s}\big) + \sum_{s=1}^{t} \mathbb{P}\big(\hat{X}_{k^*,s^*} - \mu_k \le -c_{t,s^*}\big) \Big] \\ &\le u + \sum_{t=u+1}^{n} \Big[\sum_{s=u+1}^{t} t^{-3} + \sum_{s=1}^{t} t^{-3} \Big] \le \frac{6\log(n)}{\Delta_k^2} + 1 + \frac{\pi^2}{3} \end{aligned}$$

Variants of UCB

- **UCB-V [Audibert et al., 2007] uses empirical variance:**

$$B_{k,t} \stackrel{\text{def}}{=} \widehat{\mu}_{k,t} + \sqrt{2\frac{\widehat{\sigma_{k,t}}^2 \log(1.2t)}{T_k(t)}} + \frac{3\log(1.2t)}{T_k(t)}.$$

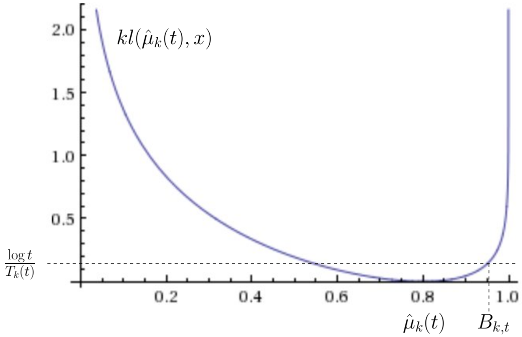
Then the expected regret is bounded as:

$$\mathbb{E}R_n \le 10\Big(\sum_{k:\Delta_k>0} \frac{\sigma_k^2}{\Delta_k} + 2\Big)\log(n).$$

KL-UCB

[Garivier & Cappé, 2011] and [Maillard et al., 2011].
For Bernoulli distributions, define the kl-UCB

$$B_{k,t} \overset{\text{def}}{=} \sup \left\{ x \in [0,1],\ \mathrm{kl}(\hat{\mu}_k(t), x) \le \frac{\log t}{T_k(t)} \right\}$$



(non-asymptotic version of Sanov's theorem)

KL-UCB

The regret of KL-UCB is then bounded as

$$\mathbb{E}R_n = \sum_{k:\Delta_k>0} \frac{\Delta_k}{\mathrm{kl}(\nu_k, \nu^*)} \log n + o(\log n).$$

This extends to several classes of distributions (one-dimensional exponential family, finitely supported, ...)
See also DMED [Honda, Takemura, 2010, 2011] and other related algorithms.

Idea: Use the full empirical distribution to get a refined UCB.

Lower bounds

For single-dimensional distributions [Lai, Robbins, 1985]:

$$\lim \inf_{n\to\infty} \frac{\mathbb{E}R_n}{\log n} \geq \sum_{k:\Delta_k>0} \frac{\Delta_k}{KL(\nu_k, \nu^*)}$$

For larger class of distributions $\mathcal{D}$ [Burnetas, Katehakis, 1996]:

$$\lim \inf_{n\to\infty} \frac{\mathbb{E}R_n}{\log n} \geq \sum_{k:\Delta_k>0} \frac{\Delta_k}{\mathcal{K}_{\inf}(\nu_k, \mu^*)},$$

where

$$\mathcal{K}_{\inf}(\nu, \mu) \overset{\text{def}}{=} \inf \left\{ KL(\nu, \nu') : \nu' \in \mathcal{D} \text{ and } \mathbb{E}_{X\sim\nu'}[X] > \mu \right\}.$$

The adversarial bandit

The rewards are no more i.i.d., but arbitrary!
At time t, simultaneously

- The adversary assigns a reward $x_{k,t} \in [0,1]$ to each arm $k \in \{1, \dots, K\}$
- The player chooses an arm k_t

The player receives the corresponding reward x_{k_t}. His goal is to maximize the sum of rewards.

Can we expect to do almost as good as the best (constant) arm?

Time	1	2	3	4	5	6	7	8	...
Arm pulled	1	2	1	1	2	1	1	1	
Reward arm 1	**1**	0.7	**0.9**	**1**	1	**1**	**0.8**	**1**	
Reward arm 2	0.9	**0**	1	0	**0.4**	0	0.6	0	

Reward obtained: 6.1. Arm 1: 7.4, Arm 2: 2.9.
Regret w.r.t. best constant strategy: $7.4 - 6.1 = 1.3$.

Notion of regret

Define the regret:

$$R_n = \max_{k\in\{1,\dots,K\}} \sum_{t=1}^{n} x_{k,t} - \sum_{t=1}^{n} x_{k_t}.$$

- Performance assessed in terms of the best constant strategy.
- Can we expect
$$\sup_{rewards} \mathbb{E} R_n/n \to 0?$$
- If the policy of the player is deterministic, there exists a reward sequence such that the performance is arbitrarily poor $\longrightarrow$ Need internal randomization.

EXP3 algorithm

EXP3 algorithm (Explore-Exploit using Exponential weights) [Auer et al, 2002]:

- $\eta > 0$ and $\beta > 0$ are two parameters of the algorithm.
- Initialize $w_1(k) = 1$ for all $k = 1, \dots, K$.
- At each round $t = 1, \dots, n$, player selects arm $k_t \sim p_t(\cdot)$, where

$$p_t(k) = (1-\beta)\frac{w_t(k)}{\sum_{i=1}^K w_t(i)} + \frac{\beta}{K},$$

with

$$w_t(k) = e^{\eta \sum_{s=1}^{t-1} \tilde{x}_s(k)},$$

where

$$\tilde{x}_s(k) = \frac{x_s(k)}{p_s(k)} \mathbf{1}\{k_s = k\}.$$

Performance of EXP3

Proposition 2.

Let $\eta \leq 1$ *and* $\beta = \eta K$. *We have* $\mathbb{E}R_n \leq \frac{\log K}{\eta} + (e-1)\eta n K$. *Thus, by choosing* $\eta = \sqrt{\frac{\log K}{(e-1)nK}}$, *it comes*

$$\sup_{\textit{rewards}} \mathbb{E}R_n \leq 2.63\sqrt{nK \log K}.$$

Properties:

- If all rewards are provided to the learner, with a similar algorithms we have [Lugosi and Cesa-Bianchi, 2006]

$$\sup_{\textit{rewards}} \mathbb{E}R_n = O(\sqrt{n \log K}).$$

Proof of Proposition 2 [part 1]

Write $W_t = \sum_{k=1}^K w_k(t)$. Notice that

$$\mathbb{E}_{k_s \sim p_s}[\tilde{x}_s(k)] = \sum_{i=1}^K p_s(i) \frac{x_s(k)}{p_s(k)} \mathbf{1}\{i = k\} = x_s(k),$$

$$\text{and } \mathbb{E}_{k_s \sim p_s}[\tilde{x}_s(k_s)] = \sum_{i=1}^K p_s(i) \frac{x_s(i)}{p_s(i)} \le K.$$

We thus have

$$\begin{aligned} \frac{W_{t+1}}{W_t} &= \sum_{k=1}^K \frac{w_k(t) e^{\eta \tilde{x}_t(k)}}{W_t} = \sum_{k=1}^K \frac{p_k(t) - \beta/K}{1-\beta} e^{\eta \tilde{x}_t(k)} \\ &\le \sum_{k=1}^K \frac{p_k(t) - \beta/K}{1-\beta} (1 + \eta \tilde{x}_t(k) + (e-2)\eta^2 \tilde{x}_t(k)^2), \end{aligned}$$

since $\eta \tilde{x}_t(k) \le \eta K/\beta = 1$, and $e^x \le 1 + x + (e-2)x^2$ for $x \le 1$.

Proof of Proposition 2 [part 2]

Thus

$$\begin{aligned}
\frac{W_{t+1}}{W_t} &\leq 1 + \frac{1}{1-\beta}\sum_{k=1}^{K} p_k(t)(\eta\tilde{x}_t(k) + (e-2)\eta^2\tilde{x}_t(k)^2), \\
\log\frac{W_{t+1}}{W_t} &\leq \frac{1}{1-\beta}\sum_{k=1}^{K} p_k(t)(\eta\tilde{x}_t(k) + (e-2)\eta^2\tilde{x}_t(k)^2), \\
\log\frac{W_{n+1}}{W_1} &\leq \frac{1}{1-\beta}\sum_{t=1}^{n}\sum_{k=1}^{K} p_k(t)(\eta\tilde{x}_t(k) + (e-2)\eta^2\tilde{x}_t(k)^2).
\end{aligned}$$

But we also have

$$\log\frac{W_{n+1}}{W_1} = \log\sum_{k=1}^{K} e^{\eta\sum_{t=1}^{n}\tilde{x}_t(k)} - \log K \geq \eta\sum_{t=1}^{n}\tilde{x}_t(k) - \log K,$$

for any $k = 1, \ldots, n$.

Proof of Proposition 2 [part 3]

Take expectation w.r.t. internal randomization of the algo, thus for all k,

$$
\begin{aligned}
\mathbb{E}\Big[(1-\beta)\sum_{t=1}^{n}\tilde{x}_t(k) - \sum_{t=1}^{n}\sum_{i=1}^{K} p_i(t)\tilde{x}_t(i)\Big] &\leq (1-\beta)\frac{\log K}{\eta} \\
&+ (e-2)\eta\mathbb{E}\Big[\sum_{t=1}^{n}\sum_{k=1}^{K} p_k(t)\tilde{x}_t(k)^2\Big] \\
\mathbb{E}\Big[\sum_{t=1}^{n}x_t(k) - \sum_{t=1}^{n}x_t(k_t)\Big] &\leq \beta n + \frac{\log K}{\eta} + (e-2)\eta nK \\
\mathbb{E}[R_n(k)] &\leq \frac{\log K}{\eta} + (e-1)\eta nK
\end{aligned}
$$

Population of bandits

- Bandit (or regret minimization) algorithms = tool for rapidly selecting the best action.
- Basic building block for solving more complex problems
- We now consider a population of bandits:

Adversarial bandits

Collaborative bandits

Game between bandits

Consider a 2-players zero-sum repeated game:
A and B play actions: 1 or 2 simultaneously, and receive the reward (for A):

A \ B	1	2
1	2	0
2	-1	1

(A likes consensus, B likes conflicts)

Now, let A and B be bandit algorithms, aiming at minimizing their regret, i.e. for player A:

$$R_n(A) \overset{\text{def}}{=} \max_{a \in \{1,2\}} \sum_{t=1}^{n} r_A(a, B_t) - \sum_{t=1}^{n} r_A(A_t, B_t).$$

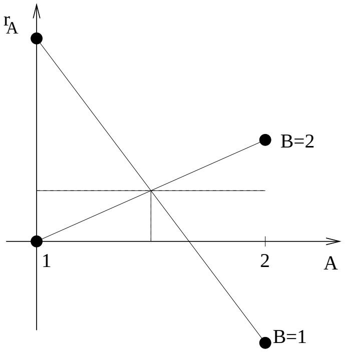
What happens?

Nash equilibrium

Nash equilibrium: (mixed) strategy for both players, such that no player has incentive for changing unilaterally his own strategy.

A \ B	1	2
1	2	0
2	-1	1

Here: A plays 1 with probability $p_A = 1/2$, B plays 1 with probability $p_B = 1/4$.



Regret minimization → Nash equilibrium

Define the regret of A:

$$R_n(A) \overset{\mathrm{def}}{=} \max_{a\in\{1,2\}} \sum_{t=1}^{n} r_A(a, B_t) - \sum_{t=1}^{n} r_A(A_t, B_t).$$

and that of B accordingly.

Proposition 3.

If both players perform a (Hannan) consistent regret-minimization strategy (i.e. $R_n(A)/n \to 0$ and $R_n(B)/n \to 0$), then the empirical frequencies of chosen actions of both players converge to a Nash equilibrium.

(Remember that EXP3 is consistent!)

Note that in general, we have convergence towards correlated equilibrium [Foster and Vohra, 1997].

Sketch of proof:

Write $p_A^n \stackrel{\text{def}}{=} \frac{1}{n}\sum_{t=1}^n \mathbf{1}_{A_t=1}$ and $p_B^n \stackrel{\text{def}}{=} \frac{1}{n}\sum_{t=1}^n \mathbf{1}_{B_t=1}$ and $r_A(p,q) \stackrel{\text{def}}{=} \mathbb{E} r_A(A\sim p, B\sim q)$.
Regret-minimization algorithm: $R_n(A)/n \to 0$ means that: $\forall \varepsilon > 0$, for n large enough,

$$\begin{aligned}
\max_{a\in\{1,2\}} \frac{1}{n}\sum_{t=1}^n r_A(a,B_t) - \frac{1}{n}\sum_{t=1}^n r_A(A_t,B_t) &\leq \varepsilon \\
\max_{a\in\{1,2\}} r_A(a,p_B^n) - r_A(p_A^n,p_B^n) &\leq \varepsilon \\
r_A(p,p_B^n) - r_A(p_A^n,p_B^n) &\leq \varepsilon,
\end{aligned}$$

for all $p \in [0,1]$. Now, using $R_n(B)/n \to 0$ we deduce that:

$$r_A(p,p_B^n) - \varepsilon \leq r_A(p_A^n,p_B^n) \leq r_A(p_A^n,q) + \varepsilon, \quad \forall p,q \in [0,1]$$

Thus the empirical frequencies of actions played by both players is arbitrarily close to a Nash strategy.

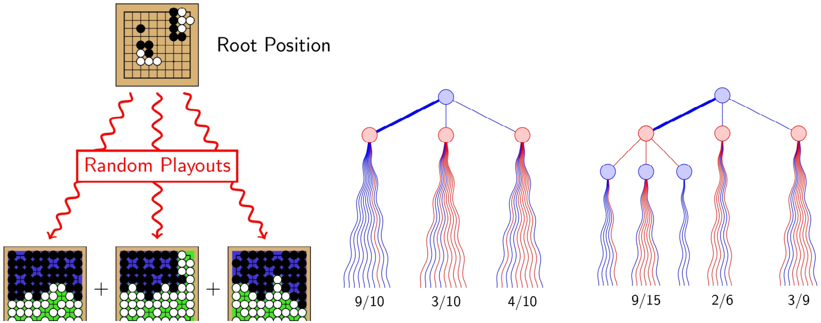
Texas Hold'em Poker

- In the 2-players Poker game, the Nash equilibrium is interesting (zero-sum game)
- A policy:
  information set (my cards + board + pot) $\rightarrow$ probabilities over decisions (check, raise, fold)
- Space of policies is huge!

Idea: Approximate the Nash equilibrium by using bandit algorithms assigned to each information set.

- This provides the world best Texas Hold'em Poker program for 2-player with pot-limit [Zinkevich et al., 2007]

Monte Carlo Tree Search



MCTS in Crazy-Stone (Rémi Coulom, 2005)

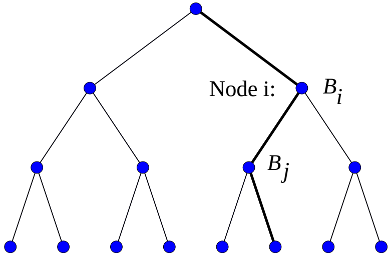
Idea: use bandits at each node.

UCB applied to Trees

Upper Confidence Bound (UCB) algo at each node

$$B_j \overset{\text{def}}{=} X_{j,n_j} + \sqrt{\frac{2\log(n_i)}{n_j}}.$$

Intuition:
- Explore first the most promising branches
- Average converges to max



- Adaptive Multistage Sampling (AMS) algorithm [Chang, Fu, Hu, Marcus, 2005]
- UCB applied to Trees (UCT) [Kocsis and Szepesvári, 2006]

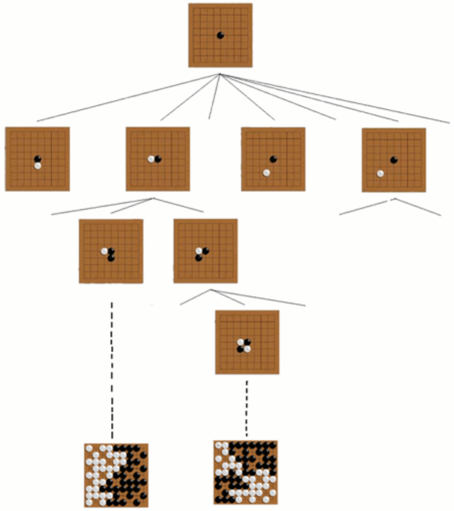
The MoGo program

[Gelly et al., 2006] + collaborative work with many others.

Features:

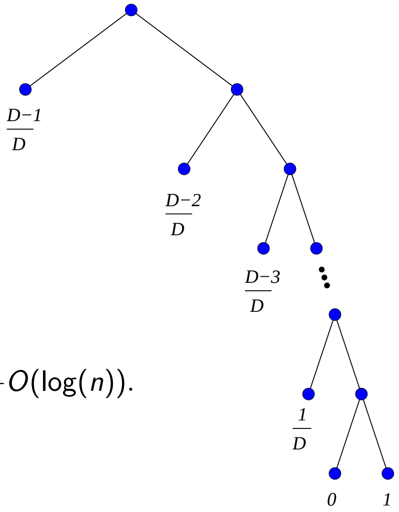
- Explore-Exploit with UCT
- Monte-Carlo evaluation
- Asymmetric tree expansion
- Anytime algo
- Use of features

Among world best programs!



No finite-time guarantee for UCT

Problem: at each node, the rewards are not i.i.d.
Consider the tree:



The left branches seem better than right branches, thus are explored for a **very** long time before the optimal leaf is eventually reached.
The expected regret is disastrous:

$$\mathbb{E}R_n = \Omega(\underbrace{\exp(\exp(\ldots\exp(}_{D \text{ times}}1)\ldots))) + O(\log(n)).$$

See [Coquelin and Munos, 2007]

Optimism for decision making

Outline:

- Optimization of deterministic Lipschitz functions
- Extension to locally smooth functions,
    - when the metric is known,
    - and when it's not
- Extension to the stochastic case
- Application to planning in MDPs

Optimization of a deterministic Lipschitz function

Problem: Find online the maximum of $f : X \to \boldsymbol{R}$, assumed to be Lipschitz:

$$|f(x) - f(y)| \le \ell(x, y).$$

Protocol:

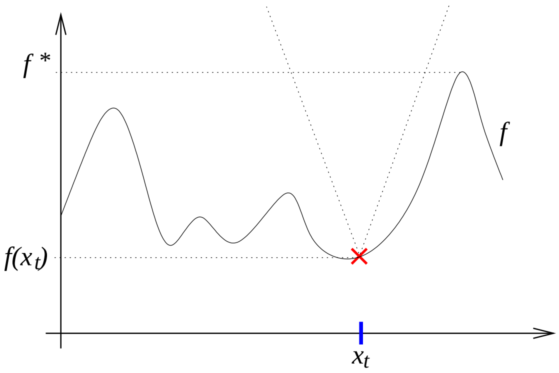
- For each time step $t = 1, 2, \dots, n$ select a state $x_t \in X$
- Observe $f(x_t)$
- Return a state $x(n)$

Performance assessed in terms of the loss

$$r_n = f^* - f(x(n)),$$

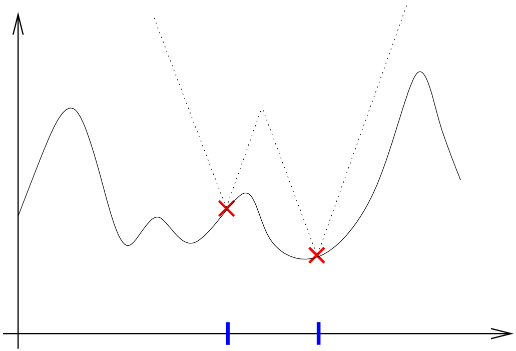
where $f^* = \sup_{x \in X} f(x)$.

Example in 1d



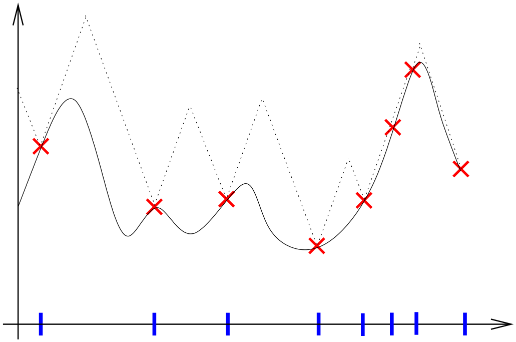
Lipschitz property $\rightarrow$ the evaluation of f at x_t provides a first upper-bound on f.

Example in 1d (continued)



New point $\rightarrow$ refined upper-bound on f.

Example in 1d (continued)



Question: where should one sample the next point?

Answer: select the point with highest upper bound!

“Optimism in the face of (partial observation) uncertainty”

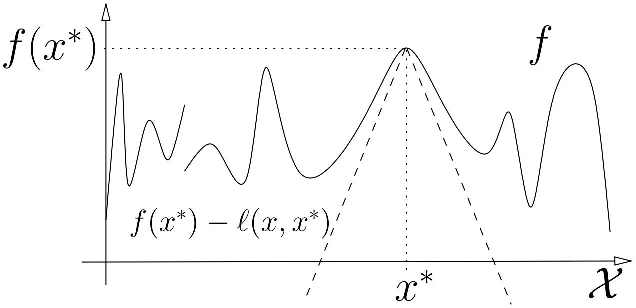
Several issues

1. Lipschitz assumption is too strong
2. Finding the optimum of the upper-bounding function may be hard!
3. What if we don't know the metric ℓ?
4. How to handle noise?

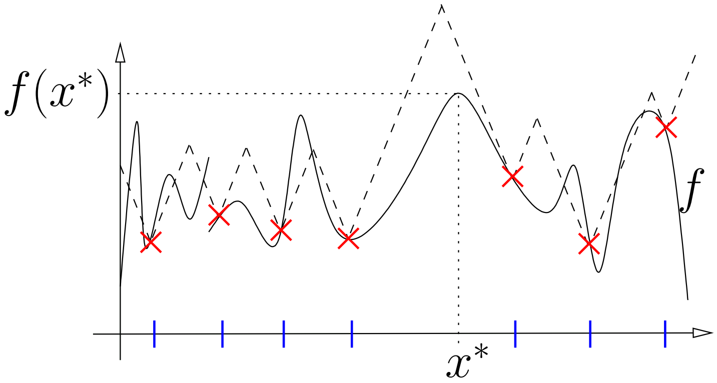
Local smoothness property

Assumption: f **is "locally smooth" around its max. w.r.t.** ℓ where ℓ is a semi-metric (symmetric, and $\ell(x,y)=0 \Leftrightarrow x=y$):
For all $x \in \mathcal{X}$,

$$f(x^*) - f(x) \leq \ell(x, x^*).$$



Local smoothness is enough!



Optimistic principle only requires:

- a true bound at the maximum
- the bounds gets refined when adding more points

Efficient implementation

Deterministic Optimistic Optimization (DOO) builds a hierarchical partitioning of the space where cells are refined according to their upper bounds.

- For $t = 1$ to n,
    - Define an upper bound for each cell:
    $$B_i = f(x_i) + \mathrm{diam}_\ell(X_i)$$
    - Select the cell with highest bound
    $$I_t = \operatorname*{argmax}_i B_i.$$
    - Expand I_t: refine the grid and evaluate f in children cells
- Return $x(n) \overset{\mathrm{def}}{=} \mathrm{argmax}_{\{x_t\}_{1\leq t\leq n}} f(x_t)$

Near-optimality dimension

Define the **near-optimality dimension** of f as the smallest $d \geq 0$ such that $\exists C$, $\forall \epsilon$, the set of ε-optimal states

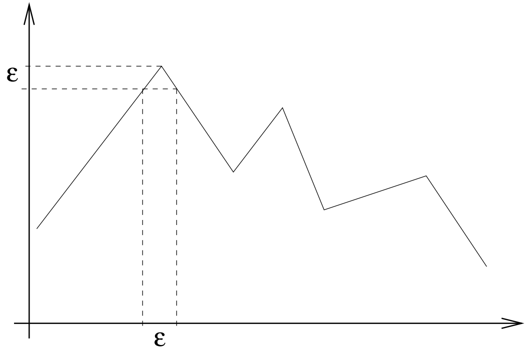
$$X_\varepsilon \overset{\text{def}}{=} \{x \in X, f(x) \geq f^* - \varepsilon\}$$

can be covered by $C\varepsilon^{-d}$ ℓ-balls of radius ε.

Example 1:

Assume the function is piecewise linear at its maximum:

$$f(x^*) - f(x) = \Theta(||x^* - x||).$$

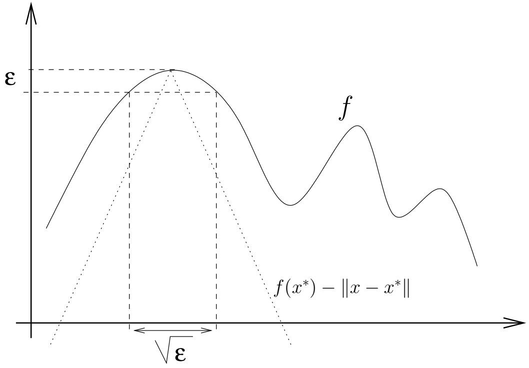


Using $\ell(x, y) = \|x - y\|$, it takes $O(\epsilon^0)$ balls of radius ϵ to cover X_ε. Thus $d = 0$.

Example 2:

Assume the function is locally quadratic around its maximum:

$$f(x^*) - f(x) = \Theta(||x^* - x||^2).$$

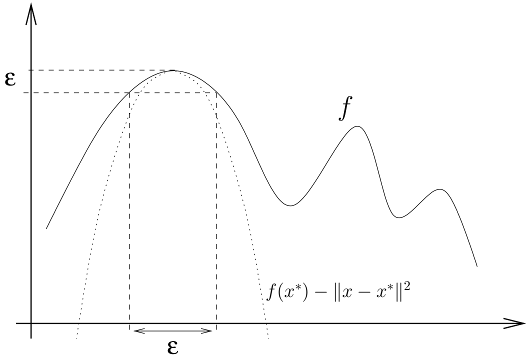


For $\ell(x, y) = ||x - y||$, it takes $O(\epsilon^{-D/2})$ balls of radius ϵ to cover X_ε (of size $O(\epsilon^{D/2})$). Thus $d = D/2$.

Example 2:

Assume the function is locally quadratic around its maximum:

$$f(x^*) - f(x) = \Theta(||x^* - x||^2)$$

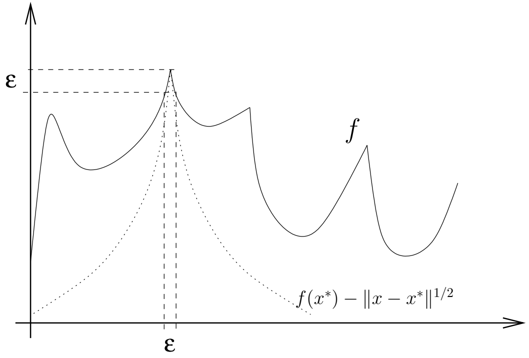


For $\ell(x, y) = ||x - y||^2$, it takes $O(\epsilon^0)$ ℓ-balls of radius ϵ to cover X_ε. Thus $d = 0$.

Example 3:

Assume the function has a square-root behavior around its maximum:

$$f(x^*) - f(x) = \Theta(||x^* - x||^{1/2})$$



For $\ell(x, y) = \|x - y\|^{1/2}$ we have $d = 0$.

Example 4:

Assume $\mathcal{X} = [0,1]^D$ and f is locally equivalent to a polynomial of degree $\alpha > 0$ around its maximum (i.e. f is α-smooth):

$$f(x^*) - f(x) = \Theta(||x^* - x||^\alpha)$$

Consider the semi-metric $\ell(x, y) = \|x - y\|^\beta$, for some $\beta > 0$.

- If $\alpha = \beta$, then $d = 0$.
- If $\alpha > \beta$, then $d = D(\frac{1}{\beta} - \frac{1}{\alpha}) > 0$.
- If $\alpha < \beta$, then the function is not locally smooth wrt ℓ.

Analysis of DOO (deterministic case)

Assume that the ℓ-diameters of the nodes of depth h decrease exponentially fast with h (i.e., $\text{diam}(h) = c\gamma^h$, for some $c > 0$ and $\gamma < 1$).
This is true for example when $\mathcal{X} = [0,1]^D$ and $\ell(x,y) = \|x-y\|^\beta$ for some $\beta > 0$.

Theorem 1.
The loss of DOO is

$$r_n = \begin{cases} \left(\frac{C}{1-\gamma^d}\right)^{1/d} n^{-1/d} & \textit{for } d > 0, \\ c\gamma^{n/C-1} & \textit{for } d = 0. \end{cases}$$

(Remember that $r_n \overset{\text{def}}{=} f(x^*) - f(x(n))$.)

About the local smoothness assumption

Assume f satisfies $f(x^*) - f(x) = \Theta(||x^* - x||^\alpha)$.

Use DOO with the semi-metric $\ell(x, y) = \|x - y\|^\beta$:

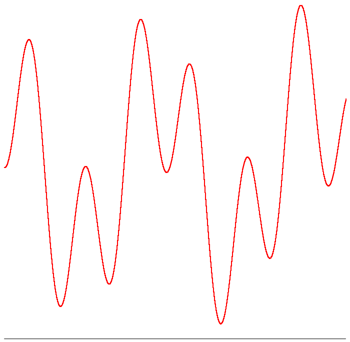
- If $\alpha = \beta$, then $d = 0$: the true "local smoothness" of the function is known, and exponential rate is achieved.
- If $\alpha > \beta$, then $d = D(\frac{1}{\beta} - \frac{1}{\alpha}) > 0$: we under-estimate the smoothness, which causes more exploration than needed.
- If $\alpha < \beta$: We over-estimate the true smoothness and DOO may fail to find the global optimum.

The performance of DOO heavilly relies on our knowledge of the true local smoothness.

Experiments [1]

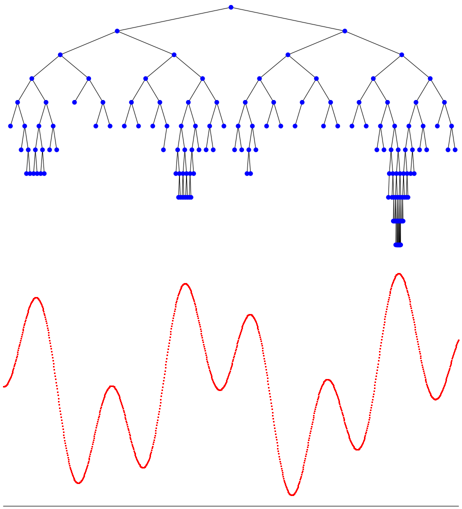
$f(x) = \frac{1}{2}(\sin(13x)\sin(27x) + 1)$ satisfies the local smoothness assumption $f(x) \geq f(x^*) - \ell(x, x^*)$ with

- $\ell_1(x, y) = 14|x - y|$ (i.e., f is globally Lipschitz),
- $\ell_2(x, y) = 222|x - y|^2$ (i.e., f is locally quadratic).



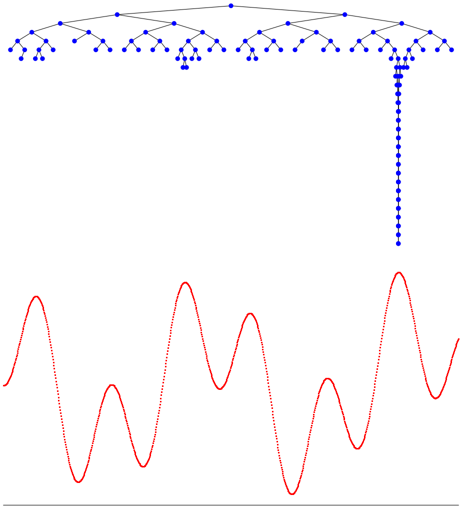
Experiments [2]

Using $\ell_1(x, y) = 14|x - y|$ (i.e., f is globally Lipschitz). $n = 150$.



Experiments [3]

Using $\ell_2(x, y) = 222|x - y|^2$ (i.e., f is locally quadratic). $n = 150$.



Experiments [4]

n	uniform grid	DOO with ℓ_1 $(d = 1/2)$	DOO with ℓ_2 $(d = 0)$
50	1.25×10^{-2}	2.53×10^{-5}	1.20×10^{-2}
100	8.31×10^{-3}	2.53×10^{-5}	1.67×10^{-7}
150	9.72×10^{-3}	4.93×10^{-6}	4.44×10^{-16}

Loss r_n for different values of n for a uniform grid and DOO with the two semi-metric ℓ_1 and ℓ_2.

What if the smoothness is unknown?

Previous algorithms heavily rely on the knowledge or the local smoothness of the function (i.e. knowledge of the best metric).

Question: When the smoothness is unknown, is it possible to implement the optimistic principle for function optimization?

DIRECT algorithm [Jones et al., 1993]

Assumes f is Lipschitz but the Lipschitz constant L is unknown.

The DIRECT algorithm expands simultaneously all nodes that may potentially contain the maximum for some value of L.

Illustration of DIRECT

The sin function and its upper bound for $L = 2$.

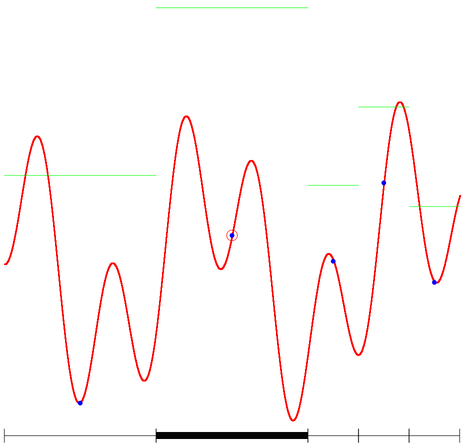
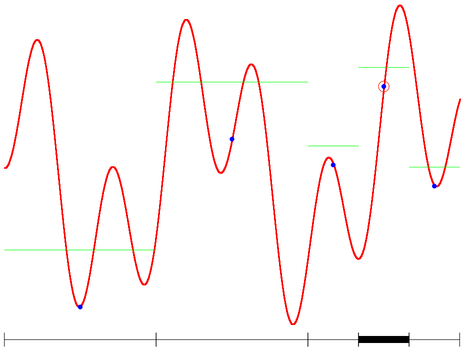


Illustration of DIRECT

The sin function and its upper bound for $L = 1/2$.



Simultaneous Optimistic Optimization (SOO)

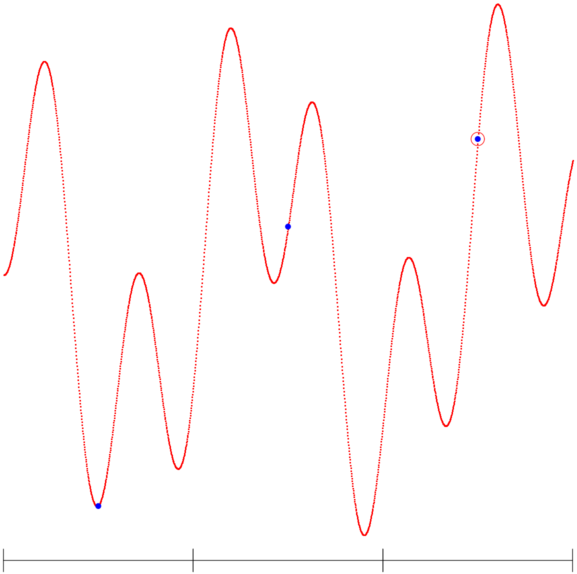
Extends DIRECT to any semi-metric ℓ and for any function locally smooth w.r.t. ℓ.

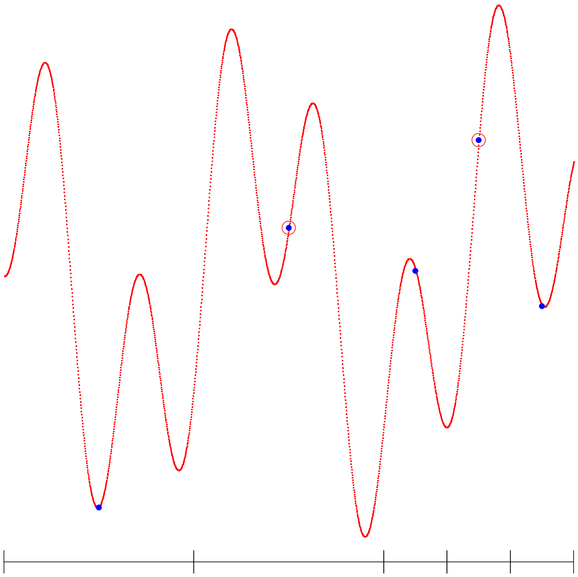
[Munos, 2011]

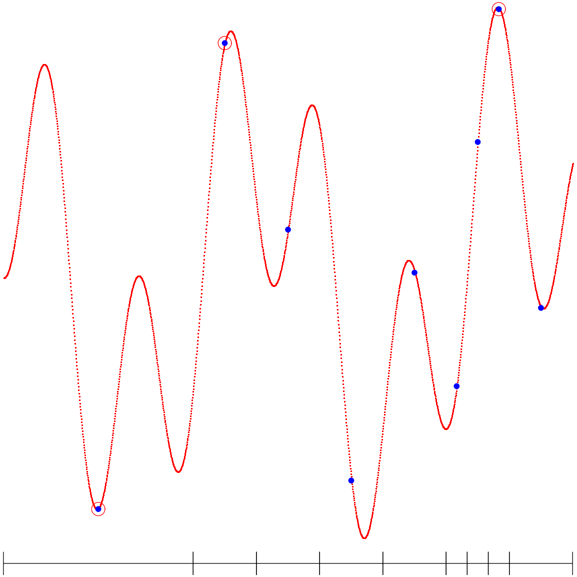
- Expand several leaves simultaneously
- SOO expands at most one leaf per depth
- SOO expands a leaf only if its value is larger that the value of all leaves of same or lower depths.
- At round t, SOO does not expand leaves with depth larger than $h_{\max}(t)$

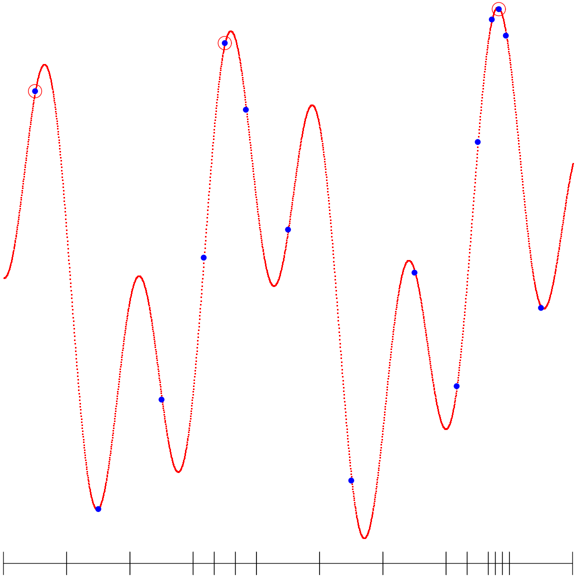
SOO algorithm

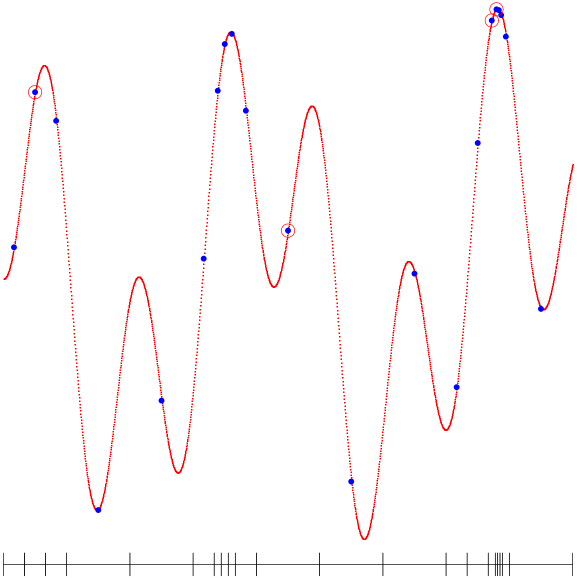
```
**Input:** the maximum depth function $t \mapsto h_{\max}(t)$
**Initialization:** $\mathcal{T}_1 = \{(0,0)\}$ (root node). Set $t = 1$.
**while** True **do**
    Set $v_{\max} = -\infty$.
    **for** $h = 0$ to $\min(\text{depth}(\mathcal{T}_t), h_{\max}(t))$ **do**
        Select the leaf $(h,j) \in \mathcal{L}_t$ of depth $h$ with max $f(x_{h,j})$ value
        **if** $f(x_{h,i}) > v_{\max}$ **then**
            Expand the node $(h,i)$, Set $v_{\max} = f(x_{h,i})$, Set $t = t + 1$
            **if** $t = n$ **then** return $x(n) = \arg\max_{(h,i)\in\mathcal{T}_n} x_{h,i}$
        **end if**
    **end for**
**end while**.
```

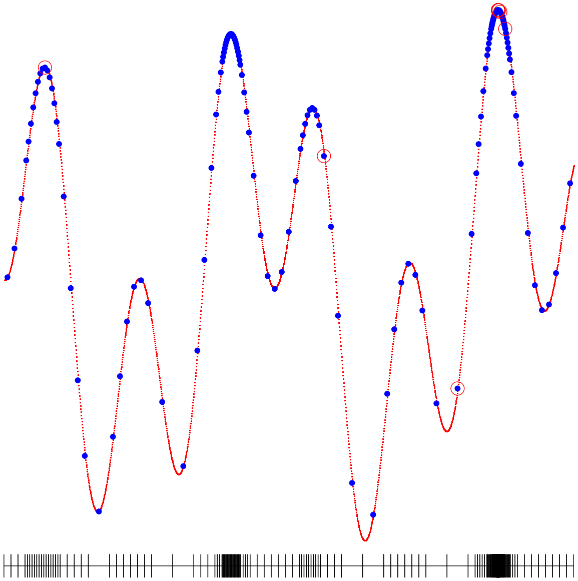


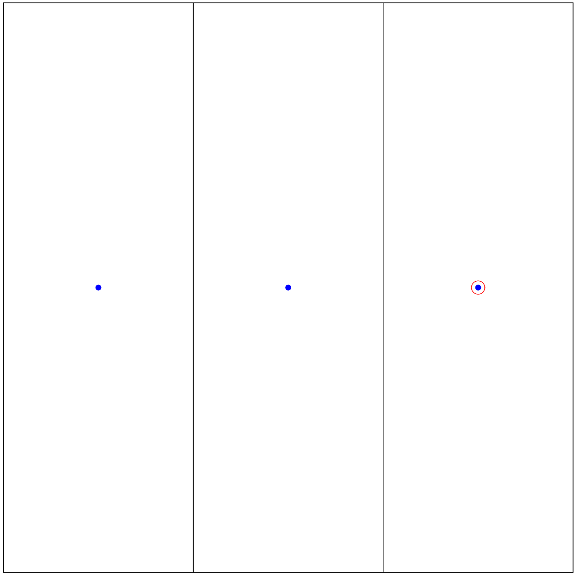


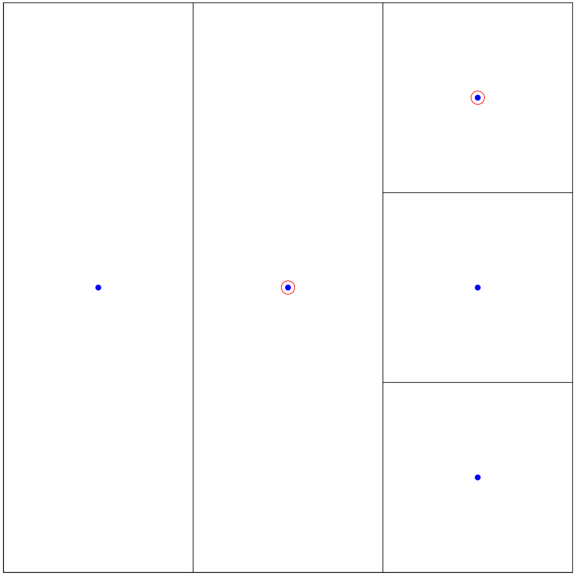


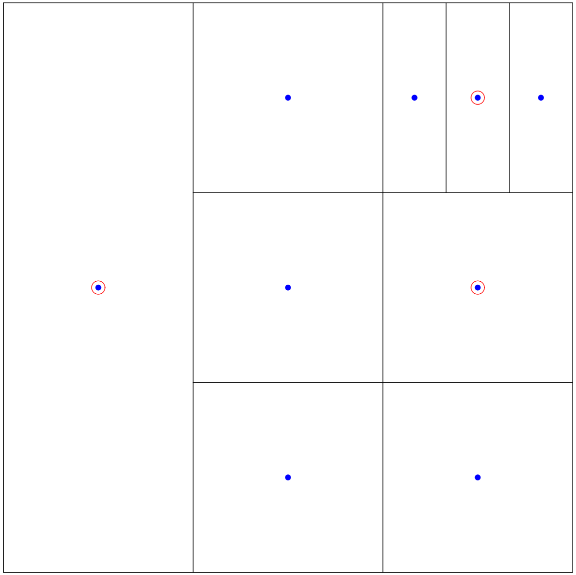


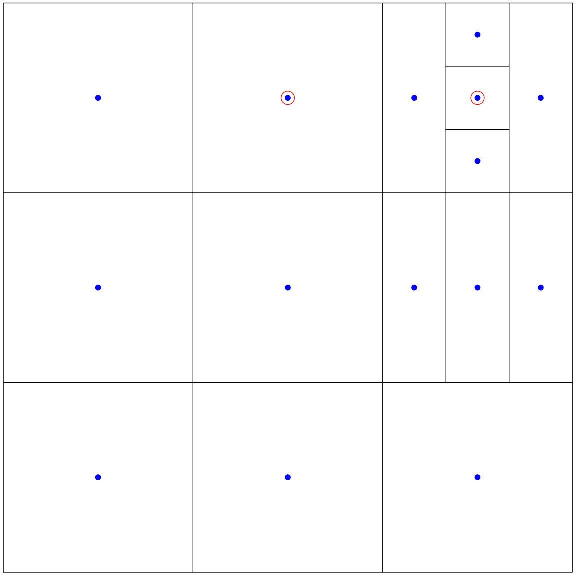


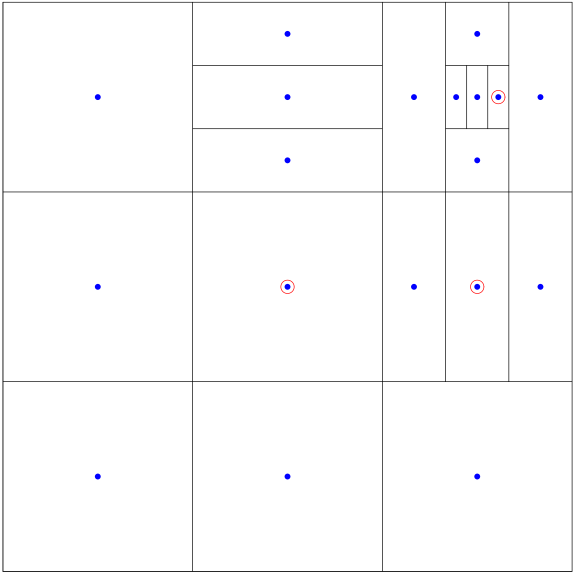


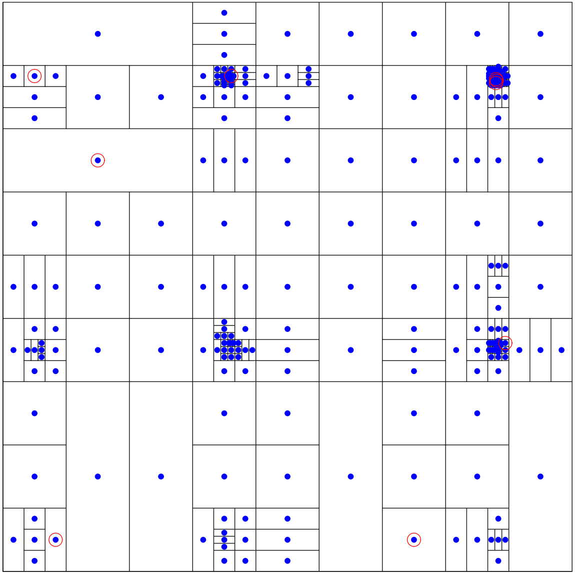












Performance of SOO

Theorem 2.
For any semi-metric ℓ such that

- *f is locally smooth w.r.t. ℓ*
- *The ℓ-diameter of cells of depth h is $c\gamma^h$*
- *The near-optimality dimension of f w.r.t. ℓ is $d = 0$,*

by choosing $h_{\max}(n) = \sqrt{n}$, the expected loss of SOO is

$$r_n \leq c\gamma^{\sqrt{n}/C-1}$$

In the case $d > 0$ a similar statement holds with $\mathbb{E}r_n = \widetilde{O}(n^{-1/d})$.

Performance of SOO

Remarks:

- Since the algorithm does not depend on ℓ, the analysis holds for the best possible choice of the semi-metric ℓ satisfying the assumptions.
- **SOO does almost as well as DOO optimally fitted** (thus "adapts" to the unknown local smoothness of f).

Numerical experiments

Again for the function $f(x) = (\sin(13x)\sin(27x) + 1)/2$ we have:

n	SOO	uniform grid	DOO with ℓ_1	DOO with ℓ_2
50	3.56×10^{-4}	1.25×10^{-2}	2.53×10^{-5}	1.20×10^{-2}
100	5.90×10^{-7}	8.31×10^{-3}	2.53×10^{-5}	1.67×10^{-7}
150	1.92×10^{-10}	9.72×10^{-3}	4.93×10^{-6}	4.44×10^{-16}

The case $d = 0$ is non-trivial!

Example:

- f is locally α-smooth around its maximum:

$$f(x^*) - f(x) = \Theta(\|x^* - x\|^\alpha),$$

  for some $\alpha > 0$.
- SOO algorithm does not require the knowledge of ℓ,
- Using $\ell(x, y) = \|x - y\|^\alpha$ in the analysis, all assumptions are satisfied (with $\gamma = 3^{-\alpha/D}$ and $d = 0$, thus the loss of SOO is $r_n = O(3^{-\sqrt{n}\alpha/(CD)})$ (stretched-exponential loss),
- This is almost as good as DOO optimally fitted!

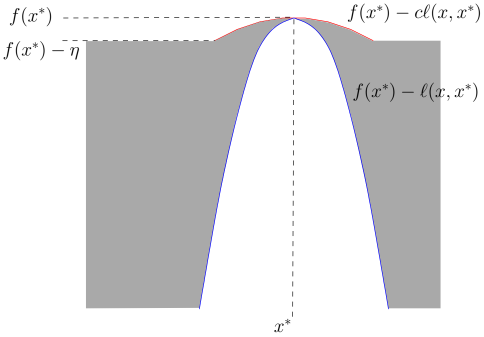
(Extends to the case $f(x^*) - f(x) \approx \sum_{i=1}^{D} c_i |x_i^* - x_i|^{\alpha_i}$)

The case $d = 0$

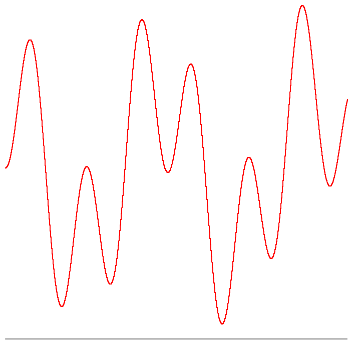
More generally, any function whose **upper- and lower envelopes around x^* have the same shape**: $\exists c > 0$ and $\eta > 0$, such that

$$\min(\eta, c\ell(x, x^*)) \leq f(x^*) - f(x) \leq \ell(x, x^*), \quad \text{for all } x \in \mathcal{X}.$$

has a near-optimality $d = 0$ (w.r.t. the metric ℓ).

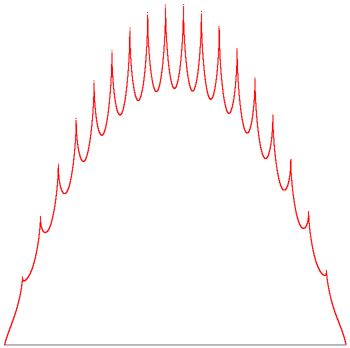


Example of functions for which $d = 0$



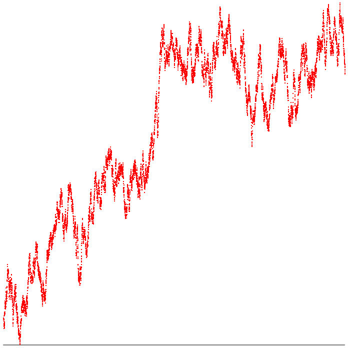
$$\ell(x, y) = c\|x - y\|^2$$

Example of functions with $d = 0$



$\ell(x, y) = c\|x - y\|^{1/2}$

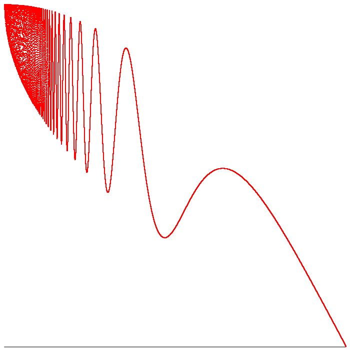
$d = 0$?



$$\ell(x, y) = c\|x - y\|^{1/2}$$

$d > 0$

$$f(x) = 1 - \sqrt{x} + (-x^2 + \sqrt{x}) * (\sin(1/x^2) + 1)/2$$



The lower-envelope is of order $1/2$ whereas the upper one is of order 2. We deduce that $d = 3/2$ and $r_n = O(n^{-2/3})$.

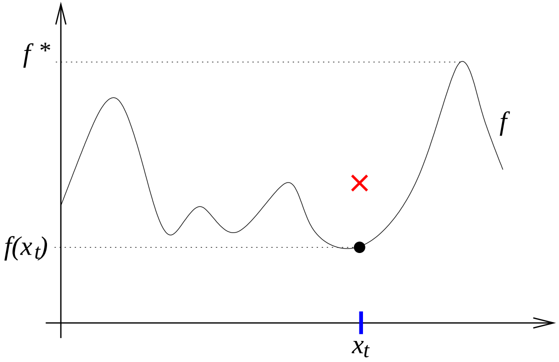
Comparison SOO versus DIRECT algorithms

- **SOO is much more general than DIRECT**: the function is only **locally smooth** and the space is **semi-metric**.
- **Finite-time analysis of SOO** (whereas only a consistency property $\lim_{n\to\infty} r_n = 0$ is available for DIRECT in [Finkel and Kelley, 2004])
- **SOO is a rank-based algorithm**: any transformation of the values while preserving their rank will not change anything in the algorithm. Thus extends to the optimization of function givens pair-wise comparisons.
- SOO is easier to implement...

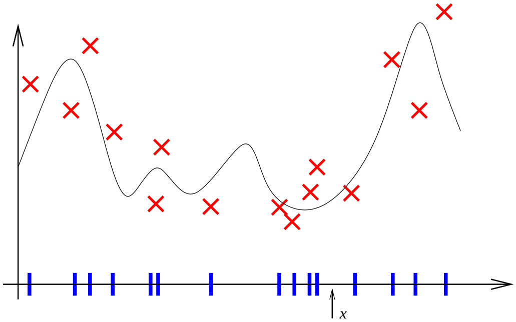
How to handle noise?

The evaluation of f at x_t is perturbed by noise:

$$r_t = f(x_t) + \epsilon_t, \text{ with } \mathbb{E}[\epsilon_t | x_t] = 0.$$

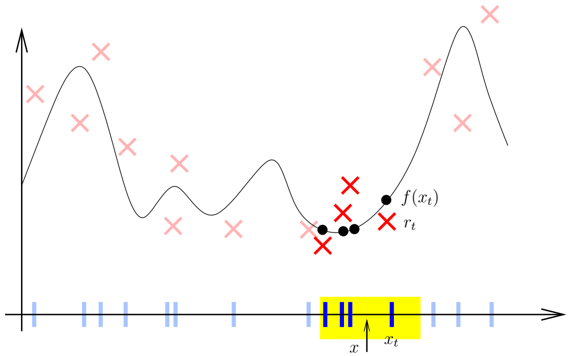


Where should one sample next?



How to define a high probability upper bound at any state x?

UCB in a given cell



For a fixed domain $X_i \ni x$ containing T_i points $\{x_t\} \in X_i$, we have that $\sum_{t=1}^{T_i} r_t - f(x_t)$ is a Martingale. Thus by Azuma's inequality, with $1/n$-confidence,

$$\frac{1}{T_i}\sum_{t=1}^{T_i} r_t + \sqrt{\frac{\log n}{2T_i}} \geq \frac{1}{T_i}\sum_{t=1}^{T_i} f(x_t).$$

Stochastic SOO (StoSOO)

A simple way to extends SOO to the case of stochastic rewards is the following:

- Select a cell i (and sample f at x_i) according to SOO based on the values

$$\widehat{\mu}_{i,t} + c\sqrt{\frac{\log n}{T_i(t)}},$$

- Expand the cell X_i only if $T_i(t) \geq k$, where k is a parameter.

Remark: This really looks like UCT, except that

- several cells are selected at each round,
- a cell is refined only when we received k samples.

Performance of StoSOO

Theorem 3 (Valko et al., 2013).

For any semi-metric ℓ such that

- *f is locally smooth w.r.t. ℓ*
- *The ℓ-diameters of the cells decrease exponentially fast with their depth,*
- *The near-optimality dimension of f w.r.t. ℓ is $d = 0$,*

by choosing $k = \frac{n}{(\log n)^3}$, $h_{\max}(n) = (\log n)^{3/2}$, the expected loss of StoSOO is

$$\mathbb{E} r_n = O\Big(\frac{(\log n)^2}{\sqrt{n}}\Big).$$

Online planning in a MDP

Setting:

- Assume we have a model of the MDP.
- The state space is large: no way to approximate the value function
- Search for the best policy given an initial state, using a finite numerical budget (number of calls to the model)

Protocol:

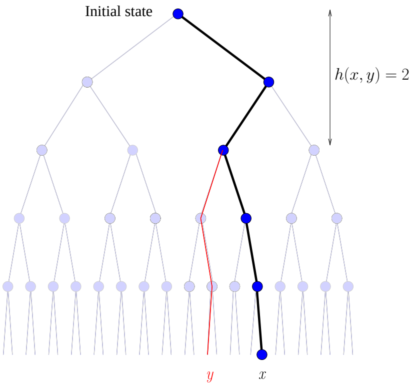
- From current state s_k, perform planning using n calls to the model and recommend action $a(n)$,
- Play $a(n)$, observe next state s_{k+1}, and repeat

$$\text{Loss:} \qquad r_n \overset{\text{def}}{=} \max_{a \in A} Q^*(s_k, a) - Q^*(s_k, a(n)).$$

Deterministic transitions and rewards

(infinite time horizon and discounted setting)

- A policy x is an infinite path
- Value $f(x) = \sum_{s\geq 0} \gamma^s r_s(x)$, where the rewards are in $[0,1]$
- Metric: $\ell(x,y) = \frac{\gamma^{h(x,y)}}{1-\gamma}$
- Prop: $f(x)$ is Lipschitz w.r.t. ℓ
- $\rightarrow$ Use optimistic search



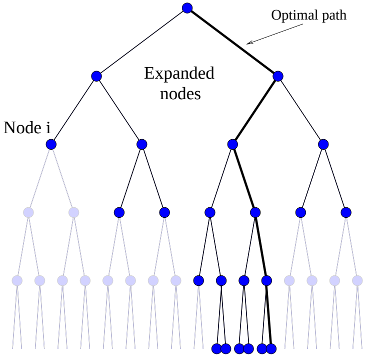
OPD algorithm

Optimistic Planning in Deterministic systems:

- Define the B-values:

$$B_i \overset{\text{def}}{=} \sum_{s=0}^{d(i)} \gamma^s r_s + \frac{\gamma^{d(i)+1}}{1-\gamma}$$

- We have $B_i \geq \max_{x \ni i} f(x)$
- For each round $t = 1$ to n, expand the node with highest B-value
- Observe reward, update B-values



Recommend the first action $a(n)$ of the best policy.

Performance of OPD

Define β such that $\mathbb{P}(\text{Random path is } \epsilon\text{-optimal}) = O(\epsilon^{\beta})$.

Define $\kappa \overset{\text{def}}{=} K\gamma^{\beta} \in [1, K]$. Then κ is the branching factor of the set of near-optimal sequences:

$$I = \Big\{ \text{sequences of length } h \text{ such that } \sum_{s=0}^{h} \gamma^s r_s \geq V^* - \frac{\gamma^{h+1}}{1-\gamma} \Big\}.$$

Property: κ relates to the near-opt. dimension $d = \frac{\log \kappa}{\log 1/\gamma}$ (the set of ϵ-optimal paths is covered by $O(\epsilon^{-d})$ ℓ-balls of radius ϵ)

Loss of OPD [Hren and Munos, 2008]:

$$r_n = O(n^{-\frac{1}{d}}) = O\left(n^{-\frac{\log 1/\gamma}{\log \kappa}} \right).$$

Performance depends on the quantity of near-optimal policies

κ-minimax lower bounds

Let $\mathcal{M}_\kappa$ the set of problems with coefficient κ.
Upper-bound of OPD uniformly over $\mathcal{M}_\kappa$

$$\sup_{M \in \mathcal{M}_\kappa} r_n(\mathcal{A}_{OPD}, M) = O\left(n^{-\frac{\log 1/\gamma}{\log \kappa}}\right).$$

We can prove that we have a κ-minimax lower-bound:

$$\sup_{\mathcal{A}} \inf_{M \in \mathcal{M}_\kappa} r_n(\mathcal{A}, M) \geq \Omega\left(n^{-\frac{\log 1/\gamma}{\log \kappa}}\right).$$

Sketch of proof:
OPD only expands nodes in I. Reciproquely, I is the set of nodes that need to be expanded in order to find the optimal path.

Extension to stochastic rewards

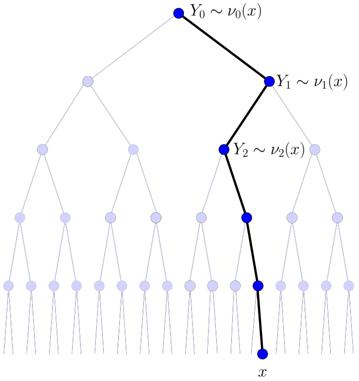
Dynamics are still deterministic thus the space of policies is still the set of sequences of actions.

Stochastic rewards:

- Reward along a path x:

$$\sum_{s\geq 0}\gamma^s Y_s,$$

  where $Y_s \sim \nu_s(x)$ where $\nu_s(x)$ is the reward distribution ($\in [0,1]$) after s actions along path x.
- Write $r_s(x) = \mathbb{E}_{X\sim\nu_s(x)}[X]$ and $f(x) = \sum_{s\geq 0}\gamma^s r_s(x)$



Then $f(x)$ is Lipschitz w.r.t. $\ell(x,y) = \frac{\gamma^{h(x,y)}}{1-\gamma}$ and one can think of applying HOO.

Using HOO for planning

Apply HOO to the search space $\mathcal{X}$:

- Assign a B-value to each finite sequence
- At each round t, select a finite sequence x_t maximizing the B-value.
- Observe sample reward $\sum_{s \geq 0} \gamma^s Y_s(x_t)$ of the path x_t and use it to update the B-values.
- The loss is

$$O(n^{-\frac{1}{d+2}}).$$

Problem: HOO does not make full use of the tree structure: It uses the sample reward of the whole path x but not the individual reward samples $Y_s(x)$ collected along the path x.

Optimistic sampling using the tree structure

Open Loop Optimistic Planning (OLOP) [Bubeck, Munos, 2010]:

- At round t, play path x_t (up to depth $h = \frac{1}{2}\frac{\log n}{\log 1/\gamma}$)
- Observe sample rewards $Y_s(x_t)$ of each node along the path x_t
- Compute empirical rewards $\hat{\mu}_t(x_{1:s})$ for each node $x_{1:s}$ of depth $s \leq h$
- Define bound for each path x:

$$B_t(x) = \min_{1\leq j\leq h} \Big[\sum_{s=0}^{j} \gamma^s \Big(\hat{\mu}_t(x_{1:s}) + \sqrt{\frac{2\log n}{T_{x_{1:s}}(t)}} \Big) + \frac{\gamma^{j+1}}{1-\gamma} \Big]$$

- Select path $x_{t+1} = \mathrm{argmax}_x B_t(x)$

This algorithm fully uses the tree structure of the rewards.

Performance of OLOP

Define

- $\beta \geq 0$ such that $\mathbb{P}(\text{Random path is } \epsilon\text{-optimal}) = O(\epsilon^{\beta})$.
- or $\kappa \overset{\text{def}}{=} K\gamma^{\beta} \in [1, K]$ the branching factor of the set of near-optimal sequences.
- or the near-optimality dimension, $d = \frac{\log \kappa}{\log 1/\gamma}$.

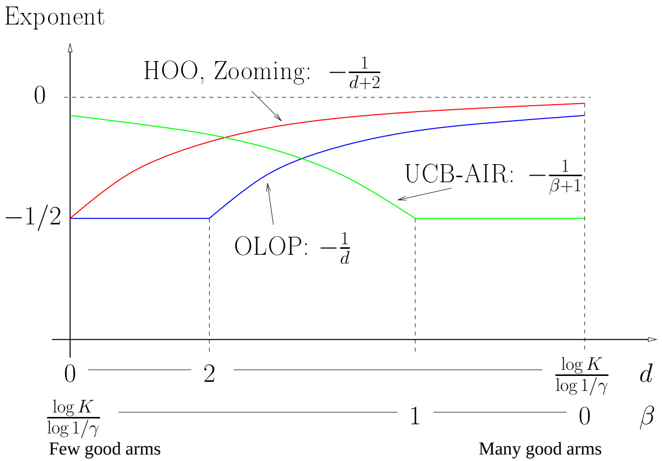
Theorem 4 (Loss of OLOP).

After n calls to the generative model,

$$\mathbb{E}r_n = f(x^*) - \mathbb{E}f(x(n)) = \begin{cases} \widetilde{O}\left(n^{-1/d}\right) & \textit{if } d \geq 2 \\ \widetilde{O}\left(1/\sqrt{n}\right) & \textit{if } d < 2 \end{cases}$$

Much better than HOO! As good as OPD for $d \geq 2$.

Comparison: OLOP, HOO, Zooming, UCB-AIR



Optimistic Planning in MDPs

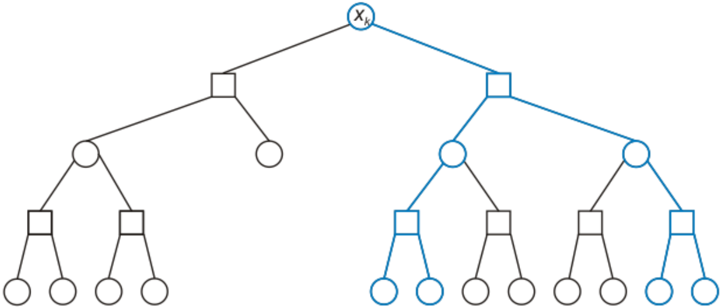
Stochastic transitions, but assume that the number of next states is finite.

Here a policy is no more a sequence of actions

OP-MDP [Bușoniu and Munos, 2012]:

- The root = current state.
- For $t = 1$ to n:
    - Compute the B-value of all nodes of the current sub-tree
    - Compute the optimistic policy
    - Select a leaf of the optimistic sub-tree and expand it (generates transitions to next states using the model)
- Return first action of the best policy

Illustration of OP-MDP

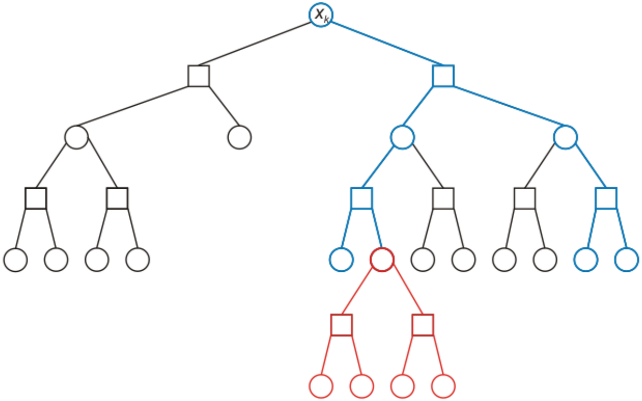


B-values: upper-bounds on the optimal value function $V^*(s)$:

$$\begin{aligned} B(s) &= \frac{1}{1-\gamma} \text{ for leaves} \\ B(s) &= \max_a \sum_{s'} p(s'|s,a)\big[r(s,a,s') + \gamma B(s')\big] \end{aligned}$$

Compute the **optimistic policy** π^+.

Optimistic Planning in MDPs



Expand leaf in π^+ with largest contribution: $\arg\max_{s\in\mathcal{L}} P(s)\frac{\gamma^{d(s)}}{1-\gamma}$, where $P(s)$ is the probability to reach s when following π^+.

Performance analysis of OP-MDP

Define X_ϵ the set of states

- whose "contribution" is at least ϵ
- and that belong to an ϵ-optimal policy

Near-optimality planning dimension: Define the measure of complexity of planning in the MDP as the smallest $d \geq 0$ such that $|X_\epsilon| = O(\epsilon^{-d})$.

Theorem 5.
The performance of OP-MDP is $r_n = O(n^{-1/d})$.

The performance depends on the quantity of states that contribute significantly to near-optimal policies

Illustration of the performance

Reminder: $r_n = O(n^{-1/d})$.

Uniform rewards and probabilities $d = \frac{\log K + \log N}{\log 1/\gamma}$ (uniform planning)

Structured rewards, uniform probabilities $d = \frac{\log N}{\log 1/\gamma}$ (uniform planning for a single policy)

Uniform rewards, concentrated probabilities $d \to \frac{\log K}{\log 1/\gamma}$ (planning in deterministic systems)

Structured rewards, concentrated probabilities $d \to 0$ (exponential rate)

Remarks: d is small when

- Structured rewards
- Heterogeneous transition probabilities

Towards d-minimax lower bounds

Let $\mathcal{M}_d$ the set of MDPs with near-optimality planning dim. d
Upper-bound of OP-MDP uniformly over $\mathcal{M}_\beta$

$$\sup_{M \in \mathcal{M}_d} r_n(\mathcal{A}_{OP-MDP}, M) \leq O(n^{-1/d}).$$

We conjecture that we have a d-minimax lower-bound:

$$\sup_{\mathcal{A}} \inf_{M \in \mathcal{M}_d} \mathbb{E} r_n(\mathcal{A}, M) \geq \Omega(n^{-1/d}).$$

Conclusions on optimistic planning

- Perform optimistic search in policy space.
- In deterministic dynamics, deterministic rewards, can be seen as a direct application of optimistic optimization
- In stochastic rewards, the structure of the reward function can help estimation of paths given samples from other paths
- In MDPs the **near-optimality planning dimension** is a new measure of the quantity of states that need to be expanded (*the set of states that significantly contribute to near-optimal policies*)
- Fast rates when the MDP has structured rewards and heterogeneous transition probabilities.
- Applications to Bayesian Reinforcement learning and planning in POMDPs.

Conclusions

The **“optimism in the face of uncertainty” principle:**

- applies in a large class of decision making problems in stochastic and deterministic environments (in unstructured and structured problems)
- provides an efficient exploration of the search space by exploring the most promising areas first
- provides a natural transition from global to local search
- Performance depends on the “smoothness” of the function around the maximum w.r.t. some metric,
    - a measure of the quantity of near-optimal solutions,
    - and our knowledge or not of this metric.

About optimism

Optimists and pessimists inhabit different worlds, reacting to the same circumstances in completely different ways.

Learning to Hope, Daisaku Ikeda.

Habits of thinking need not be forever. One of the most significant findings in psychology in the last twenty years is that individuals can choose the way they think.

Learned Optimism, Martin Seligman.

Humans do not hold a positivity bias on account of having read too many self-help books. Rather, optimism may be so essential to our survival that it is hardwired into our most complex organ, the brain.

The Optimism Bias: A Tour of the Irrationally Positive Brain, Tali Sharot.

Thanks !!!

See the review paper

From bandits to Monte-Carlo Tree Search: The optimistic principle applied to optimization and planning.

are available from my web page:

`http://chercheurs.lille.inria.fr/∼munos/`