

Hybridization of dynamic optimization methodologies

Jérémie Decock

Advisor: Olivier Teytaud

I-Lab: LRI - Inria & Artelys

Ph.D. Defense

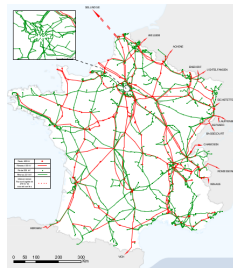
November 28, 2014



Subject

Comparison and hybridation of Sequential Decision Making (SDM) algorithms

- ▶ Long-term reward
- ▶ Interdependence among decisions
- ▶ Exponential number of solutions



Application field

Decision aids for Power systems management

Modelling Sequential Decision Making Problems

Notations

- ▶ s_t = state at time t
- ▶ a_t = action (decision) at time t
- ▶ $s' = s_{t+1}$

- ▶ $T(s, a, s')$ = probability of $s \rightarrow s'$ when action a
- ▶ $r(s, a)$ = reward when action a in s
- ▶ $\pi(s)$ = action to execute in s (policy)

Bellman Methods

Bellman Values

The *Bellman equation* gives the best value we can expect for any given state (assuming the optimal policy π^* is followed).

$$V^{\pi^*}(\mathbf{s}) = r(\mathbf{s}) + \gamma \max_{\mathbf{a} \in \mathcal{A}} \left[\sum_{\mathbf{s}' \in \mathcal{S}} T(\mathbf{s}, \mathbf{a}, \mathbf{s}') V(\mathbf{s}') \right]$$

The optimal (stationary) policy π^* is defined using the best expected value V^{π^*} and using the principle of *Maximum Expected Utility* as follow

$$\pi^*(\mathbf{s}) = \arg \max_{\mathbf{a} \in \mathcal{A}} \left[\sum_{\mathbf{s}' \in \mathcal{S}} T(\mathbf{s}, \mathbf{a}, \mathbf{s}') V^{\pi^*}(\mathbf{s}') \right]$$

Bellman Methods

Computing Bellman Values

$$V^{\pi^*}(\mathbf{s}) = r(\mathbf{s}) + \gamma \max_{\mathbf{a} \in \mathcal{A}} \left[\sum_{\mathbf{s}' \in \mathcal{S}} T(\mathbf{s}, \mathbf{a}, \mathbf{s}') V(\mathbf{s}') \right]$$

There are $|\mathcal{S}|$ Bellman equations, one for each state. This system of equations cannot be solved analytically because Bellman equations contains non-linear terms (due to the "max" operator).

Bellman Methods

[Bellman57]

Value Iteration: the stationary case

Bellman update is used in Value Iteration to update V at each iteration.

$$V_{i+1}(\mathbf{s}) \leftarrow r(\mathbf{s}) + \gamma \max_{\mathbf{a} \in \mathcal{A}} \left[\sum_{\mathbf{s}' \in \mathcal{S}} T(\mathbf{s}, \mathbf{a}, \mathbf{s}') V_i(\mathbf{s}') \right]$$

State of the art in Power Systems

Summary

	Pros	Cons
S(D)DP	large constrained \mathcal{A} polynomial time decision making asymptotically find the optimum	not anytime convex problems only (SDDP) small \mathcal{S} Markovian random process
DPS	anytime large \mathcal{S} works with non linear functions no random process constraint	slow on large \mathcal{A} hardly handles decision constraints

Contribution

Merge both approaches !

Direct Value Search

Combine instantaneous reward and valorization:

$$\Pi(\mathbf{s}_t) = \arg \max_{\mathbf{a}_t} r(\mathbf{s}_t, \mathbf{a}_t) + V(\mathbf{s}_{t+1})$$

$$V(\mathbf{s}_{t+1}) = \underbrace{\alpha_t \cdot \mathbf{s}_{t+1}}_{\text{linear}}$$

$$\alpha_t = \underbrace{\pi_{\theta}(\mathbf{s}_t)}_{\text{non linear}}$$

- ▶ Given θ , decision making solved as a LP
- ▶ Non-linear mapping for choosing the parameters of the LP from the current state

Requires the optimization of θ (noisy black-box optimization problem)

Direct Value Search

Step 1 (offline): compute $\pi_{\theta}(\cdot)$

Build parametric policy π_{θ}

Require:

- a parametric policy $\pi_{\theta}(\cdot)$ where π_{θ} is a mapping from \mathcal{S} to \mathcal{A} ,
- a Stochastic Decision Process SDP,
- an initial state s

Ensure:

- a parameter $\hat{\theta}^*$ leading to a policy $\pi_{\hat{\theta}^*}(\cdot)$

Find a parameter $\hat{\theta}^*$ maximizing the expectation of the following fitness function

$\theta \mapsto \text{Simulate}(s, \text{SDP}, \pi_{\theta})$

with a given non-linear noisy optimization algorithm (e.g. SA-ES, CMA-ES)

return $\hat{\theta}^*$

Direct Value Search

Step 1 (offline): compute $\pi_\theta(\cdot)$

Simulate(s_0 , SDP, π_θ)

$r \leftarrow 0$

for $t \leftarrow t_0, t_0 + h, t_0 + 2h, \dots, T$ **do**

$l_{t\dots k} \leftarrow \text{get_forecast}(\cdot)$

$\alpha \leftarrow \pi_\theta(\mathbf{s}_t^+)$

$\mathbf{a}_{t\dots k} \leftarrow \arg \max_{\mathbf{a}} r(\mathbf{a}_{t\dots k}, l_{t\dots k}, \mathbf{s}_t) + \alpha_{\mathbf{s}}^\top \cdot \mathbf{s}_{t+k-1}$

$r \leftarrow r + \text{SDP_reward}(\mathbf{s}_t, \mathbf{a}_{t\dots h}, l_{t\dots h})$ \leftarrow not necessarily linear

$\mathbf{s}_{t+h} \leftarrow \text{SDP_transition}(\mathbf{s}_t, \mathbf{a}_{t\dots h}, l_{t\dots h})$ \leftarrow not necessarily linear

end for

return r

Direct Value Search

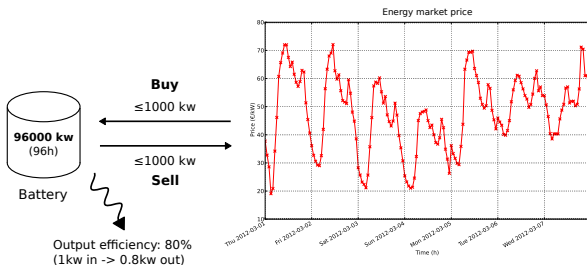
Step 2 (online): use $\pi_\theta(\cdot)$ to solve the actual problem

The offline optimisation of π_θ can be stopped at any time.

Then

- ▶ we have π_θ , an approximation of state's marginal value
- ▶ we can use it to solve the actual problem, the same manner as in *Simulate*

DVS: A proof of concept



Goal: find a policy which maximises gains

Buy (and store) when the market price is low, sell when the market price is high.

- ▶ The market price is stochastic.
- ▶ 10 constrained batteries.

Baselines

Recourse planning without final valorization

Bellman values \rightarrow null ($\alpha = 0$)

$$\Pi(\mathbf{s}, t) = \arg \max_{\mathbf{a}_t} \max_{\mathbf{a}_{t+1}, \dots, \mathbf{a}_{t+k-1}} \mathbb{E} [r_t + \dots + r_{t+k-1}]$$

Recourse planning with constant marginal valorization

Bellman values \rightarrow linear function of total stock ($\alpha = \text{constant}$)

$$\Pi(\mathbf{s}, t) = \arg \max_{\mathbf{a}_t} \max_{\mathbf{a}_{t+1}, \dots, \mathbf{a}_{t+k-1}} \mathbb{E} [r_t + \dots + r_{t+k-1} + \alpha \cdot \mathbf{s}_{t+k-1}]$$

with optimized constant marginal valorization α .



Noisy Optimization: Lower Bounds on Runtimes

Introduction

The aim of this 2nd contribution

- ▶ Study convergence rate for numerical optimization with noisy objective function
- ▶ Reduce the gap between upper and lower bounds
 - ▶ for a given family of noisy objective functions
 - ▶ under some assumptions

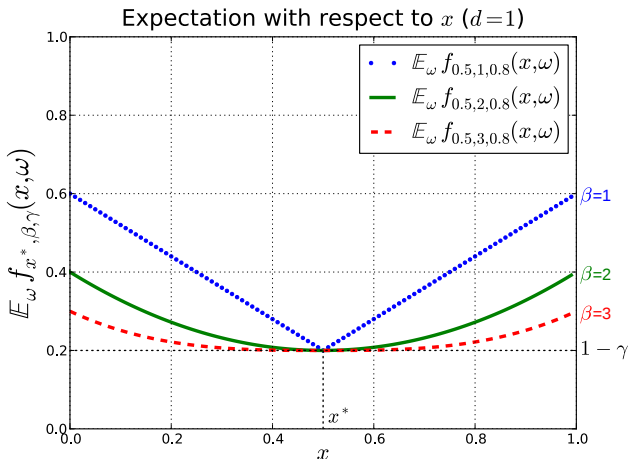
Considered family F of objective functions

$$f_{\mathbf{x}^*,\beta,\gamma} : [0, 1]^d \rightarrow \{0, 1\}$$

$$x \mapsto \mathcal{B} \left(\underbrace{\gamma \left(\frac{\|\mathbf{x} - \mathbf{x}^*\|}{\sqrt{d}} \right)^\beta}_{p} + (1 - \gamma) \right)$$

- ▶ $f_{\mathbf{x}^*,\beta,\gamma} \in F$: a stochastic objective function (a random variable)
- ▶ $\mathcal{B}(p)$: the Bernoulli distribution with parameter p
- ▶ $d \in \mathbb{N}^*$: the dimension of the domain of $f_{\mathbf{x}^*,\beta,\gamma}$
- ▶ $\mathbf{x}^* \in [0, 1]^d$: the optimum
- ▶ $\beta \in \mathbb{R}_+^*$: the "flatness" of the expectation $\mathbb{E}f$ around \mathbf{x}^*
- ▶ $\gamma \in [0, 1]$: a noise parameter (variance at \mathbf{x}^*)

Considered objective functions



Considered objective functions

- ▶ we assume that the optimum is unique
- ▶ we consider noisy optimization in the case of local convergence

Lower bound \rightarrow concerns all families including F

The experimental setting for our noisy optimization setting

Require: ω , ω' , \mathbf{x}^* , β , γ and (unknown).

for all $n = 1, 2, 3, \dots$ **do**

$\mathbf{x}_{x^*,n,\omega,\omega'} = \text{Optimize}(\mathbf{x}_{x^*,1,\omega,\omega'}, \dots, \mathbf{x}_{x^*,n-1,\omega,\omega'}, y_1, \dots, y_{n-1}, \omega')$

if $\omega_n \leq \mathbb{E}f_{\mathbf{x}^*,\beta,\gamma}(\mathbf{x}_{x^*,n,\omega,\omega'})$ **then**

$y_n = 1$

else

$y_n = 0$

end if

end for

return $\mathbf{x}_{x^*,n,\omega,\omega'}$

The framework requires:

- ▶ ω uniform random variable for simulating the Bernoulli in f
- ▶ ω' a random seed of the algorithm (optimizer's internal randomness)
- ▶ \mathbf{x}^* the optimum (that minimizes $\mathbb{E}_\omega f_{\mathbf{x}^*,\beta,\gamma}(\mathbf{x}, \omega)$)
- ▶ β and γ two fixed parameters of f

The experimental setting for our noisy optimization setting

Require: ω , ω' , \mathbf{x}^* , β , γ and (unknown).

for all $n = 1, 2, 3, \dots$ **do**

$\mathbf{x}_{\mathbf{x}^*, n, \omega, \omega'} = \text{Optimize}(\mathbf{x}_{\mathbf{x}^*, 1, \omega, \omega'}, \dots, \mathbf{x}_{\mathbf{x}^*, n-1, \omega, \omega'}, y_1, \dots, y_{n-1}, \omega')$

if $y_n \leq \mathbb{E}f_{\mathbf{x}^*, \beta, \gamma}(\mathbf{x}_{\mathbf{x}^*, n, \omega, \omega'})$ **then**

$y_n = 1$

else

$y_n = 0$

end if

end for

return $\mathbf{x}_{\mathbf{x}^*, n, \omega, \omega'}$

It's an iterative process. For each iteration:

- ▶ *Optimize* (the optimization algorithm) returns the next point to visit
 - ▶ looking for \mathbf{x}^*
 - ▶ according to some inputs
- ▶ This point is evaluated by the *fitness function* (if-then-else statement)

The experimental setting for our noisy optimization setting

Require: ω , ω' , \mathbf{x}^* , β , γ and (unknown).

for all $n = 1, 2, 3, \dots$ **do**

$\mathbf{x}_{\mathbf{x}^*, n, \omega, \omega'} = \text{Optimize}(\mathbf{x}_{\mathbf{x}^*, 1, \omega, \omega'}, \dots, \mathbf{x}_{\mathbf{x}^*, n-1, \omega, \omega'}, y_1, \dots, y_{n-1}, \omega')$

if $\omega_n \leq \mathbb{E}f_{\mathbf{x}^*, \beta, \gamma}(\mathbf{x}_{\mathbf{x}^*, n, \omega, \omega'})$ **then**

$y_n = 1$

else

$y_n = 0$

end if

end for

return $\mathbf{x}_{\mathbf{x}^*, n, \omega, \omega'}$

Optimize makes its recommendation according to:

- ▶ the sequence of former visited points \mathbf{x}_n
- ▶ their binary noisy fitness values y_n
- ▶ the optimizer's internal randomness ω'

The experimental setting for our noisy optimization setting

Require: ω , ω' , \mathbf{x}^* , β , γ and (unknown).

for all $n = 1, 2, 3, \dots$ **do**

$\mathbf{x}_{\mathbf{x}^*, n, \omega, \omega'} = \text{Optimize}(\mathbf{x}_{\mathbf{x}^*, 1, \omega, \omega'}, \dots, \mathbf{x}_{\mathbf{x}^*, n-1, \omega, \omega'}, y_1, \dots, y_{n-1}, \omega')$

if $\omega_n \leq \mathbb{E}f_{\mathbf{x}^*, \beta, \gamma}(\mathbf{x}_{\mathbf{x}^*, n, \omega, \omega'})$ **then**

$y_n = 1$

else

$y_n = 0$

end if

end for

return $\mathbf{x}_{\mathbf{x}^*, n, \omega, \omega'}$

The fitness function $f_{\mathbf{x}^*, \beta, \gamma}$ outputs

- ▶ 1 if random ($= \omega_n$) is less than $\mathbb{E}f_{\mathbf{x}^*, \beta, \gamma}(\mathbf{x}_{\mathbf{x}^*, n, \omega, \omega'})$
- ▶ 0 otherwise

The experimental setting for our noisy optimization setting

Require: ω , ω' , \mathbf{x}^* , β , γ and (unknown).

for all $n = 1, 2, 3, \dots$ **do**

$\mathbf{x}_{\mathbf{x}^*, n, \omega, \omega'} = \text{Optimize}(\mathbf{x}_{\mathbf{x}^*, 1, \omega, \omega'}, \dots, \mathbf{x}_{\mathbf{x}^*, n-1, \omega, \omega'}, y_1, \dots, y_{n-1}, \omega')$

if $\omega_n \leq \mathbb{E}f_{\mathbf{x}^*, \beta, \gamma}(\mathbf{x}_{\mathbf{x}^*, n, \omega, \omega'})$ **then**

$y_n = 1$

else

$y_n = 0$

end if

end for

return $\mathbf{x}_{\mathbf{x}^*, n, \omega, \omega'}$

Eventually, the estimation of the optimum is returned

Sampling strategies

Sampling close to the current estimation of the optimum

- ▶ most evolution strategies do that

Sampling far from the current estimation of the optimum

- ▶ when f is learnable
- ▶ the optimizer can use a model of f to sample far from x^*
- ▶ *optimize's* outputs can have different meanings
 - ▶ be the most informative points to sample (exploration)
 - ▶ provide an estimate of $\arg \min \mathbb{E}f$ (recommendation)

These strategies lead to different convergence rates.

Our assumptions

We assume that the optimization algorithm

- ▶ does not require a model of the objective function
- ▶ samples close to the optimum (first strategy)

The *locality* assumption

$$\forall f \in F, \quad P \left(\forall i \leq n, \quad \|\mathbf{x}_i - \mathbf{x}^*\| \leq \frac{C(d)}{i^\alpha} \right) \geq 1 - \frac{\delta}{2}$$

- ▶ for some $0 < \delta < 1/2$
- ▶ $C(d) > 0$: a constant depending on d only
- ▶ $\alpha > 0$: convergence speed (large α implies a fast convergence)

Noisy optimization complexity

What is the best theoretical convergence rate of an optimization algorithm on $f \in F$ assuming this *locality assumption* ?

$$\forall f \in F, \quad P \left(\forall i \leq n, \quad \|\mathbf{x}_i - \mathbf{x}^*\| \leq \frac{C(d)}{i^\alpha} \right) \geq 1 - \frac{\delta}{2}$$

Question

What is the supremum of possible α ?

State of the art: upper and lower bounds

[Roley2010,Shamir,Chen88]

Consider α such that

$$\forall f \in F, \quad P\left(\forall i \leq n, \quad \|\mathbf{x}_i - \mathbf{x}^*\| \leq \frac{C(d)}{i^\alpha}\right) \geq 1 - \frac{\delta}{2}$$

	$\gamma = 1$ (small noise)	$\gamma < 1$ (large noise)
Proved rate for R-EDA	$\frac{1}{\beta} \leq \alpha$	$\frac{1}{2\beta} \leq \alpha$
Former lower bounds	$\alpha \leq 1$	$\alpha \leq \frac{1}{2}$
R-EDA experimental rates	$\alpha = \frac{1}{\beta}$	$\alpha = \frac{1}{2\beta}$
Rate by active learning	$\alpha = \frac{1}{2}$	$\alpha = \frac{1}{2}$

What we prove

Consider α such that

$$\forall f \in F, \quad P\left(\forall i \leq n, \quad \|\mathbf{x}_i - \mathbf{x}^*\| \leq \frac{C(d)}{i^\alpha}\right) \geq 1 - \frac{\delta}{2}$$

	$\gamma = 1$ (small noise)	$\gamma < 1$ (large noise)
Proved rate for R-EDA	$\frac{1}{\beta} \leq \alpha$	$\frac{1}{2\beta} \leq \alpha$
Former lower bounds	$\alpha \leq 1$	$\alpha \leq \frac{1}{2}$
This proof (lower bounds)	$\alpha \leq \frac{1}{\beta}$	$\alpha \leq \frac{1}{\beta}$ (recently: $1/4$ for $\beta = 2$)

Key ideas

Key point

if $\omega_i < \mathbb{E}f(x^*)$ or $\omega_i > \mathbb{E}f(x_i)$, the function evaluation is the same for all $x^* \Rightarrow$ let us show that this happens for most i .

Definitions

- ▶ $N =$ number of ω_i between $\mathbb{E}f(x^*)$ and $\mathbb{E}f(x_i)$ for $i \leq n$.
- ▶ $X_{n,\Omega}$ the set of points which can be chosen as n^{th} sampled point if the random sequence is $\Omega = (\omega_1, \dots, \omega_n, \dots)$.

$\text{Card } X_{n,\Omega} \leq 2^N$ We will show that locality $\Rightarrow N$ small. (intuition: N (depends on n) is the number of i which bring information; the locality assumption implies that N is small, i.e. we receive little information)

Lemma

N is probably upper bounded by $O(\sum_{i=1}^n 1/i^{\alpha\beta})$.

(i.e. for most evaluations, the result does not depend on x^* !)

Proof

probability of $\omega_i \in (\mathbb{E}f(x^*), \mathbb{E}f(x_i))$ is $O(1/i^{\alpha\beta})$ by locality assumption.

\Rightarrow hence the result by summation (for expectations).

\Rightarrow hence the result by Chebyshev (with large probability).

Theorem

Theorem

Assume the objective function $f \in F$

Assume the *locality assumption*

$$\forall f \in F, \quad P \left(\forall i \leq n, \quad \|\mathbf{x}_i - \mathbf{x}^*\| \leq \frac{C(d)}{i^\alpha} \right) \geq 1 - \frac{\delta}{2}$$

Then $\alpha \leq 1/\beta$.

Proof of the theorem (1)

Let us show that $\alpha\beta \leq 1$

In order to do so, let us assume, in order to get a contradiction, that $\alpha\beta > 1$

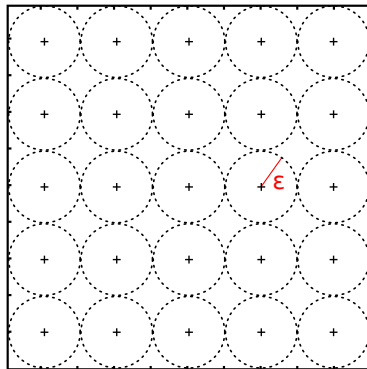
Proof of the theorem (2)

Definition

Consider R a set of points with lower bounded distance to each other:

- ▶ two distinct elements of R are at distance greater than 2ϵ from each other with

$$\epsilon = \frac{C(d)}{j^\alpha}$$



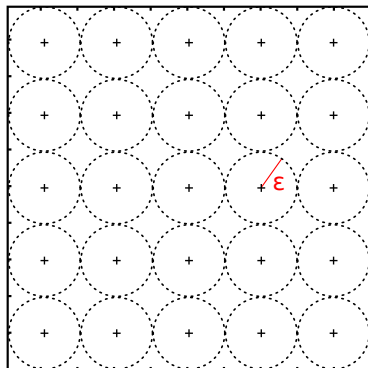
Proof of the theorem (3)

If x^* is uniformly drawn in $R...$

- ▶ *optimize* should have the opportunity to choose points which at distance $\leq \epsilon$ of (almost) each possible x^*

That is to say:

- ▶ $X_{n,\Omega}$ has points at distance $\leq \epsilon$ of a certain percentage $(1 - \delta/2)$ of elements in R



Proof of the theorem (4)

But...

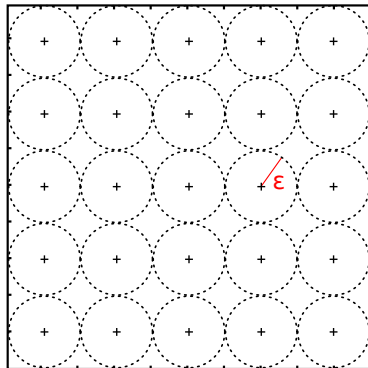
- ▶ if ϵ decreases (x_i approach x^*) then $|R|$ increases
- ▶ and $X_{n,\Omega}$ is finite (with great probability, by Lemma)

Up to a certain timestep, the locality assumption can't be respected.

We have a contradiction on our assumption.

$\alpha\beta > 1$ is wrong

QED



Stratified Sampling

Each ω_j is randomly drawn conditionally to a stratum.

The domain of ω is partitioned into disjoint strata S_1, \dots, S_N .

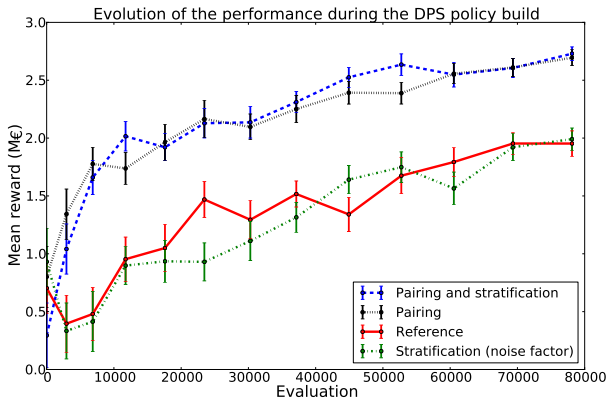
The stratification function $i \mapsto s(i)$ is chosen by the noisy optimization algorithm.

ω_j is randomly drawn conditionally to $\omega_j \in S_{s(i)}$.

$$\hat{\mathbb{E}}f(\mathbf{x}, \omega) = \sum_{i \in \{1, \dots, n\}} \frac{P(\omega \in s(i))f(\mathbf{x}, \omega_i)}{\text{Card} \{j \in \{1, \dots, n\}; \omega_j \in s(i)\}}$$



Experiments



Conclusions

- ▶ DVS: combining DPS and SDP for
 - ▶ fast decision making (polynomial)
 - ▶ anytime optimization
 - ▶ initialized at MPC (which is great)
 - ▶ no need for tree representation
- ▶ CRN:
 - ▶ no proof of good properties
 - ▶ but empirically really good





Future work

- ▶ Mathematical analysis of DVS
- ▶ More intensive testing of DVS
- ▶ Optimization at the level of investments
- ▶ Non-stochastic uncertainties




Bibliography

- ▶ *NEWAVE versus ODIN: comparison of stochastic and deterministic models for the long term hydropower scheduling of the interconnected brazilian system.* M. Zambelli et al., 2011.
- ▶ *Dynamic Programming and Suboptimal Control: A Survey from ADP to MPC.* D. Bertsekas, 2005. (MPC = deterministic forecasts)
- ▶ Astrom 1965
- ▶ *Renewable energy forecasts ought to be probabilistic!* P. Pinson, 2013 (WIPFOR talk)
- ▶ *Training a neural network with a financial criterion rather than a prediction criterion* Y. Bengio, 1997 (quite practical application of direct policy search, convincing experiments)

References I

-  S. Astete-Morales, J. Liu, and O. Teytaud, *log-log convergence for noisy optimization*, Proceedings of EA 2013, LLNCS, Springer, 2013, p. accepted.
-  alexander shapiro, *Analysis of stochastic dual dynamic programming method*, European Journal of Operational Research **209** (2011), no. 1, 63–72.
-  R. Bellman, *Dynamic programming*, Princeton Univ. Press, 1957.
-  Yoshua Bengio, *Using a financial training criterion rather than a prediction criterion*, CIRANO Working Papers 98s-21, CIRANO, 1998.

References III

-  Gérard Cornuéjols, *Revival of the gomory cuts in the 1990's*, Annals OR **149** (2007), no. 1, 63–66.
-  Z. L. Chen and W. B. Powell, *Convergent cutting-plane and partial-sampling algorithm for multistage stochastic linear programs with recourse*, J. Optim. Theory Appl. **102** (1999), no. 3, 497–524.
-  Christopher J. Donohue and John R. Birge, *The abridged nested decomposition method for multistage stochastic linear programs with relatively complete recourse*, Algorithmic Operations Research **1** (2006), no. 1.

References IV






Vaclav Fabian, *Stochastic approximation of minima with improved asymptotic speed*, Ann. Math. Statist. **38** (1967), no. 1, 191–200.







S. Hong, P.O. Malaterre, G. Belaud, and C. Dejean, *Optimization of irrigation scheduling for complex water distribution using mixed integer quadratic programming (MIQP)*, HIC 2012 – 10th International Conference on Hydroinformatics (Hamburg, Allemagne), IAHR, 2012, pp. p. – p. (Anglais).

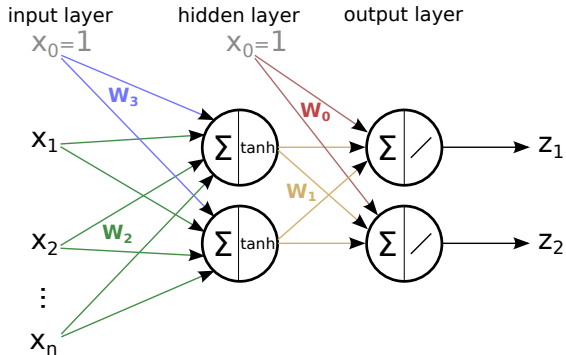
References V

-  Verena Heidrich-Meisner and Christian Igel, *Hoeffding and Bernstein races for selecting policies in evolutionary direct policy search*, ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning (New York, NY, USA), ACM, 2009, pp. 401–408.
-  Nikolaus Hansen, Sibylle D Müller, and Petros Koumoutsakos, *Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es)*, *Evolutionary Computation* **11** (2003), no. 1, 1–18.
-  Holger Heitsch and Werner Römisch, *Scenario tree reduction for multistage stochastic programs*, *Computational Management Science* **6** (2009), no. 2, 117–133.

References VII

-  W.-B. Powell, *Approximate dynamic programming*, Wiley, 2007.
-  M. V. F. Pereira and L. M. V. G. Pinto, *Multi-stage stochastic optimization applied to energy planning*, Math. Program. **52** (1991), no. 2, 359–375.
-  A. Ruszczyński, vol. 10, ch. Decomposition methods, North-Holland Publishing Company, Amsterdam, 2003.
-  S. Uryasev and R.T. Rockafellar, *Optimization of conditional value-at-risk*, Research report (University of Florida. Dept. of Industrial & Systems Engineering), Department of Industrial & Systems Engineering, University of Florida, 1999.

Parametric policies



$$z = \mathbf{W}_0 + \mathbf{W}_1 \tanh(\mathbf{W}_2 \mathbf{x} + \mathbf{W}_3)$$

Parametric policies

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}, \quad \mathbf{z} = \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_k \end{pmatrix}, \quad \mathbf{W}_3 = \begin{pmatrix} w_{10}^{(1)} \\ w_{20}^{(1)} \\ \vdots \\ w_{m0}^{(1)} \end{pmatrix}, \quad \mathbf{W}_0 = \begin{pmatrix} w_{10}^{(2)} \\ w_{20}^{(2)} \\ \vdots \\ w_{k0}^{(2)} \end{pmatrix},$$

$$\mathbf{W}_2 = \begin{pmatrix} w_{11}^{(1)} & w_{12}^{(1)} & \cdots & w_{1n}^{(1)} \\ w_{21}^{(1)} & w_{22}^{(1)} & \cdots & w_{2n}^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m1}^{(1)} & w_{m2}^{(1)} & \cdots & w_{mn}^{(1)} \end{pmatrix}, \quad \mathbf{W}_1 = \begin{pmatrix} w_{11}^{(2)} & w_{12}^{(2)} & \cdots & w_{1m}^{(2)} \\ w_{21}^{(2)} & w_{22}^{(2)} & \cdots & w_{2m}^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ w_{k1}^{(2)} & w_{k2}^{(2)} & \cdots & w_{km}^{(2)} \end{pmatrix}.$$

$$\mathbf{z} = \mathbf{W}_0 + \mathbf{W}_1 \tanh(\mathbf{W}_2 \mathbf{x} + \mathbf{W}_3)$$

Self-adaptive Evolution Strategy (SA-ES) with revaluations

Require:

$K > 0$, $\lambda > \mu > 0$, a dimension $d > 0$, τ (usually $\tau = \frac{1}{\sqrt{2d}}$).

Initialize parent population $P_\mu = \{(\mathbf{x}_1, \sigma_1), (\mathbf{x}_2, \sigma_2), \dots, (\mathbf{x}_\mu, \sigma_\mu)\}$
with $\forall i \in \{1, \dots, \mu\}$, $\mathbf{x}_i \in \mathbb{R}^d$ and $\sigma_i = 1$.

while stop condition do

Generate the offspring population $P_\lambda = \{(\mathbf{x}'_1, \sigma'_1), (\mathbf{x}'_2, \sigma'_2), \dots, (\mathbf{x}'_\lambda, \sigma'_\lambda)\}$
where each individual is generated by:

1. *Select* (randomly) ρ parents from P_μ .
2. *Recombine* the ρ selected parents to form a recombinant individual (\mathbf{x}', σ') .
3. *Mutate* the strategy parameter: $\sigma' \leftarrow \sigma' e^{\tau \mathcal{N}(0,1)}$.
4. *Mutate* the objective parameter: $\mathbf{x}' \leftarrow \mathbf{x}' + \sigma' \mathcal{N}(\mathbf{0}, \mathbf{1})$.

Select the new parent population P_μ taking the μ best form $P_\lambda \cup P_\mu$.

end while

Fabian

- 1: Input: an initial $x_1 = 0 \in \mathbb{R}^d$, $\frac{1}{2} > \gamma > 0$, $a > 0$, $c > 0$, $m \in \mathbb{N}$, weights $w_1 > \dots > w_m$ summing to 1, scales $1 \geq u_1 > \dots > u_m > 0$.
- 2: $n \leftarrow 1$
- 3: **while** (true) **do**
- 4: Compute $\sigma_n = c/n^\gamma$.
- 5: Evaluate the gradient g at x_n by finite differences, averaging over $2m$ samples per axis:

$$\forall i, j \in \{1, \dots, d\} \times \{1 \dots m\}, x_n^{(i,j)+} = x_n + u_j e_i,$$

$$\forall i, j \in \{1, \dots, d\} \times \{1 \dots m\}, x_n^{(i,j)-} = x_n - u_j e_i,$$

$$\forall i \in \{1, \dots, d\}, g^{(i)} = \frac{1}{2\sigma_n} \sum_{j=1}^m w_j \left(f(x_n^{(i,j)+}) - f(x_n^{(i,j)-}) \right).$$

- 6: Apply $x_{n+1} \leftarrow x_n - \frac{a}{n} g$
- 7: $n \leftarrow n + 1$
- 8: **end while**

Fabian's stochastic gradient algorithm with finite differences. Several variants have been defined, in particular versions in which only one point (or a constant number of points, independently of the dimension) is evaluated at each iteration.

Lemmas - Introduction (1)

The objective function

$$f_{\mathbf{x}^*, \beta, \gamma}(\mathbf{x}) = \mathcal{B} \left(\gamma \left(\frac{\|\mathbf{x} - \mathbf{x}^*\|}{\sqrt{d}} \right)^\beta + (1 - \gamma) \right)$$

and the *locality assumption*

$$\forall f \in F, \quad P \left(\forall i \leq n, \quad \underbrace{\|\mathbf{x}_i - \mathbf{x}^*\|}_{\leq \frac{C(d)}{i^\alpha}} \right) \geq 1 - \frac{\delta}{2}$$

imply

$$\underbrace{\mathbb{E}f(\mathbf{x}^*)}_{1-\gamma} \leq \mathbb{E}f(\mathbf{x}_n) \leq \underbrace{\mathbb{E}f(\mathbf{x}^*)}_{1-\gamma} + \frac{\gamma}{d^{\beta/2}} \frac{C(d)^\beta}{n^{\alpha\beta}}$$

with probability at least $1 - \delta/2$

(f and $\mathbb{E}f(\mathbf{x})$ are short notations for $f_{\mathbf{x}^*, \beta, \gamma}$ and $\mathbb{E}_\omega f(\mathbf{x}, \omega)$)

Lemmas - Introduction (2)

$$\begin{aligned}
\|\mathbf{x}_i - \mathbf{x}^*\| &\leq \frac{C(d)}{i^\alpha} \quad (\text{the locality assumption}) \\
\Leftrightarrow \underbrace{\gamma \left(\frac{\|\mathbf{x}_i - \mathbf{x}^*\|}{\sqrt{d}} \right)^\beta + (1 - \gamma)} &\leq \underbrace{(1 - \gamma)} + \gamma \left(\frac{C(d)}{i^\alpha \sqrt{d}} \right)^\beta \\
\Leftrightarrow \mathbb{E}f(\mathbf{x}_i) &\leq \mathbb{E}f(\mathbf{x}^*) + \gamma \left(\frac{C(d)}{i^\alpha \sqrt{d}} \right)^\beta \\
\Leftrightarrow \mathbb{E}f(\mathbf{x}_i) &\leq \mathbb{E}f(\mathbf{x}^*) + \frac{\gamma C(d)^\beta}{i^{\alpha\beta} \sqrt{d}^\beta}
\end{aligned}$$

Proof

Consider $f_{\mathbf{x}^*} = f_{\mathbf{x}^*, \beta, \gamma}$ with \mathbf{x}^* uniformly distributed in R . Then:

$$\begin{aligned}
 & P(\|\mathbf{x}_n - \mathbf{x}^*\| \leq \epsilon) \\
 & \leq \mathbb{E}_{\Omega} P_{\mathbf{x}^*}(\mathbf{x}^* \in \text{Enl}(X_{n, \Omega}, \epsilon)) \\
 & \leq P(\#X_{n, \Omega} \leq C) P_{\mathbf{x}^*}(\mathbf{x}^* \in \text{Enl}(X_{n, \Omega}, \epsilon) | \#X_{n, \Omega} \leq C) \\
 & \quad + P(\#X_{n, \Omega} > C) \\
 & \leq \left(1 - \frac{\delta}{2}\right) \frac{C}{C'} + \frac{\delta}{2} \\
 & < 1 - \frac{\delta}{2}
 \end{aligned}$$

where $\text{Enl}(U, \epsilon)$ is the ϵ -enlargement of U defined as:

$$\text{Enl}(U, \epsilon) = \{\mathbf{x}; \exists \mathbf{x}' \in U, \|\mathbf{x} - \mathbf{x}'\| \leq \epsilon\}.$$

This contradicts the locality assumption.

This concludes the proof of $\alpha\beta \leq 1$.

R-EDA (1)

Algorithm 2 R-EDA: algorithm for optimizing noisy fitness functions. *Bernstein* denotes a Bernstein race, as defined in Algorithm 3. The initial domain is $[x_0^-, x_0^+] \in \mathbb{R}^D$, δ is the confidence parameter. This algorithm goes back to [15, 16].

```

n ← 0
while True do
  // Pick the coordinate with highest uncertainty
  c_n = arg max_i (x_n^+)_i - (x_n^-)_i
  δ_n^max = (x_n^+)_{c_n} - (x_n^-)_{c_n}
  for i ∈ [[1, 3]] do
    // Consider the middle point
    x'_n{}^i ← 1/2(x_n^- + x_n^+)
    // The c_n^{th} coordinate may take 3 ≠ values
    (x'_n{}^i)_{c_n} ← (x_n^-)_{c_n} + (i-1)/2(x_n^+ - x_n^-)_{c_n}
  end for
  (good_n, bad_n) = Bernstein(x'_n{}^1, x'_n{}^2, x'_n{}^3, 6δ / (π^2(n+1)^2)).
  // A good and a bad point
  Let H_n be the halfspace
  {x ∈ ℝ^D; ||x - good_n|| ≤ ||x - bad_n||}
  Split the domain: [x_{n+1}^-, x_{n+1}^+] = H_n ∩ [x_n^-, x_n^+]
  n ← n + 1
end while

```

R-EDA (2)

Algorithm 3 Bernstein race between 3 points. Eq. 3 is Bernstein's inequality to compute the precision for empirical estimates (see e.g. [18, p124]); $\hat{\sigma}_i$ is the empirical estimate of the standard deviation of point x_i 's associated random variable $F_t(x_i)$ (it is 0 in the first iteration, which does not alter the algorithm's correctness); $\hat{f}(x)$ is the average of the fitness measurements at x .

Bernstein(x_1, x_2, x_3, δ')

$T = 0$

repeat

$T \leftarrow T + 1$

Evaluate the fitness of points x_1, x_2, x_3 once, *i.e.* evaluate the noisy fitness at each of these points.

Evaluate the precision:

$$\epsilon_{(T)} = 3 \log \left(\frac{3\pi^2 T^2}{6\delta'} \right) / T + \max_i \hat{\sigma}_i \sqrt{2 \log \left(\frac{3\pi^2 T^2}{6\delta'} \right) / T}. \quad (3)$$

until Two points (*good*, *bad*) satisfy $\hat{f}(\text{bad}) - \hat{f}(\text{good}) \geq 2\epsilon$
 — **return** (*good*, *bad*)

R-EDA (3)

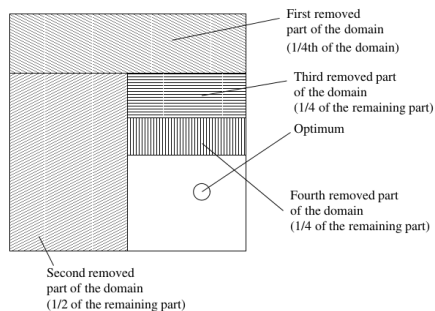


Fig. 1. Noisy optimization algorithm (cf Algorithm 2). At each iteration, a main axis is selected (the one on which the domain has maximum range). Three equally spaced points are generated in the domain on this axis (this is the offspring). Then, a Bernstein race is applied for choosing a “good” and a “bad” arm among these points. The domain is reduced thanks to this knowledge, removing one fourth or one half of the domain (depending on the position of the good arm and of the bad arm - the best case is when the good and the bad arm are diametrically opposed: see Fig. 2).

Theorem's proof

Step 2

$$(??) \iff \ln \left(\frac{f(\mathbf{x})}{f(\mathbf{x}')} \right) \geq \max \left(\ln \left(\frac{1}{\eta} \right), \ln \left(\frac{K'}{K''} \right) + \ln \left(\frac{\|\mathbf{x}\|}{\|\mathbf{x}'\|} \right) \right)$$

$$f(\mathbf{x}') \leq \eta f(\mathbf{x}) \iff \frac{1}{\eta} f(\mathbf{x}') \leq f(\mathbf{x})$$

$$\iff \frac{1}{\eta} \leq \frac{f(\mathbf{x})}{f(\mathbf{x}')}$$

$$\iff \ln \left(\frac{f(\mathbf{x})}{f(\mathbf{x}')} \right) \geq \ln \left(\frac{1}{\eta} \right)$$

$$f(\mathbf{x}') \leq \frac{K''}{K'} \frac{\|\mathbf{x}'\|}{\|\mathbf{x}\|} f(\mathbf{x}) \iff f(\mathbf{x}') \frac{K'}{K''} \frac{\|\mathbf{x}\|}{\|\mathbf{x}'\|} \leq f(\mathbf{x})$$

$$\iff \frac{K'}{K''} \frac{\|\mathbf{x}\|}{\|\mathbf{x}'\|} \leq \frac{f(\mathbf{x})}{f(\mathbf{x}')}$$

$$\iff \ln \left(\frac{K'}{K''} \frac{\|\mathbf{x}\|}{\|\mathbf{x}'\|} \right) \leq \ln \left(\frac{f(\mathbf{x})}{f(\mathbf{x}')} \right)$$

$$\iff \ln \left(\frac{f(\mathbf{x})}{f(\mathbf{x}')} \right) \geq \ln \left(\frac{K'}{K''} \right) + \ln \left(\frac{\|\mathbf{x}\|}{\|\mathbf{x}'\|} \right)$$

Theorem's proof

Step 2

if $l' \geq \ln(c')$ then

- ▶ $l < \ln(c)$ (otherwise it couldn't be a "win")

$$\begin{aligned}
 l' - \ln(c') &\leq \ln(c + \max_i \|\delta_i\|) - \ln(c') \\
 &\leq \ln(c) + \ln(1 + \max_i \|\delta_i\|/c) - \ln(c') \\
 &\leq \ln(c/c') + \max_i \|\delta_i\|/c \\
 &\leq \max_i \|\delta_i\|/c
 \end{aligned}$$

$$\begin{aligned}
 l' - \ln(c') = \ln\left(\frac{\|x + \sigma\delta_i\|}{\sigma}\right) - \ln(c') &\leq \ln\left(\frac{\|x\|}{\sigma} + \delta_i\right) - \ln(c') \\
 &\leq \ln\left(c * \left(1 + \frac{\delta_i}{c}\right)\right) - \ln(c') \\
 &\leq \ln(c) + \ln\left(1 + \max_i \|\delta_i\|/c\right) - \ln(c') \\
 &\leq \ln(c/c') + \max_i \|\delta_i\|/c \\
 &\leq \max_i \|\delta_i\|/c
 \end{aligned}$$

Theorem's Proof

Step 3

Summing iterations

Hence if $t > n_0$,

$$\ln(f(\mathcal{X}_t)) \leq \ln(f(\mathcal{X}_1)) - (t - n_0) \times \sum_i \frac{\max\left(\ln\left(\frac{1}{\eta}\right), \ln\left(\frac{K'}{K''}\right) + \Delta\right)}{\min\left(1 + \ln\left(\frac{c}{b}\right) \frac{\Delta}{\ln(2)}, 1 + \ln\left(\frac{c}{b}\right) \frac{\max_j \ln(\|\delta_j\|)}{\ln(2)}\right)}$$

$$\iff \ln(f(\mathcal{X}_t)) - \ln(f(\mathcal{X}_1)) \leq -(t - n_0) \times C$$

$$\iff \frac{\ln\left(\frac{f(\mathcal{X}_t)}{f(\mathcal{X}_1)}\right)}{t - n_0} \leq -C$$

$$\Rightarrow \frac{\ln(\|\mathcal{X}_t\|)}{t} \leq K < 0 \quad (\text{theorem})$$

with C a positive constant.

Corollary's proof

Step 2 (1/4)

Step 2: showing that σ small leads to high acceptance rate and σ high leads to small acceptance rate.

Thanks to the bounded conditioning, there exists $\epsilon > 0$ s.t.

$$s' < \frac{1}{2}s \tag{1}$$

$$\text{with } s = \sup \left\{ \frac{\sigma}{\|\mathbf{x}\|}; \sigma, \mathbf{x}, f \text{ s.t. } p \geq \frac{\epsilon}{2} \right\}$$

$$\text{and } s' = \inf \left\{ \frac{\sigma}{\|\mathbf{x}\|}; \sigma, \mathbf{x}, f \text{ s.t. } p < \frac{1}{2} - \frac{\epsilon}{2} \right\}$$

because $s' \rightarrow 0$ and $s \rightarrow \infty$ as $\epsilon \rightarrow 0$.

Corollary's proof

Step 2 (2/4)

Notes

$$\hat{s} = \sup \left\{ \frac{\sigma}{\|\mathbf{x}\|}; \sigma, \mathbf{x}, f \text{ s.t. } \hat{p} \geq \epsilon \right\}$$

$$\hat{s}' = \inf \left\{ \frac{\sigma}{\|\mathbf{x}\|}; \sigma, \mathbf{x}, f \text{ s.t. } \hat{p} < \frac{1}{2} - \epsilon \right\}$$

Then

$$\sup_{\mathbf{x}, f, \sigma > 0} |\hat{p} - p| \leq \epsilon/2$$

implies

$$\frac{1}{2}\hat{s} \geq \frac{1}{2}s \quad \text{and} \quad s' \geq \hat{s}'$$

Corollary's proof

Step 2 (3/4)

So

$$\hat{s}' \leq s' < \frac{1}{2}s \leq \frac{1}{2}\hat{s}$$

and

$$\hat{s}' \leq \frac{1}{2}\hat{s}$$

Corollary's proof

Step 2 (4/4)

This provides k_1 , k_2 , c' and b' as follows for Eqs. ?? and ??:

$$\begin{aligned} \frac{1}{b'} &= \hat{s} = \sup \left\{ \frac{\sigma}{\|\mathbf{x}\|}; \sigma, \mathbf{x}, f \text{ s.t. } \hat{p} \geq \epsilon \right\} \\ \frac{1}{c'} &= \hat{s}' = \inf \left\{ \frac{\sigma}{\|\mathbf{x}\|}; \sigma, \mathbf{x}, f \text{ s.t. } \hat{p} < \frac{1}{2} - \epsilon \right\} \\ k_1 &= \lfloor \epsilon k \rfloor \\ k_2 &= \left\lceil \left(\frac{1}{2} - \epsilon \right) k \right\rceil \end{aligned}$$

Eqs. above imply $c' \geq 2b'$

Corollary's proof

Step 3

k large enough yield

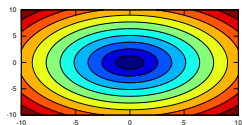
$$b^{-1} = \sup \left\{ \frac{\sigma}{\|\mathbf{x}\|}; \sigma, \mathbf{x}, f \text{ s.t. } \hat{p} > k_1/k \right\},$$

$$c^{-1} = \inf \left\{ \frac{\sigma}{\|\mathbf{x}\|}; \sigma, \mathbf{x}, f \text{ s.t. } \hat{p} < k_2/k \right\},$$

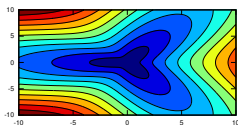
which provide assumptions 2 and 5 with $b < c$

Assumptions 2 and 5 then imply $b < b'$ and $c' < c$

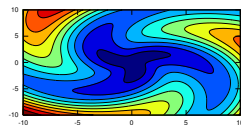
Non quasi-convex functions



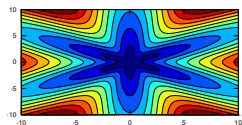
(a)



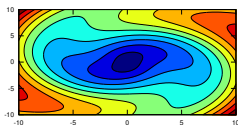
(b)



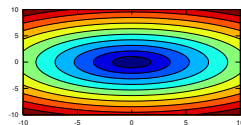
(c)



(d)



(e)



(f)