

# 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  $\nu_k$  (with support in  $[0, 1]$ ), whose law is unknown,
- At each time  $t$ , choose an arm  $k_t$  and receive reward  $x_t \stackrel{i.i.d.}{\sim} \nu_{k_t}$ .
- **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:

- **$\epsilon$ -greedy exploration**: choose apparent best action with proba  $1 - \epsilon$ , or random action with proba  $\epsilon$ ,
- **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  $\propto \exp(\beta \hat{X}_k)$  (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:

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

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  $\mu_k$ . Indeed:

$$\mathbb{P}(\hat{X}_{k,s} - \mu_k \geq \sqrt{\frac{3 \log(t)}{2s}}) \leq \frac{1}{t^3},$$

$$\mathbb{P}(\hat{X}_{k,s} - \mu_k \leq -\sqrt{\frac{3 \log(t)}{2s}}) \leq \frac{1}{t^3}$$

Reminder of Chernoff-Hoeffding inequality:

$$\mathbb{P}(\hat{X}_{k,s} - \mu_k \geq \epsilon) \leq e^{-2s\epsilon^2}$$

$$\mathbb{P}(\hat{X}_{k,s} - \mu_k \leq -\epsilon) \leq e^{-2s\epsilon^2}$$

## 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:

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



## 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:

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

Now, taking the expectation of both sides,

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

## Variants of UCB

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

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

Then the expected regret is bounded as:

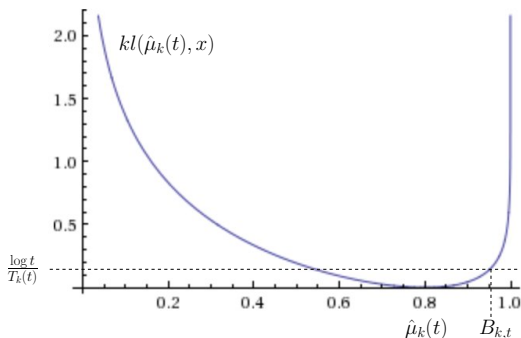
$$\mathbb{E}R_n \leq 10 \left( \sum_{k:\Delta_k > 0} \frac{\sigma_k^2}{\Delta_k} + 2 \right) \log(n).$$

# KL-UCB

[Garivier & Cappé, 2011] and [Maillard et al., 2011].

For Bernoulli distributions, define the kl-UCB

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



(non-asymptotic version of Sanov's theorem)

# KL-UCB

The regret of KL-UCB is then bounded as

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

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]:

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

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

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

where

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

## 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	<b>1</b>	0.7	<b>0.9</b>	<b>1</b>	1	<b>1</b>	<b>0.8</b>	<b>1</b>	
Reward arm 2	0.9	<b>0</b>	1	0	<b>0.4</b>	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

$$\sup_{\text{rewards}} \mathbb{E} R_n / n \rightarrow 0?$$

- If the policy of the player is deterministic, there exists a reward sequence such that the performance is arbitrarily poor  
 → 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  $\mathbb{E}R_n \leq \frac{\log K}{\eta} + (e-1)\eta nK$ . Thus, by choosing  $\eta = \sqrt{\frac{\log K}{(e-1)nK}}$ , it comes

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

Properties:

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

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

## Proof of Proposition 2 [part 1]

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

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

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

We thus have

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

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

## 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  $k = 1, \dots, n$ .

## Proof of Proposition 2 [part 3]

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

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

## 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:

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

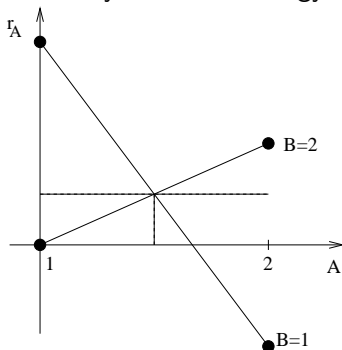
What happens?

## 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 $\rightarrow$ Nash equilibrium

Define the regret of A:

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

and that of B accordingly.

### Proposition 3.

*If both players perform a (Hannan) consistent regret-minimization strategy (i.e.  $R_n(A)/n \rightarrow 0$  and  $R_n(B)/n \rightarrow 0$ ), 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:  $R_n(A)/n \rightarrow 0$  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  $R_n(B)/n \rightarrow 0$  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

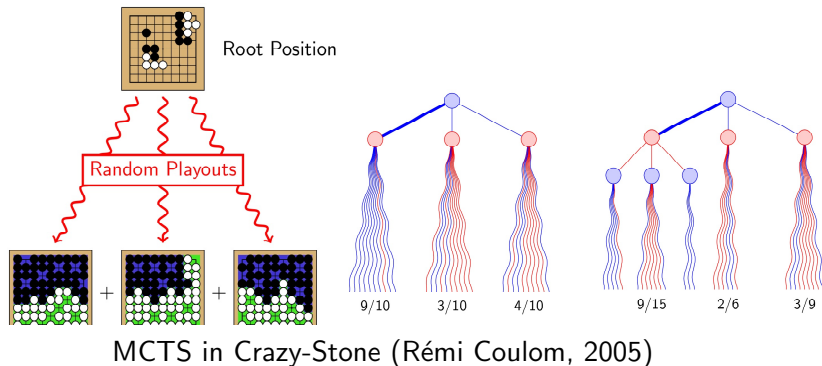
- In the 2-players Poker game, the Nash equilibrium is interesting (zero-sum game)
- A policy:  
information set (my cards + board + pot)  $\rightarrow$  probabilities over decisions (check, raise, fold)
- Space of policies is huge!



**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



Idea: use bandits at each node.

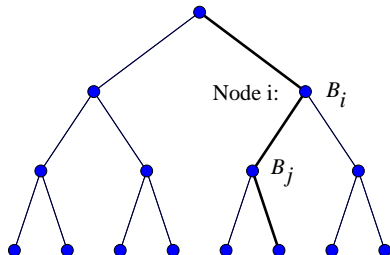
## UCB applied to Trees

Upper Confidence Bound (UCB) algo at each node

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

**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]



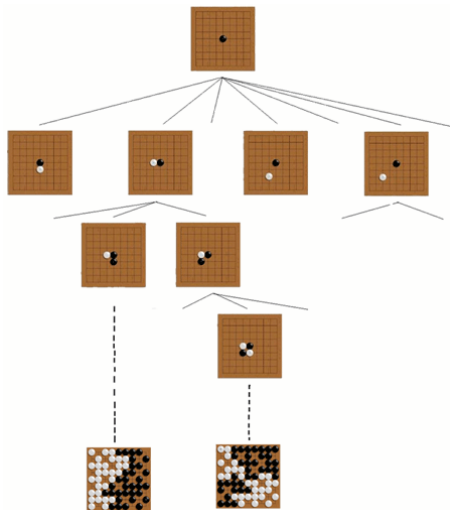
# The MoGo program

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

Features:

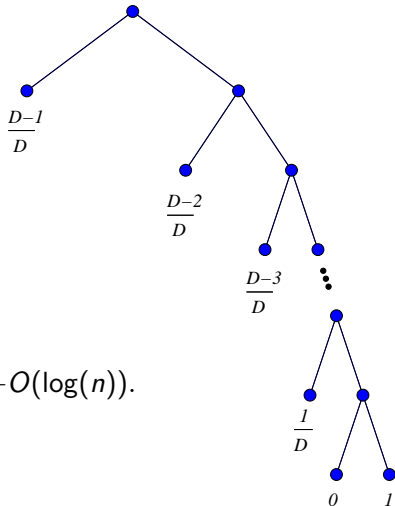
- Explore-Exploit with UCT
- Monte-Carlo evaluation
- Asymmetric tree expansion
- Anytime algo
- Use of features

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:

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

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  $f : X \rightarrow \mathbf{R}$ , assumed to be Lipschitz:

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

**Protocol:**

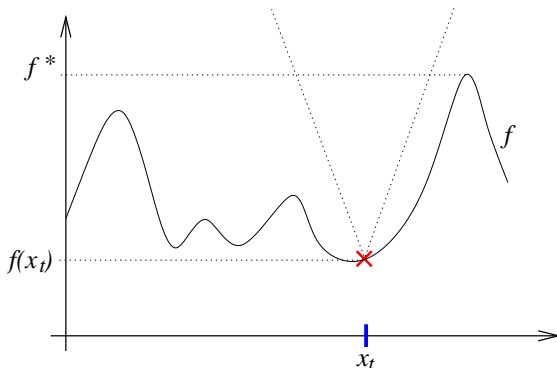
- For each time step  $t = 1, 2, \dots, n$  select a state  $x_t \in X$
- Observe  $f(x_t)$
- Return a state  $x(n)$

Performance assessed in terms of the loss

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

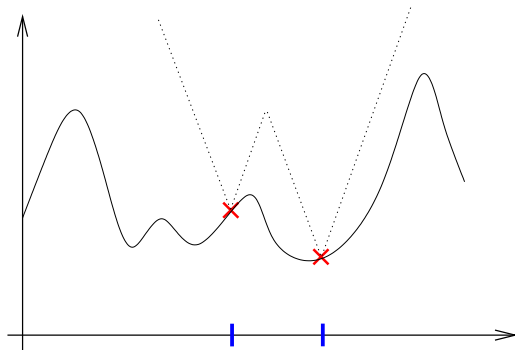
where  $f^* = \sup_{x \in X} f(x)$ .

## Example in 1d



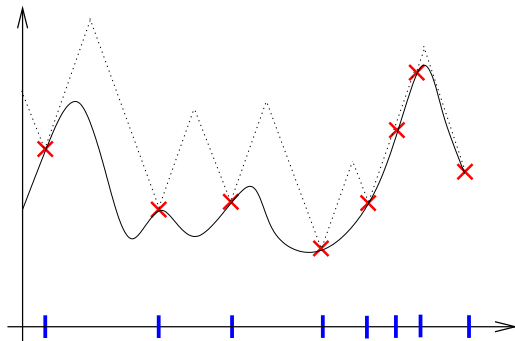
Lipschitz property  $\rightarrow$  the evaluation of  $f$  at  $x_t$  provides a first upper-bound on  $f$ .

## 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”**

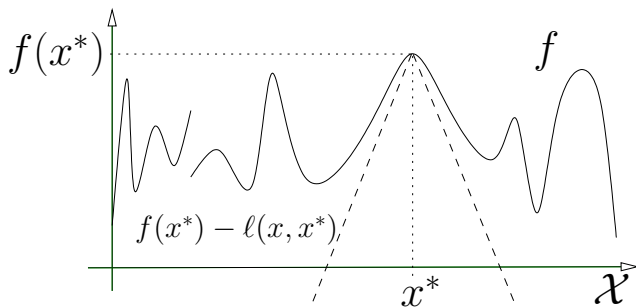
## Several issues

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  $\ell$ ?
4. How to handle noise?

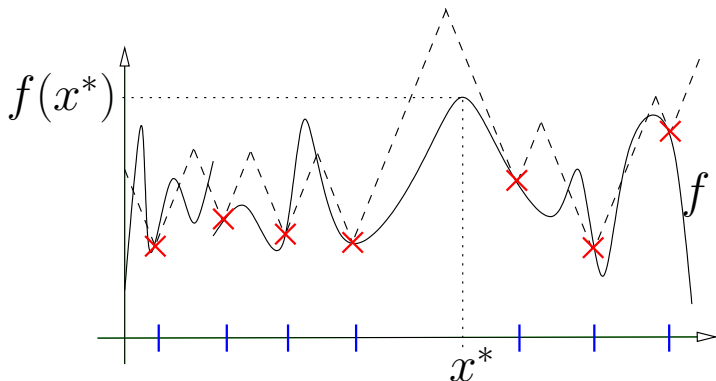
## Local smoothness property

Assumption:  $f$  is **“locally smooth”** around its max. w.r.t.  $\ell$  where  $\ell$  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:

$$B_i = f(x_i) + \text{diam}_\ell(X_i)$$

- Select the cell with highest bound

$$I_t = \underset{i}{\operatorname{argmax}} B_i.$$

- Expand  $I_t$ : refine the grid and evaluate  $f$  in children cells
- Return  $x(n) \stackrel{\text{def}}{=} \operatorname{argmax}_{\{x_t\}_{1 \leq t \leq n}} f(x_t)$



## 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  $\epsilon$ -optimal states

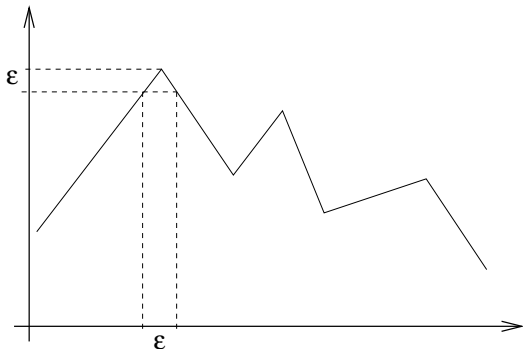
$$X_\epsilon \stackrel{\text{def}}{=} \{x \in X, f(x) \geq f^* - \epsilon\}$$

can be covered by  $C\epsilon^{-d}$   $\ell$ -balls of radius  $\epsilon$ .

## Example 1:

Assume the function is piecewise linear at its maximum:

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

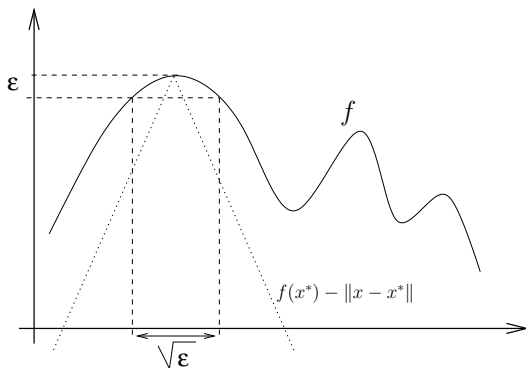


Using  $\ell(x, y) = \|x - y\|$ , it takes  $O(\epsilon^0)$  balls of radius  $\epsilon$  to cover  $X_\epsilon$ . Thus  $d = 0$ .

## Example 2:

Assume the function is locally quadratic around its maximum:

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

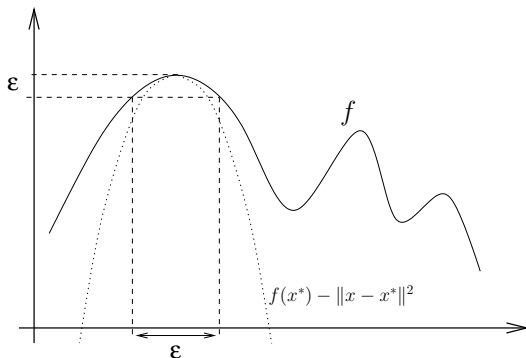


For  $\ell(x, y) = \|x - y\|$ , it takes  $O(\epsilon^{-D/2})$  balls of radius  $\epsilon$  to cover  $X_\epsilon$  (of size  $O(\epsilon^{D/2})$ ). Thus  $d = D/2$ .

## Example 2:

Assume the function is locally quadratic around its maximum:

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

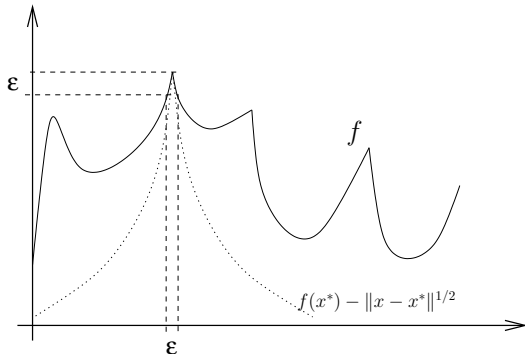


For  $\ell(x, y) = \|x - y\|^2$ , it takes  $O(\epsilon^0)$   $\ell$ -balls of radius  $\epsilon$  to cover  $X_\epsilon$ . Thus  $d = 0$ .

## 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  $\alpha$ -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  $\ell$ .

## Analysis of DOO (deterministic case)

Assume that the  $\ell$ -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*

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

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

## 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.

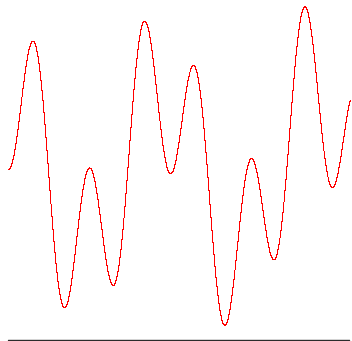
The performance of DOO heavily relies on our knowledge of the true local smoothness.



## Experiments [1]

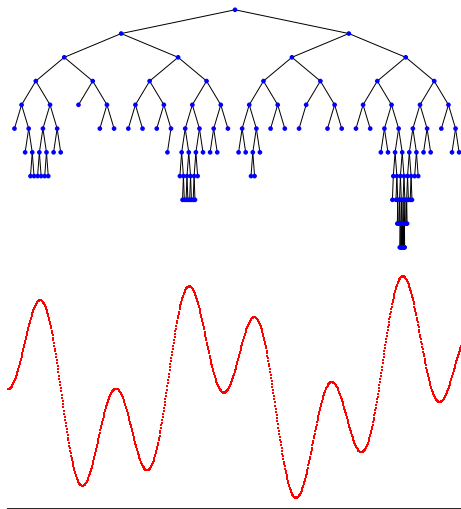
$f(x) = \frac{1}{2}(\sin(13x)\sin(27x) + 1)$  satisfies the local smoothness assumption  $f(x) \geq f(x^*) - \ell(x, x^*)$  with

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



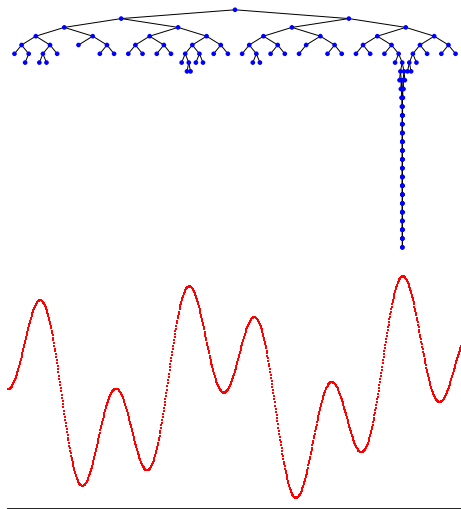
## Experiments [2]

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 $\ell_1$ ( $d = 1/2$ )	DOO with $\ell_2$ ( $d = 0$ )
50	$1.25 \times 10^{-2}$	$2.53 \times 10^{-5}$	$1.20 \times 10^{-2}$
100	$8.31 \times 10^{-3}$	$2.53 \times 10^{-5}$	$1.67 \times 10^{-7}$
150	$9.72 \times 10^{-3}$	$4.93 \times 10^{-6}$	$4.44 \times 10^{-16}$

Loss  $r_n$  for different values of  $n$  for a uniform grid and DOO with the two semi-metric  $\ell_1$  and  $\ell_2$ .

## 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?

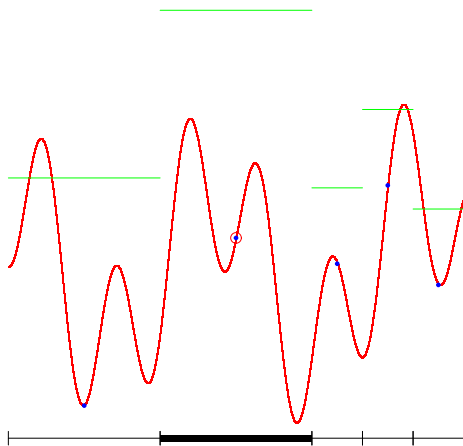
## DIRECT algorithm [Jones et al., 1993]

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  $L$ .

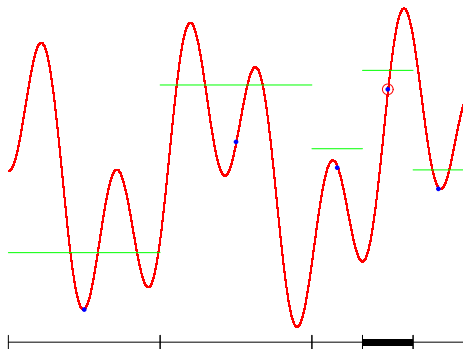
## Illustration of DIRECT

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  $\ell$  and for any function locally smooth w.r.t.  $\ell$ .

[Munos, 2011]

- Expand several leaves simultaneously
- SOO expands at most one leaf per depth
- SOO expands a leaf only if its value is larger than 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**

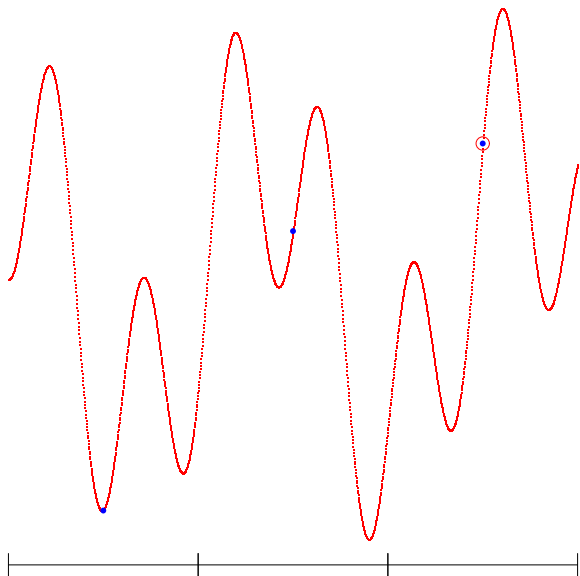
            Expand the node  $(h, i)$ , Set  $v_{\max} = f(x_{h,i})$ , Set  $t = t + 1$

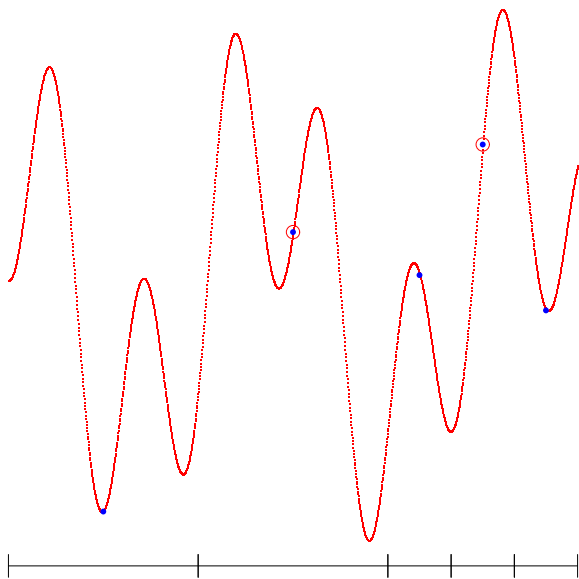
**if**  $t = n$  **then** return  $x(n) = \arg \max_{(h,i) \in \mathcal{T}_n} x_{h,i}$

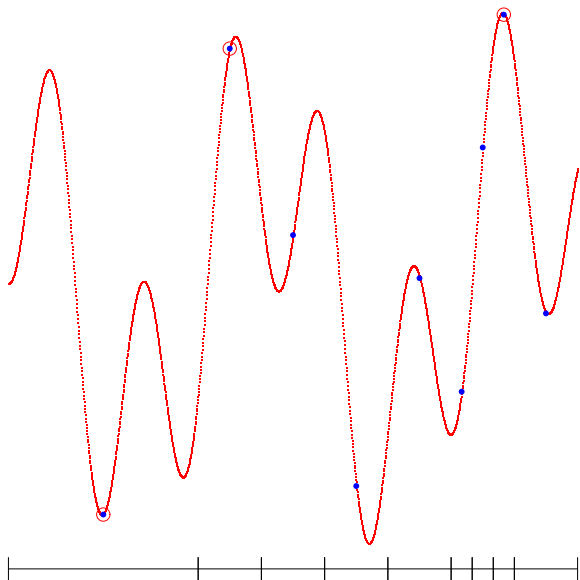
**end if**

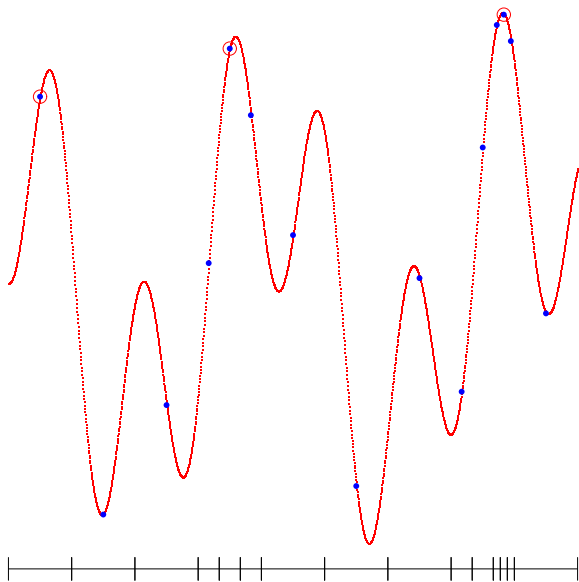
**end for**

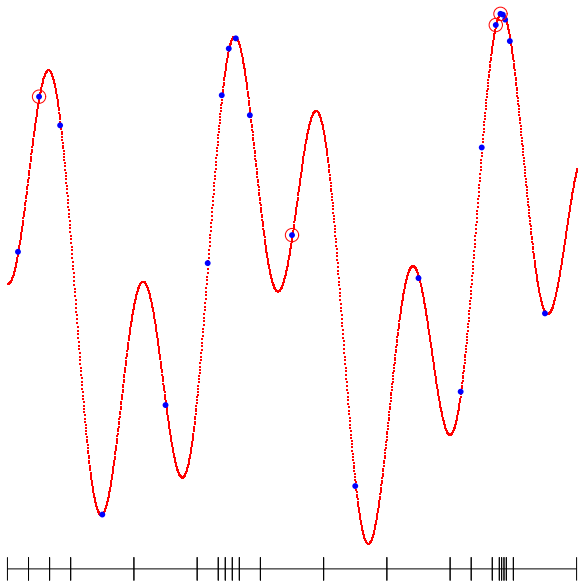
**end while.**

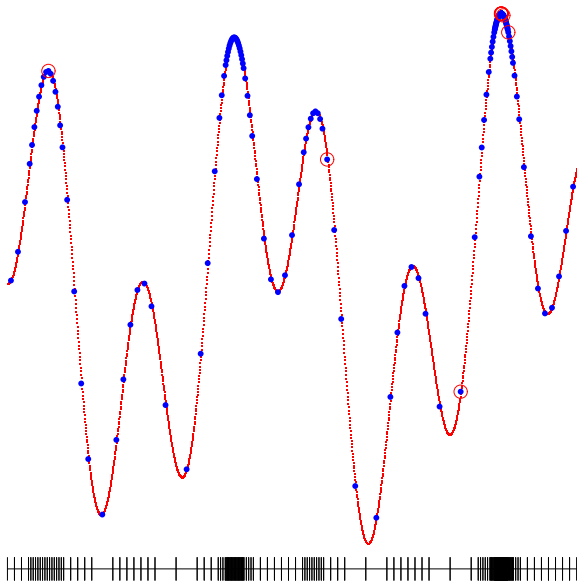




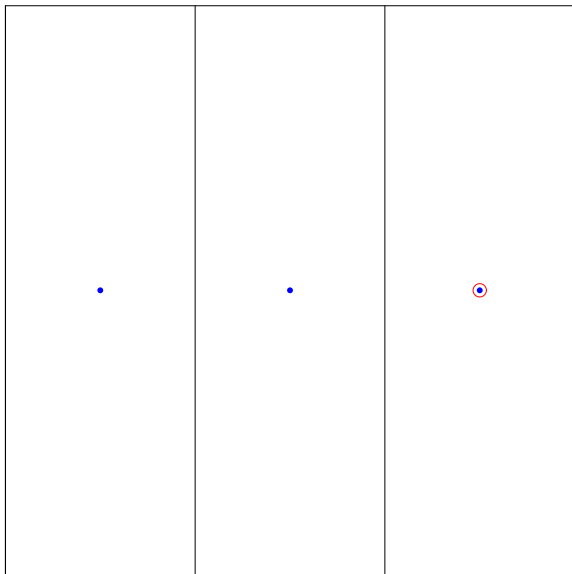


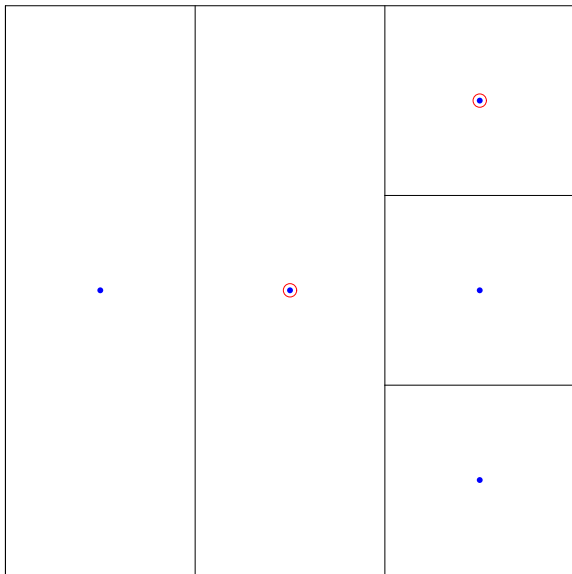


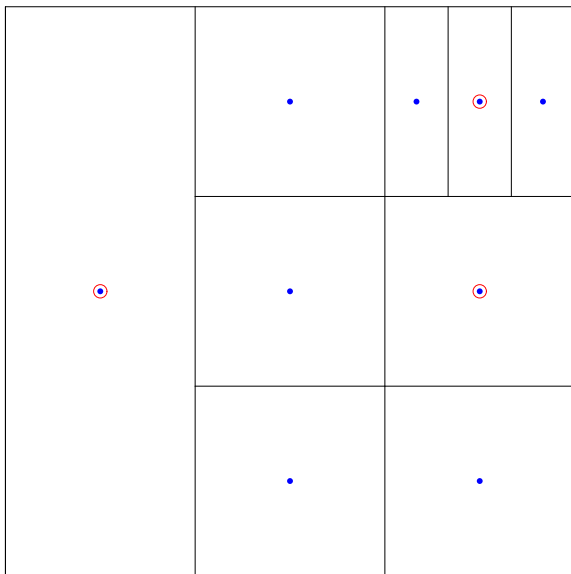


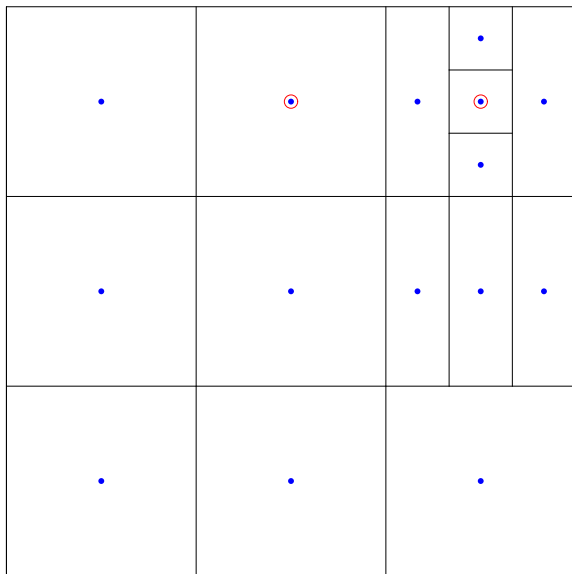


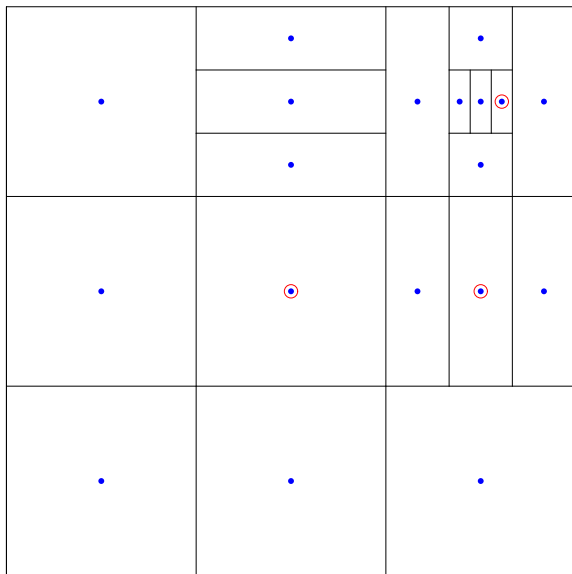


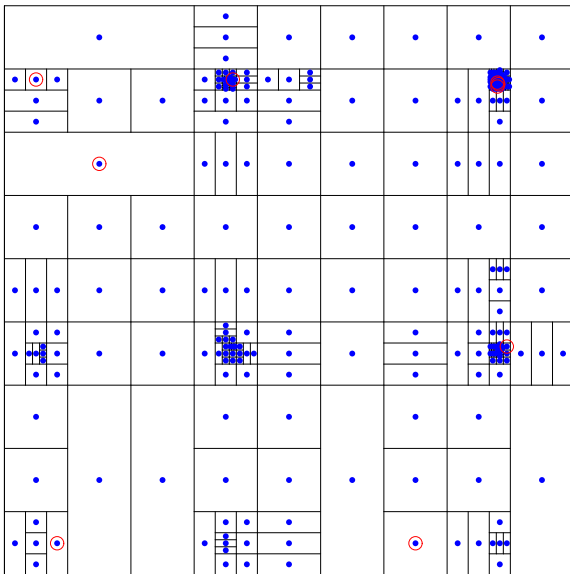












## Performance of SOO

### Theorem 2.

For any semi-metric  $\ell$  such that

- $f$  is locally smooth w.r.t.  $\ell$
- The  $\ell$ -diameter of cells of depth  $h$  is  $c\gamma^h$
- The near-optimality dimension of  $f$  w.r.t.  $\ell$  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  $\mathbb{E}r_n = \tilde{O}(n^{-1/d})$ .

# Performance of SOO

## Remarks:

- Since the algorithm does not depend on  $\ell$ , the analysis holds for the best possible choice of the semi-metric  $\ell$  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 $\ell_1$	DOO with $\ell_2$
50	$3.56 \times 10^{-4}$	$1.25 \times 10^{-2}$	$2.53 \times 10^{-5}$	$1.20 \times 10^{-2}$
100	$5.90 \times 10^{-7}$	$8.31 \times 10^{-3}$	$2.53 \times 10^{-5}$	$1.67 \times 10^{-7}$
150	$1.92 \times 10^{-10}$	$9.72 \times 10^{-3}$	$4.93 \times 10^{-6}$	$4.44 \times 10^{-16}$

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

Example:

- $f$  is locally  $\alpha$ -smooth around its maximum:

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

for some  $\alpha > 0$ .

- SOO algorithm does not require the knowledge of  $\ell$ ,
- 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!

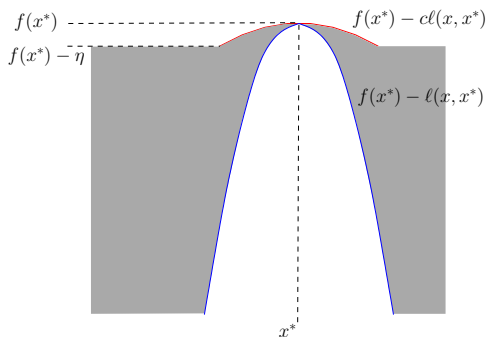
(Extends to the case  $f(x^*) - f(x) \approx \sum_{i=1}^D c_i |x_i^* - x_i|^{\alpha_i}$ )

## The case $d = 0$

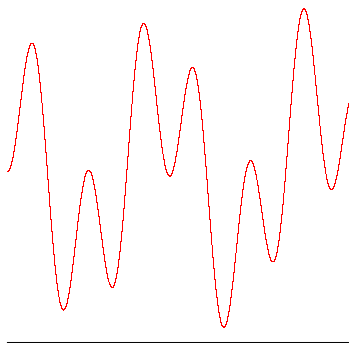
More generally, any function whose **upper- and lower envelopes around  $x^*$  have the same shape**:  $\exists c > 0$  and  $\eta > 0$ , such that

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

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

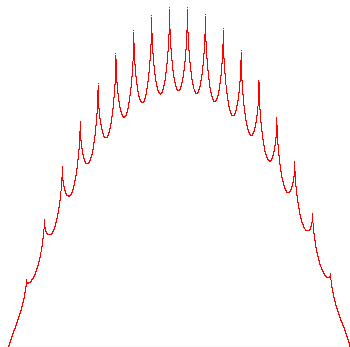


## Example of functions for which $d = 0$

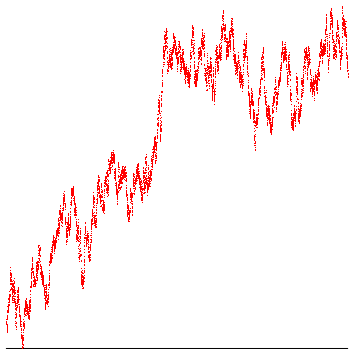


$$\ell(x, y) = c\|x - y\|^2$$

## Example of functions with $d = 0$



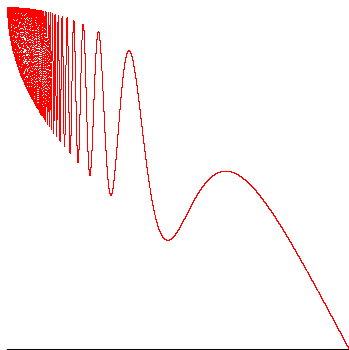
$$\ell(x, y) = c\|x - y\|^{1/2}$$

$d = 0?$ 

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

$$d > 0$$

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



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

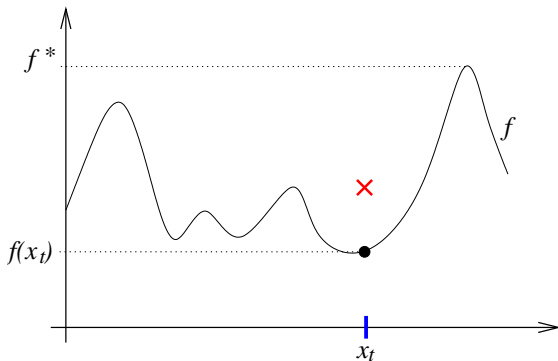
- **SOO is much more general than DIRECT**: the function is only **locally smooth** and the space is **semi-metric**.
- **Finite-time analysis of SOO** (whereas only a consistency property  $\lim_{n \rightarrow \infty} r_n = 0$  is available for DIRECT in [Finkel and Kelley, 2004])
- **SOO is a rank-based 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...



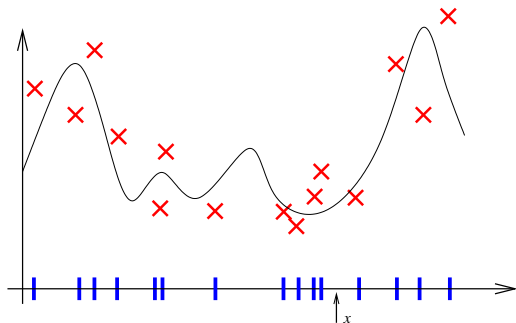
## How to handle noise?

The evaluation of  $f$  at  $x_t$  is perturbed by noise:

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

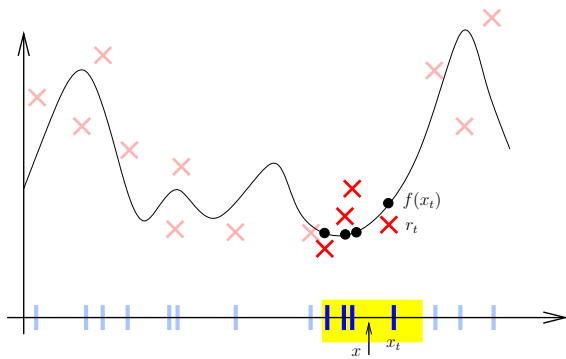


## Where should one sample next?



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 extend 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

$$\hat{\mu}_{i,t} + c \sqrt{\frac{\log n}{T_i(t)'}}$$

- 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  $\ell$  such that

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

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

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

# 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:

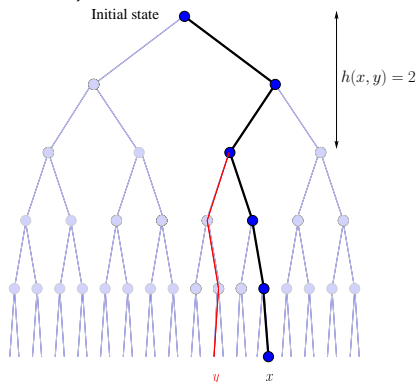
- From current state  $s_k$ , perform planning using  $n$  calls to the model and recommend action  $a(n)$ ,
- Play  $a(n)$ , observe next state  $s_{k+1}$ , and repeat

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

# 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.  $\ell$
- $\rightarrow$  Use optimistic search



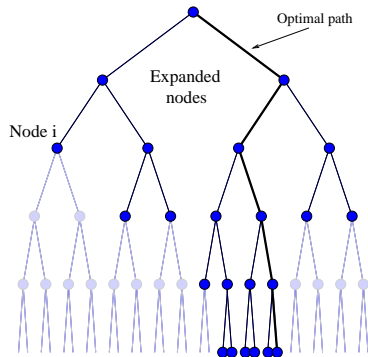
# OPD algorithm

Optimistic Planning in Deterministic systems:

- Define the B-values:

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

- We have  $B_i \geq \max_{x \in \mathcal{X}_i} f(x)$
- 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  $\beta$  such that  $\mathbb{P}(\text{Random path is } \epsilon\text{-optimal}) = O(\epsilon^\beta)$ .

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

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

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

**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**

## $\kappa$ -minimax lower bounds

Let  $\mathcal{M}_\kappa$  the set of problems with coefficient  $\kappa$ .

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  $\kappa$ -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  $I$ . Reciprocally,  $I$  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:

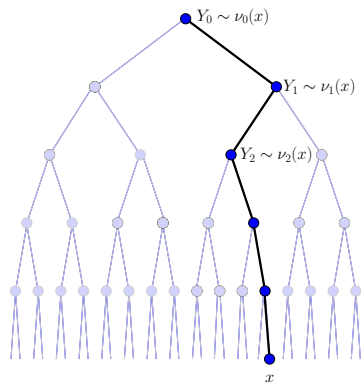
- Reward along a path  $x$ :

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

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  $x$ :

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

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

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  $\kappa \stackrel{\text{def}}{=} K\gamma^\beta \in [1, K]$  the branching factor of the set of near-optimal sequences.
- or the near-optimality dimension,  $d = \frac{\log \kappa}{\log 1/\gamma}$ .

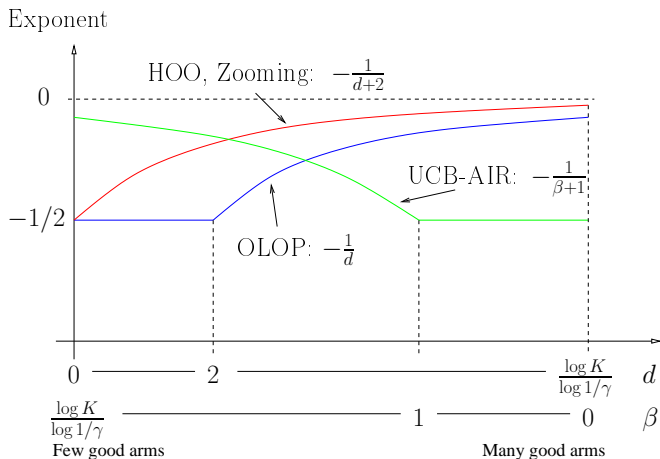
### Theorem 4 (Loss of OLOP).

After  $n$  calls to the generative model,

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

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.

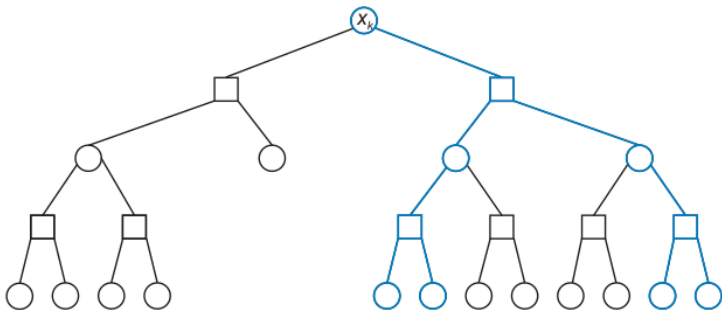
Here a policy is no more a sequence of actions

**OP-MDP** [Buşoniu 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



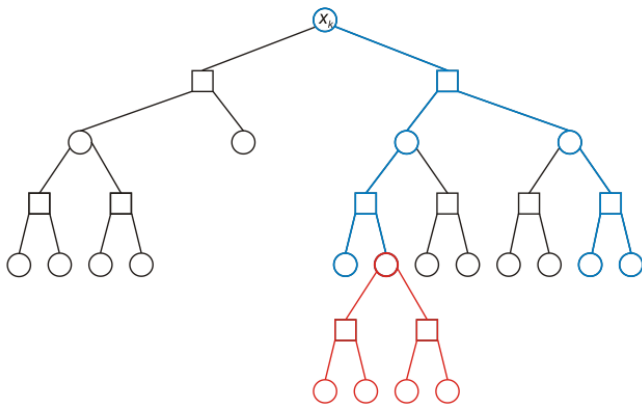
B-values: upper-bounds on the optimal value function  $V^*(s)$ :

$$B(s) = \frac{1}{1-\gamma} \text{ for leaves}$$

$$B(s) = \max_a \sum_{s'} p(s'|s, a) [r(s, a, s') + \gamma B(s')]$$

Compute the **optimistic policy**  $\pi^+$ .

# Optimistic Planning in MDPs



Expand leaf in  $\pi^+$  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  $\pi^+$ .

# Performance analysis of OP-MDP

Define  $X_\epsilon$  the set of states

- whose “contribution” is at least  $\epsilon$
- and that belong to an  $\epsilon$ -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**  $d \rightarrow \frac{\log K}{\log 1/\gamma}$   
(planning in deterministic systems)

**Structured rewards, concentrated probabilities**  $d \rightarrow 0$   
(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  $\mathcal{M}_d$

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

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/>