Power systems and uncertainties: revisited through games

Intro: uncertainties in PS

Power systems.

Bias, bias correction: by resampling.

Non-stochastic uncertainties: Nash method.

If time enough, portfolio methods (?).

ADEME

Agence de l'Environnement et de la Maîtrise de l'Energie

informatiques mathématiques

Power systems : key issues

- **Real time:** Network topology optimization
- **Short term:** When to use to hydroelectricity ?
- **Long term :**
  - Should we build big connections France-XXX ?
  - Should we build big wind farms ?

==> with X billions, what should I do ?

Goal: build tools ***and*** answer these questions

Power systems : old and new

- Old: production → transmission → distribution
- New: all mixed + more uncertainties

+ renewable

+ storage

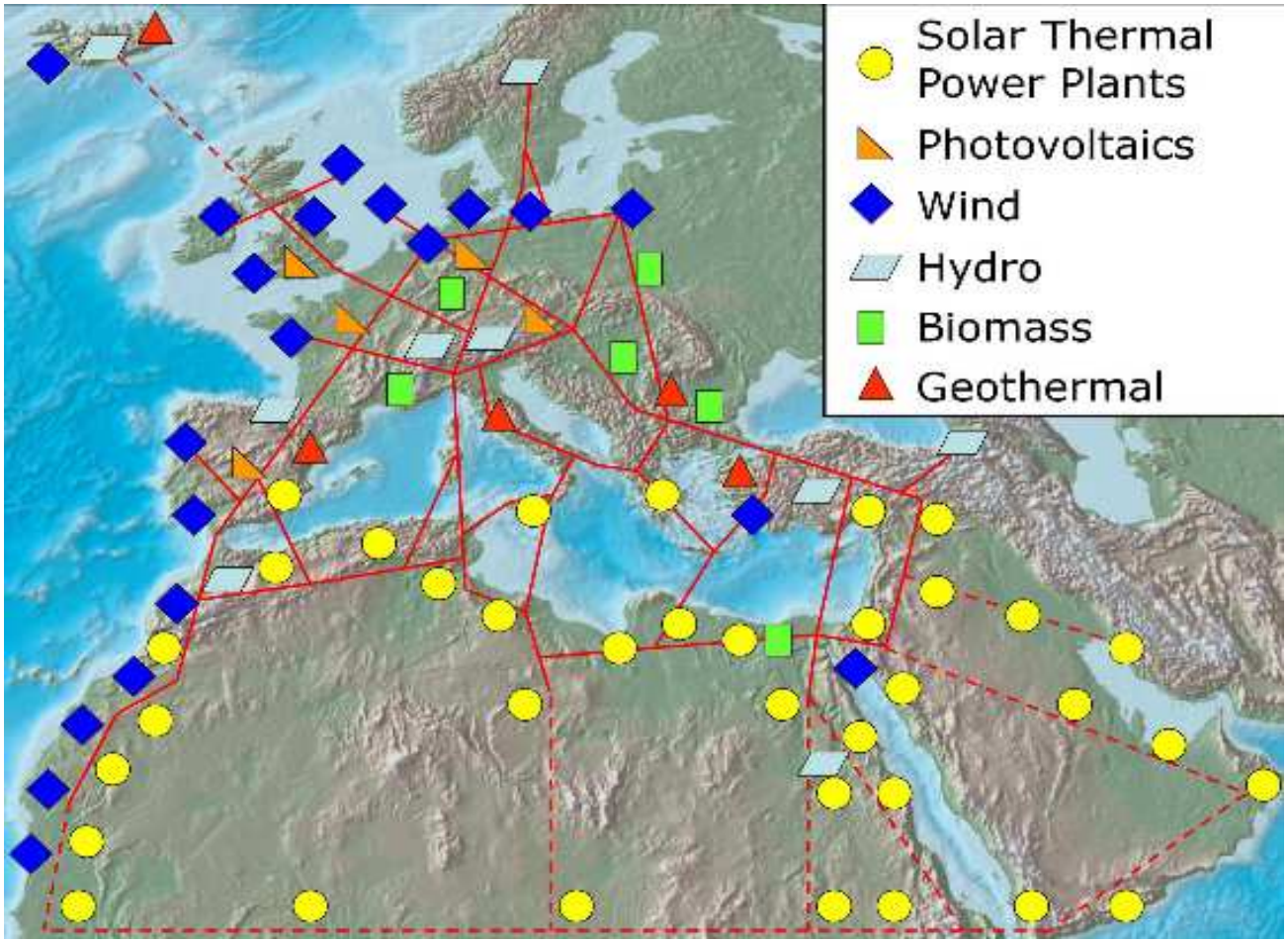
- CO^2 ?

+ distributed

+ demand-side

+ HVDC

? ? ?



Power systems

Power systems

reward

.

.

Power systems

$\mathbb{E}$ **reward**

. weather

. (+ load, faults, ...)

Power systems

argmax $\mathbb{E}$ **reward**

. policy weather

.

Power systems

$\mathbb{U}$ **argmax** $\mathbb{E}$ **reward**

techn. policy weather

geopol.

Power systems

Argmax $\mathbb{U}$ **argmax** $\mathbb{E}$ **reward**

investment techn. policy weather

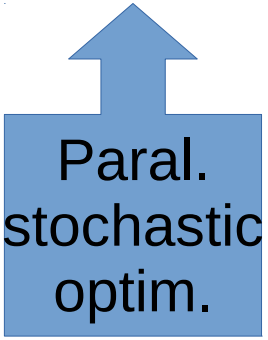
geopol.

Power systems

Argmax $\mathbb{U}$ **argmax** $\mathbb{E}$ **reward**

investment techn. policy weather

geopol.

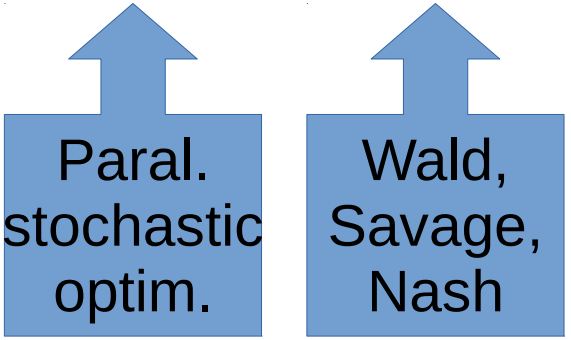


Power systems

Argmax $\mathbb{U}$ **argmax** $\mathbb{E}$ **reward**

investment techn. policy weather

geopol.

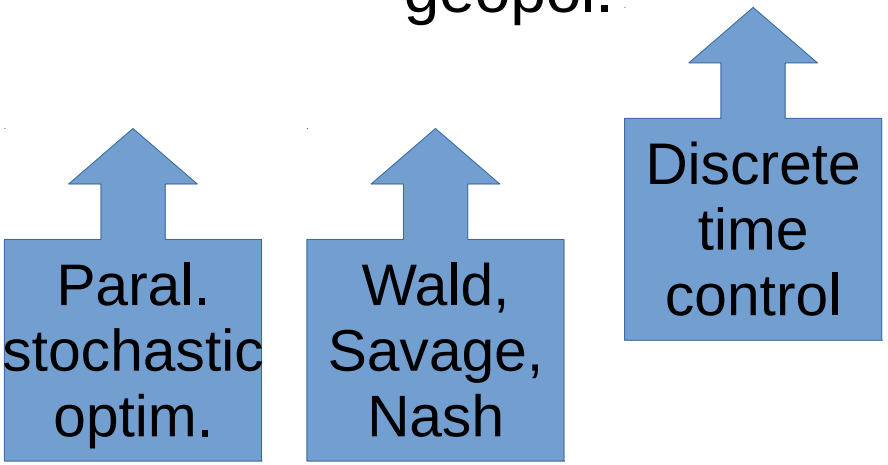


Power systems

Argmax $\mathbb{U}$ **argmax** $\mathbb{E}$ **reward**

investment techn. policy weather

geopol.

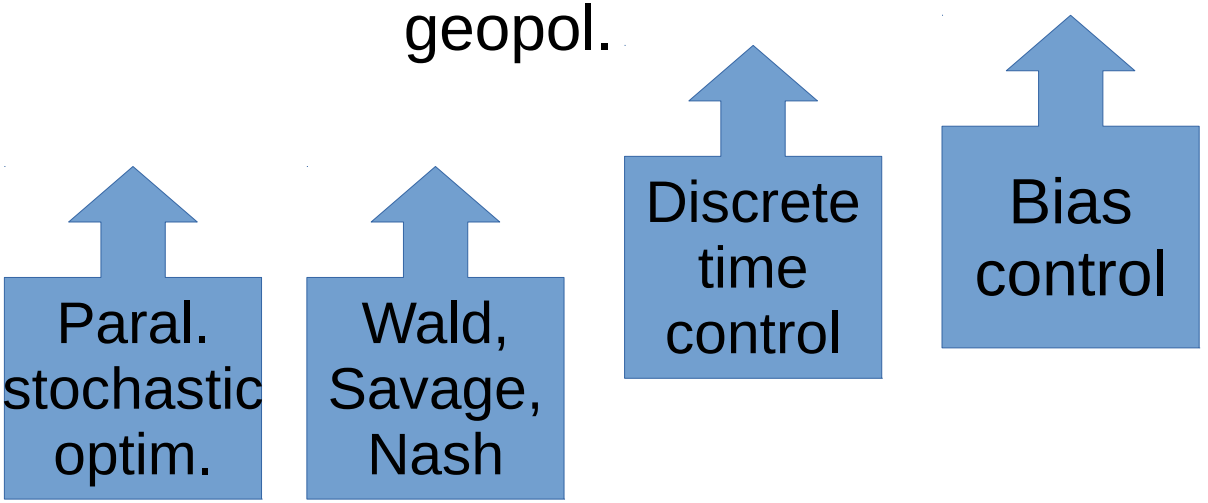


Power systems

Argmax $\mathbb{U}$ **argmax** $\mathbb{E}$ **reward**

investment techn. policy weather

geopol.

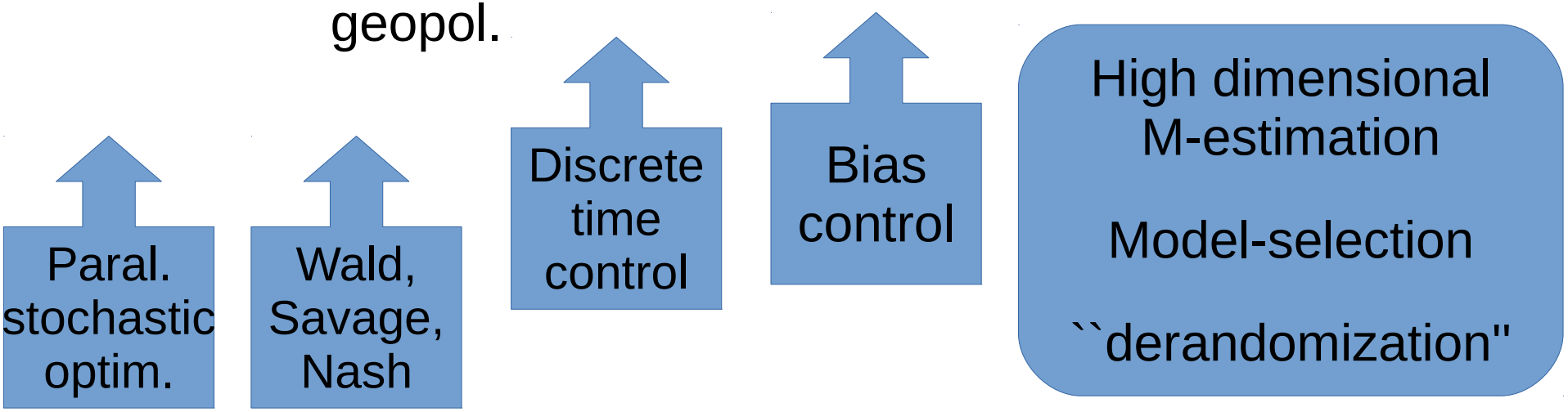


Power systems

Argmax $\mathbb{U}$ argmax $\mathbb{E}$ reward

investment techn. policy weather

geopol.

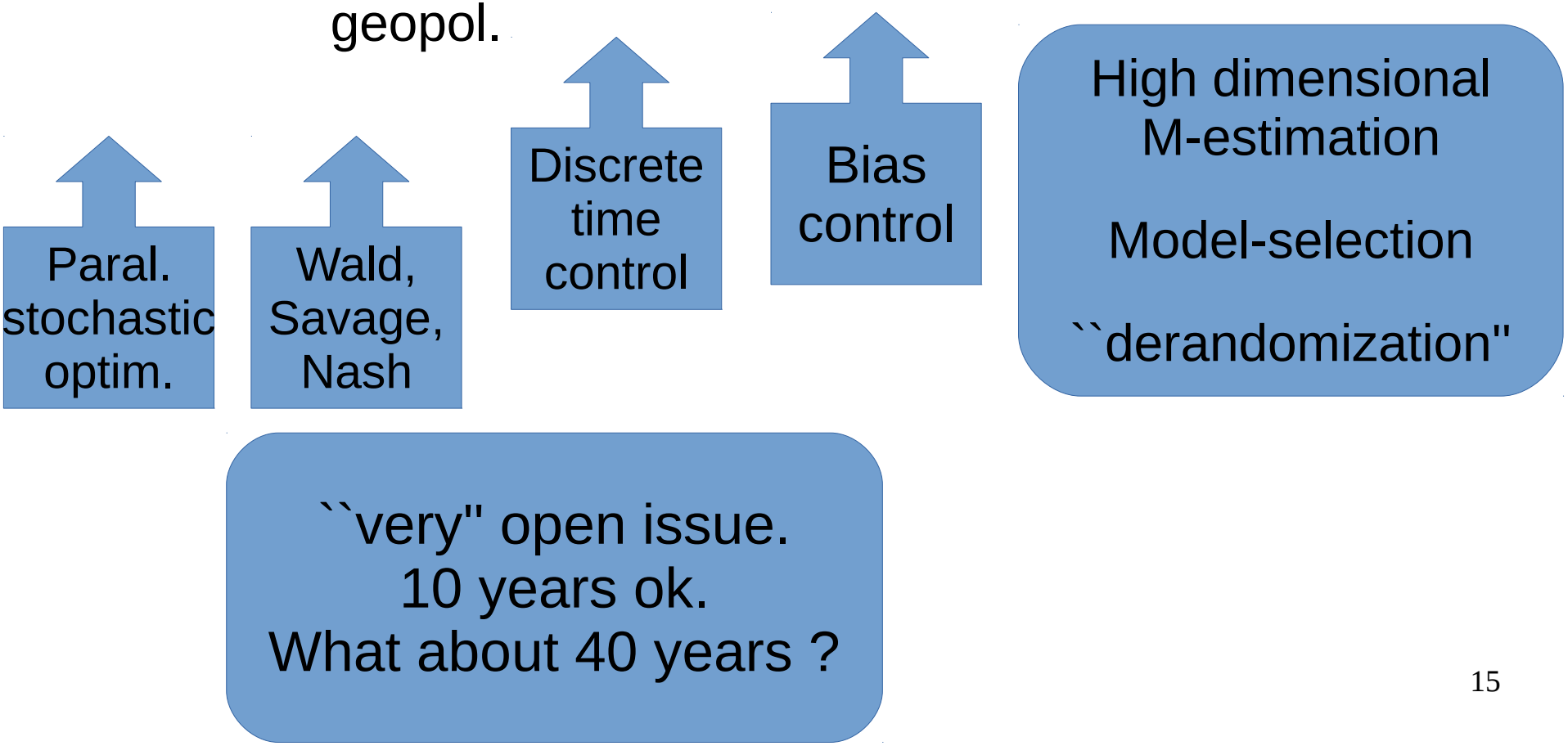


Power systems

Argmax $\mathbb{U}$ argmax $\mathbb{E}$ reward

investment techn. policy weather

geopol.



Various uncertainties

- Data unavailable
- Stochastic
  - Weather <== average on data is (≈) ok
  - Faults <== average on data is not ok (otherwise many lines have fault probability zero)
- Non stochastic (truth hidden or partly adversarial)
  - Technology
  - Gas curtailment
  - CO2 penalization

Power systems and uncertainties

Bias correction

When average on data; uncertainty = coverage

Non stochastic uncertainties

Oh my god how to deal with that ?

Noisy optimization

When we have a model of stoc. uncertainties

ADEME

Agence de l'Environnement et de la Maîtrise de l'Energie

informatiques mathématiques

Bias correction

Maximum height, from a finite sample:

The maximum height of 13 randomly drawn humans

is less than the maximum height of all humans.

Bias correction, more surprising

95% quantile of height, from a finite sample:

The .95 quantile of height of 13 randomly drawn humans

is less than the .95 quantile of all humans.

```
R=[];for i=1:1000;x=randn(10000,1);q=empirical_inv(.95,x);aq=empirical_inv(.95,x(1:100));R=[R;q,aq];end;mean(R(:,1)>R(:,2))
```

Bias correction, capacity

- Optimal capacity ?

  Optimal capacities = argmin E cost(x,random)

  approx. capacities = argmin (cost(x,r1) + … + cost(x,rn))

- Also biased! Usually, approx. capacities << opt. capacities.

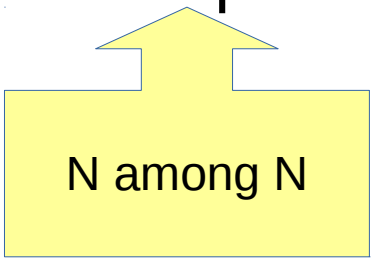
- The optimal capacity on average over a sample is less than the “real” optimal capacity.

  Surprising ?

Bias correction by bootstrap, ideas

Definitions:

- x0 = Real optimum, against the real (unknown) distribution
- x1 = Optimum against the sample
- x2 = Optimum against a resample of the sample



Idea:

- x2 is to x1, what x1 is to x0. (Efron)
- So E(x2-x1) = E(x1-x0) (approx)
- So Ex0 = 2Ex1 - Ex2 (approx)
- So let us use 2x1 – Ex2 (expectation over resamplings)

Bias correction by bootstrap

Standard approximate $\hat{x}(S) = \hat{x}$ minimizes:

$$x \mapsto \hat{\mathbb{E}}_{s \in S} f(s, x) = \frac{1}{n} \sum_{i=1}^{n} f(s_i, x)$$

Bootstrap correction:

$$\hat{x}_{bs} = 2\hat{x} - \hat{\mathbb{E}}_{N,\hat{S}} \hat{\hat{x}} = 2\hat{x} - \left[\frac{1}{N} \sum_{j=1}^{N} \hat{x}(\hat{S}_j) \right]$$

Classical approximation (twice!)

Bootstrap Version.

Bias correction by Jackknife

(often better, for bias correction)

Standard approximate $\hat{x}(S) = \hat{x}$ minimizes:

$$x \mapsto \hat{\mathbb{E}}_{s \in S} f(s, x) = \frac{1}{n} \sum_{i=1}^{n} f(s_i, x)$$

Jackknife correction:

$$\hat{x}_{jk} = n\hat{x} - (n-1)\hat{\mathbb{E}}_{N,\hat{S}}\hat{\hat{x}} = n\hat{x} - (n-1)\left[\frac{1}{N}\sum_{j=1}^{N}\hat{x}(\hat{S}_j)\right]$$

Optimum on
all but one

Cross validation

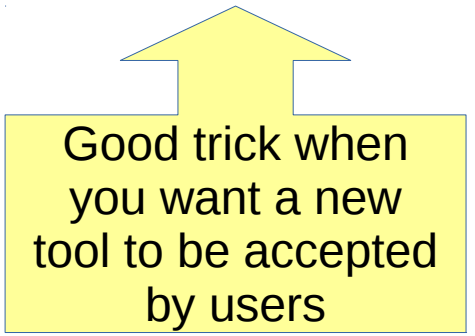
- For example, 10 scenarios (possible winters)
- You have an algorithm for choosing the optimal capacity.
- How to check that it works ?
- Bad solution: apply your algorithm on the 10 scenarios.
- Better solution: apply your algorithm on 9 scenarios, check on the 10th

(and then rotate)

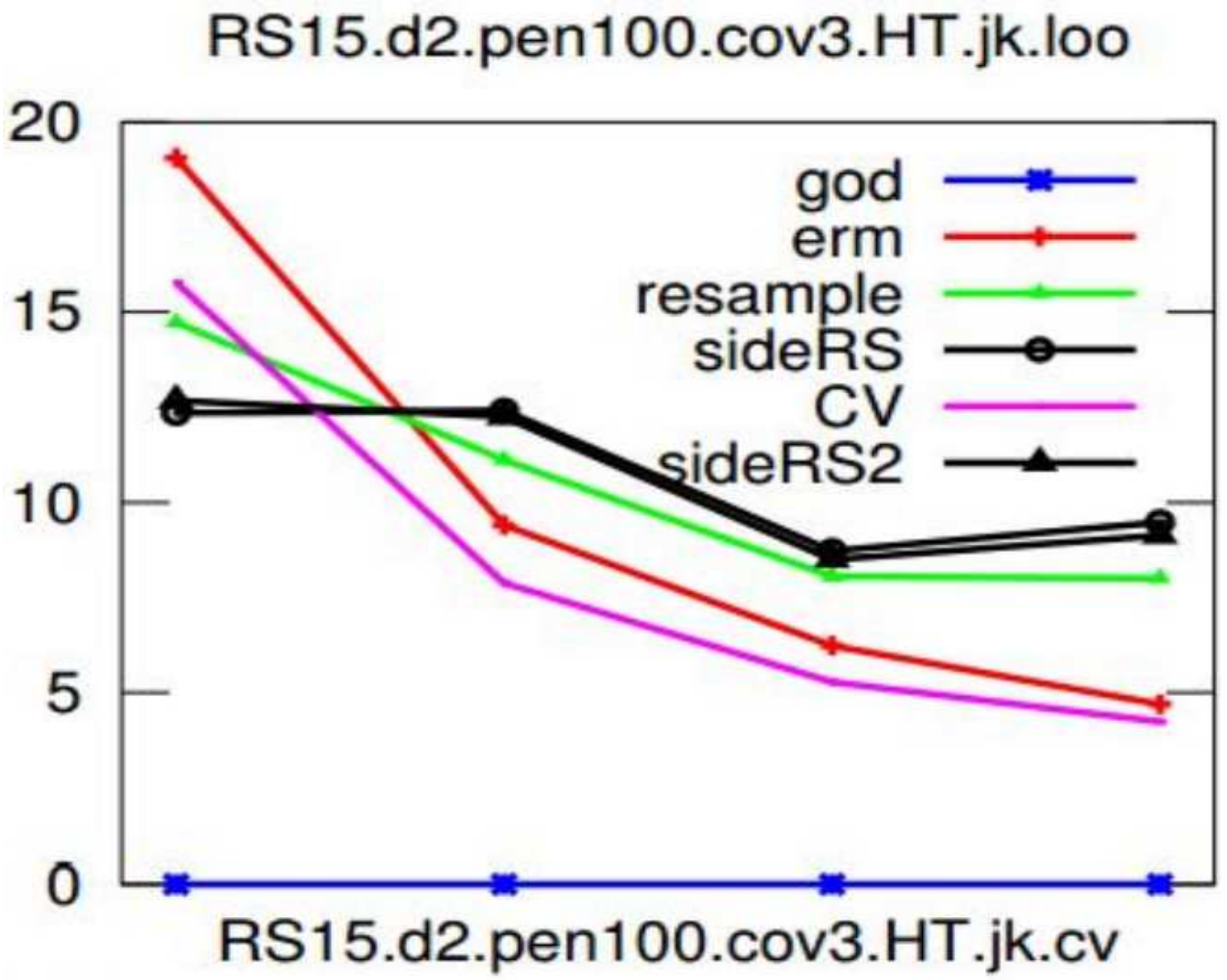
==> 100% mandatory in machine learning research
==> not widely used in capacity expansion planning

Bias correction: many tricks

- Apply both the classical estimate, the new one, variants of the new one, and choose between them by cross-validation <== excellent, but expensive
- Average the bias correction over multiple homogeneous capacities
- Prefer the correction only if crossval predicts a clear success (at least 10%) <== safe



Bias correction



Bias & bias correction

Main take-home messages:

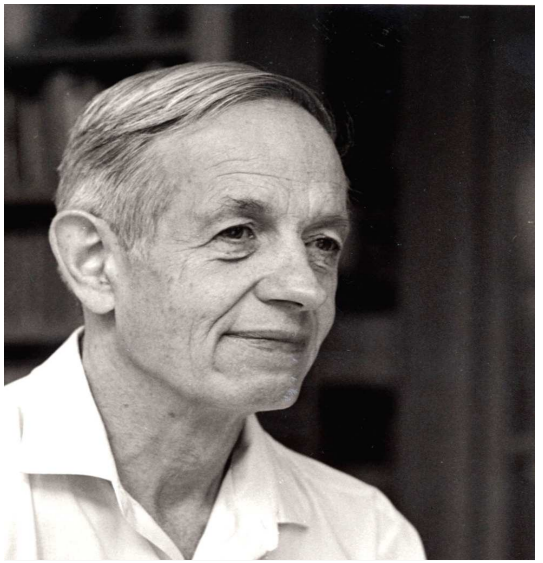
- The cost of stochastic systems is usually underestimated (bias 1)
- Optimal capacities are usually underestimated (bias 2)

A simple alternate method:
check by cross-validation if
+1%, +2%, +4%, +8%, +16%
are better than +0%

Power systems and uncertainties

Non stochastic uncertainties:

Wald, Savage and Nash

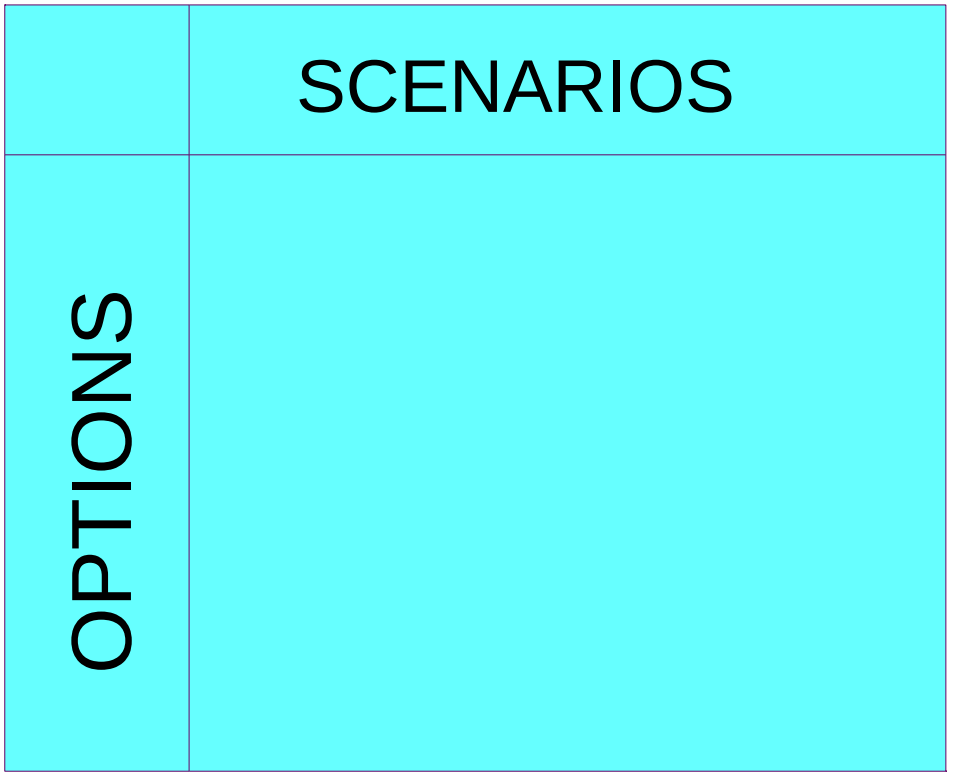


ADEME

Agence de l'Environnement
et de la Maîtrise de l'Energie

informatiques mathématiques

The problem: non probabilistic uncertainties



We have plenty of options, there are plenty of possible scenarios.

No probability + computing one entry takes time.

What should we decide ?

Non stochastic uncertainties

- Uncertainties:
  - Stochastic: weather, load
  - Not stochastic: CO2 penalization, gas curtailment... (typically, scenario)
- Model: R(k,s) = reward if option k,
- How to deal with it ?
  - Scenario based: human checks all R(k,s), make a decision :-)
  - Wald: choose k maximizing min R(k,.) <== worst case
  - Savage:
    - R'(k,s)= max R(.,s) – R(k,s)
    - k minimizing max R'(k,.) <== best “regret”

In both cases, this is optimal
when nature decides **after** us
and knows what we have chosen:
- Wald: Nature knows and hates us
- Savage: Nature wants us to have regrets

Doesn't it look like a game ?

	Rock / paper/scissor		
Rock/paper/scissor	0	1	-1
	-1	0	1
	1	-1	0

We are ok for “Nature” to be adversarial, but not an opponent who knows our decision ==>

Nature decides first, privately.

Non stochastic uncertainties
a (hard to read) proposal

Nash:

s (probabilistic) such that max E R(.,s) minimal

eq. k (probabilistic!) such that min E R(k,.) maximal

==> worst case, for a Nature who does not know our decision

==> slightly philosophical, do you prefer Wald, Savage, or Nash, or a expert decision ?

Non stochastic uncertainties clearer proposal (hopefully)

Given:

- A list of “s” scenarios (possibly huge)
- A list of “k” options (possibly huge)
- A function R(.,.)

Comp. Time ~ m log(m)

where m= #K + #S

Output:

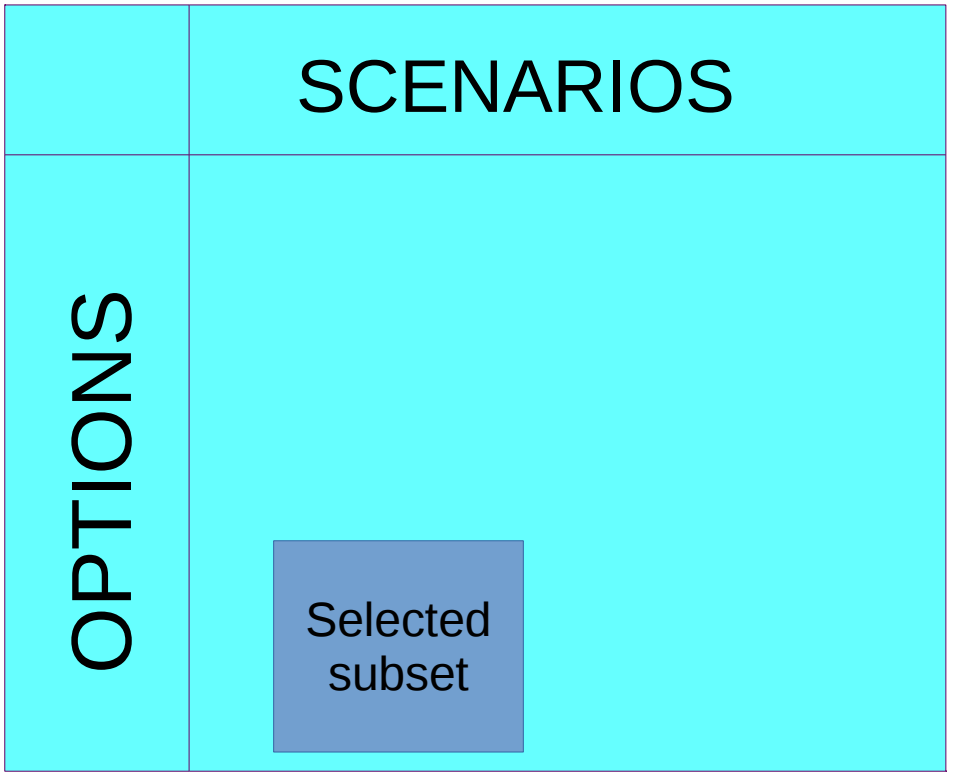
- a sublist of scenarios (approx. Nash optimal),
- a sublist of options (approx. Nash optimal),
- probabilities on them

==> cheaper than Wald / Savage, and does selection

K policies, S scenarios (no probabilities), how to make a decision ?

METHOD	EXTRACTION OF POLICIES	EXTRACTION OF CRITICAL SCENARIOS	COMPUTATIONAL COST	INTERPRETATION
Wald	One	One per policy	$K \times S$	Nature decides later, minimizing our reward.
Savage	One	One per policy	$K \times S$	Nature decides later, maximizing our regret.
Nash	Nash-optimal	Nash-optimal	$(K + S) \times \log(K + S)$	Nature decides privately, before us.
Scenarios	Handcrafted	Handcrafted	$K' \times S'$	Human expertise

Short version



Intuitively cool: we naturally select a subset of scenarios / options, and Nature decides privately, before us.

Practically cool: fast.

Problem: output = stochastic decision (you decide nuclear power plants with a dice ?)

Power systems and uncertainties

WHEN UNCERTAINTIES ARE A STOCHASTIC MODEL

Our experience:

- noisy optimization is a nightmare
- hard to find the best algorithm
- don't choose: combine ==>

  portfolio

ADEME

Agence de l'Environnement et de la Maîtrise de l'Energie

informatiques mathématiques

Noisy optimization

- Two families of optimization principles
  - Optimizing in front of a (corrected) archive of possibilities
  - Optimizing in front of a stochastic model

- Pros/cons:
  - Archive = real data
  - Model = can deal with extreme events and new scenarios <== faults in networks (the death of the N-1 criterion ?)

  ==> this part, **robust** noisy optimization,

  thanks to portfolio methods

Portfolio methods in noisy optimization

I. Introduction: portfolios

II.Portfolio methods

III. Portfolio NOT in noisy optimization

IV. Portfolio methods in noisy optimization

Who is the best ? ==> portfolio methods (portfolio = pissing contest…)

	Alg. 1	Alg. 2	Alg. 3	Alg. 4
Games	MC (Monte Carlo)	MCTS	Alphabeta	DPS (direct policy search)
Control	PID	MCTS	Fuzzy	Bellman-style
Supervised ML on R^d	NeuralNet	SVM	Fuzzy	Decision trees
	Linear reg.	Gen. linear regression	...	
Feature selection	Feature ranking	Embedded approaches	Optimization	
Bandits	UCB variants	Uniform	SR (successive reject)	

FPS

==> The best algorithm is not always the same, by far.
How to design robust algorithms ?

Who is the best ?

(pissing contest...)

	Alg. 1	Alg. 2	Alg. 3	Alg. 4
Games	MC (Monte Carlo)	MCTS	Alphabeta	DPS (direct policy search)
Control	PID	MCTS	Fuzzy	Bellman-style
Supervised ML on R^d	NeuralNet	SVM	Fuzzy	Decision trees
	Linear reg.	Gen. linear regression	...	
Feature selection	Feature ranking	Embedded approaches	Optimization	
Bandits	UCB variants	Uniform	SR (successive reject)	

Chess

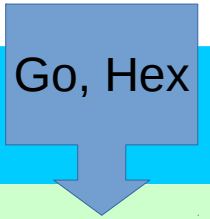
==> The best algorithm is not always the same, by far.

How to design robust algorithms ?

Who is the best ?

(pissing contest...)

	Alg. 1	Alg. 2	Alg. 3	Alg. 4
Games	MC (Monte Carlo)	MCTS	Alphabeta	DPS (direct policy search)
Control	PID	MCTS	Fuzzy	Bellman-style
Supervised ML on R^d	NeuralNet	SVM	Fuzzy	Decision trees
	Linear reg.	Gen. linear regression	...	
Feature selection	Feature ranking	Embedded approaches	Optimization	
Bandits	UCB variants	Uniform	SR (successive reject)	



==> The best algorithm is not always the same, by far.

How to design robust algorithms ?

Who is the best ?

(pissing contest...)

Fog of war

	Alg. 1	Alg. 2	Alg. 3	Alg. 4
Games	MC (Monte Carlo)	MCTS	Alphabeta	DPS (direct policy search)
Control	PID	MCTS	Fuzzy	Bellman-style
Supervised ML on R^d	NeuralNet	SVM	Fuzzy	Decision trees
	Linear reg.	Gen. linear regression	...	
Feature selection	Feature ranking	Embedded approaches	Optimization	
Bandits	UCB variants	Uniform	SR (successive reject)	

==> The best algorithm is not always the same, by far.

How to design robust algorithms ?

Who is the best ?

(pissing contest...)

	Alg. 1	Alg. 2	Alg. 3	Alg. 4
Games	MC (Monte Carlo)	MCTS	Alphabeta	DPS (direct policy search)
Control	PID	MCT	Fuzzy	Bellman-style
Supervised ML on R^d	NeuralNet	SVM	Fuzzy	Decision trees
	Linear reg.	Gen. linear regression	...	
Feature selection	Feature ranking	Embedded approaches	Optimization	
Bandits	UCB variants	Uniform	SR (successive reject)	

For publishing

==> The best algorithm is not always the same, by far.

How to design robust algorithms ?

Who is the best ?

(pissing contest...)

	Alg. 1	Alg. 2	Alg. 3	Alg. 4
Games	MC (Monte Carlo)	MCTS	Alphabeta	DPS (direct policy search)
Control	PID	MCTS	Fuzzy	Bellman-style
Supervised ML on R^d	NeuralNet	SVM	Fuzzy	Decision trees
	Linear reg.	Gen. linear regression	...	
Feature selection	Feature ranking	Embedded approaches	Optimization	
Bandits	UCB variants	Uniform	SR (successive reject)	

When it must work

==> The best algorithm is not always the same, by far.

How to design robust algorithms ?

The design of robust algorithms

- Do not run only one algorithm.
- Run several algorithms, concurrently.

The design of robust algorithms

- Do not run only one algorithm.
- Run several algorithms, concurrently.
- Pick up the best response.

The design of robust algorithms

- Do not run only one algorithm.
- Run several algorithms, concurrently.
- Pick up the best response.

if for all problems, out of 11 algorithms:

ten of the algorithms need >= 40 hours to complete,

but one algorithm completes in 10 minutes (not always the same)

The design of robust algorithms

- Do not run only one algorithm.
- Run several algorithms, concurrently.
- Pick up the best response.

if for all problems, out of 11 algorithms:

ten of the algorithms need >= 40 hours to complete,

but one algorithm completes in 10 minutes (not always the same)

Then:

- Average time =110 minutes (10 minutes, multiplied by 11 for concur.)

  + no variance

The design of robust algorithms

- Do not run only one algorithm.
- Run several algorithms, concurrently.
- Pick up the best response.

if for all problems, out of 11 algorithms:

ten of the algorithms need >= 40 hours to complete,

but one algorithm completes in 10 minutes (not always the same)

Then:

- Average time =110 minutes (10 minutes, multiplied by 11 for concur.)

  + no variance
- Whereas with sequential runs a.t. = 200 hours+10minutes + variance

Portfolio methods: you already do it

Usually, we compare algorithms offline in large datasets.

Possible improvements:

- online comparison ?
- problem-specific comparison ?
- hybridization ?
- different budget / algo ?

Portfolio methods: you already do it

Usually, we compare algorithms offline in large datasets.

Possible improvements:

- online comparison ?
- problem-specific comparison ?
- hybridization ?
- more budget for better solvers

==> this is portfolio

Portfolio methods

I. Introduction: ML and portfolios

II.Portfolio methods: basics

III. Portfolio NOT in noisy optimization

IV. Portfolio methods in noisy optimization

Glossary (1)

- Offline: we decide the best in advance (from related datasets)
- Online: dynamically

Glossary (1)

- Offline: we decide the best in advance (from related datasets)
- Online: dynamically

- Fair: all algs use the same budget
- Unfair: more for the (seemingly) best

Glossary (2)

- Sharing: algorithms can provide information to each other

Glossary (2)

- Sharing: algorithms can provide information to each other

- Chaining: current best predictor / controller is used by all algorithms (implies that all algorithms have related representations)

Glossary (2)

- Sharing: algorithms can provide information to each other
- Chaining: current best predictor / controller is used by all algorithms (implies that all algorithms have related representations)
- Parallel: each solver on (at least) one core

Glossary (2)

- Sharing: algorithms can provide information to each other
- Chaining: current best predictor / controller is used by all algorithms (implies that all algorithms have related representations)
- Parallel: each solver on (at least) one core
- Anytime algorithm: performs well in spite of not knowing its computation time in advance

Competence map

- Competence map = mapping

  (problem features PF, algorithm features AF)
  ==> CM(PF,AF)=predicted performance

- Choose algorithm = argmax CM(PF,.)

Portfolio methods

I. Introduction: ML and portfolios

II.Portfolio methods

III. Portfolio NOT in noisy optimization 

IV. Portfolio methods in noisy optimization

Most classical portfolio: combinatorial optimization

- Why combinatorial optimization ?
  - Huge difference between performances of algorithms (prob. dep.)
  - Checking who is the best is easy

Most classical portfolio: combinatorial optimization

- Why combinatorial optimization ?
  - Huge difference between performances of algorithms (prob. dep.)
  - Checking who is the best is easy

- Results:
  - Stable, robust, competitive even against specialized tools in each category
  - Routinely wins SAT competitions (cf http://www.cs.ubc.ca/labs/beta/Projects/SATzilla/)

==> *something has moved in AI :-)*

(not only deep networks :-))

1. What does the state of the art tells us in portfolio ?

No free lunch [Wolpert & many]: all algorithms perform equally on average on all optimization problems ==> so we can not find “one” optimal algorithm.

Does this justify portfolio ?

- Yes, there is no optimal algorithm (so ==> portfolio...).

2. What does the state of the art tells us in portfolio ?

No free lunch [Wolpert & many]: all algorithms perform equally on average on all optimization problems ==> so we can not find “one” optimal algorithm.

Does this justify portfolio ?

- Yes, there is no optimal algorithm.
- No: even the portfolio is not better :-)

3. What does the state of the art tells us in portfolio ?

Samulowitz and Memisevic:

<< use orthogonal solvers in your portfolio >>

i.e. try not to have similar solvers in your portfolio

3. What does the state of the art tells us in portfolio ?

Samulowitz and Memisevic:

<< use orthogonal solvers in your portfolio >>

i.e. try not to have similar solvers in your portfolio ==> we will see a proof of this later.

4. What does the state of the art tells us in portfolio ?

What if all solvers are indeed one solver with various parametrizations ?

Then you have a *structure*, a *metric*, on your solvers ==> this is parameter tuning ==> bilevel optimization rather than really “portfolio”.

==> Here, unstructured family of solvers.

==> See “portfolio / parameter tuning” in [Kotthoff 2012]

5. What does the state of the art tells us in portfolio ?

Unfair distribution of budget:

- 50% for best, 25% for second, 12.5% for third, etc. [Gagliolo, Schmidthuber '05'06]

5. What does the state of the art tells us in portfolio ?

Unfair distribution of budget:

- 50% for best, 25% for second, 12.5% for third, etc. [Gagliolo, Schmidthuber '05'06]
- Finite time with fair budget, then only the best. [Pulina, Tacchella '09 + detailed comparison]

5. What does the state of the art tells us in portfolio ?

Unfair distribution of budget:

- 50% for best, 25% for second, 12.5% for third, etc. [Gagliolo, Schmidthuber '05'06]
- Finite time with fair budget, then only the best. [Pulina, Tacchella '09 + detailed comparison]
- 90% for offline best, 10% for all others (just in case!) [Kadioglu et al, '11]

5. What does the state of the art tells us in portfolio ?

Unfair distribution of budget:

- 50% for best, 25% for second, 12.5% for third, etc. [Gagliolo, Schmidthuber '05'06]
- Finite time with fair budget, then only the best. [Pulina, Tacchella '09 + detailed comparison]
- 90% for offline best, 10% for all others (just in case!) [Kadioglu et al, '11]
- Best first, even if fair (if not parallel).

6. What does the state of the art tells us in portfolio ?

Parallelism is great & easy!

Many experiments in [Hamadi 2013].

7. What does the state of the art tells us in portfolio ?

Chaining & sharing: tricky but can work... we did not succeed much with that in ML :-(

[see Vassilevska et al. 06]

Portfolio methods

I. Introduction: ML and portfolios

II.Portfolio methods

III. Portfolio NOT in ML (optim, bio)

IV. Portfolio methods in noisy optimization 

Portfolio in noisy optimization

Key difference: checking performance takes time & is uncertain:

- hold out + test (supervised learning on dataset)
- or random generation of new test cases (control)

Comparing two candidates might take more time than generating them!

Waow! How comparing can take more time than solving ? (Black-box case)

Example: parametric controller

- Black-box f(x,w): test my controller
  - if I have parameter x;
  - if random processes = w
  - and returns the performance
- Randomized black-box f(x):
  - if I have parameter x
  - with w randomly drawn
  - and returns the performance

Black box noisy optimization:

- I request f(x1),f(x2),...,f(xn)
- I propose x
- The real optimum is x*
- My regret is Rn=f(x*)-f(x)

In many cases (papers by Fabian, Chen, Shamir) I get
E Rn=O(1/n)
i.e. budget = 1/precision

whereas comparing xA and xB:

- Ef(xA) estimated std O(1/sqrt(n))
- Ef(xB) estimated std O(1/sqrt(n))

==> budget = 1/precision ! ! !

Why ? Should be easier!
No, because in the first case we sample “far” from x* and “learn” the shape of E f

Waow! How comparing can take more time than solving ? (White-box case)

Example: parametric controller

- White-box f(x,w): test my controller
  - if I have parameter x;
  - if random processes = w
  - and returns the performance
    +the gradient
- Randomized black-box f(x):
  - if I have parameter x
  - with w randomly drawn
  - and returns the performance
    +the gradient

Black box noisy optimization:

- I request f(x1),f(x2),...,f(xn) + grads
- I propose x
- The real optimum is x*
- My regret is Rn=f(x*)-f(x)

==> similar rates!

whereas comparing xA and xB:

- Ef(xA) estimated std O(1/sqrt(n))
- Ef(xB) estimated std O(1/sqrt(n))
- ==> budget = 1/precision ! ! !

==> still true!

Key point

- There are many problems for which the natural criterion is noisy and leads to rates of the form regret = $O(1/n^{\alpha})$
- And the comparison is $O(1/n^{\beta})$

==> Noisy optimization,

with or without gradients;

or just with comparisons;

==> or many maximum likelihood / ERM / SRM rates

ok, not always, but often :-)

So let us consider this case.

s*n*, r*n* and lag

Many solutions for writing portfolios.

In all cases, you have roughly these parameters:

s*n*, r*n* and lag

Many solutions for writing portfolios.

In all cases, you have roughly these parameters:

- How frequently you compare

s*n*, r*n* and lag

Many solutions for writing portfolios.

In all cases, you have roughly these parameters:

- How frequently you compare
- How precisely you compare (adaptive ? Maybe but no “race” *(or you might spend a huge time)*)

s(*n*), r(*n*) and lag(u)

Many solutions for writing portfolios.

In all cases, you have roughly these parameters:

- How frequently you compare
  (r(n) = #evals at n^{th} comparison)
- How precisely you compare
  (s(n) = budget for n^{th} comparison)
- More surprising, who you compare: *old* outputs
  (lag(u) << u is the index of compared outputs)

Because old outputs are cheaper to compare!

Drawback: fair budget. ==> Lazy evaluation: do evaluations only when they are needed

- for comparison (cheap because lag(u)<<u)
- or for the output (cheap because only a few algos – the optimal ones)

Almost readable theorem

If solver i has regret $(C_{(i)}+o(1)) / n^{\alpha(i)}$, and

$$r_n = \lceil n^{3+r+r'} \rceil;$$
$$\mathrm{LAG}(n) = \lceil \log(n) \rceil;$$
$$s_n = \lceil n^{1+r'} \rceil, r > 0 \text{ and } r' \geq 1, n \in \mathbb{N}^*.$$

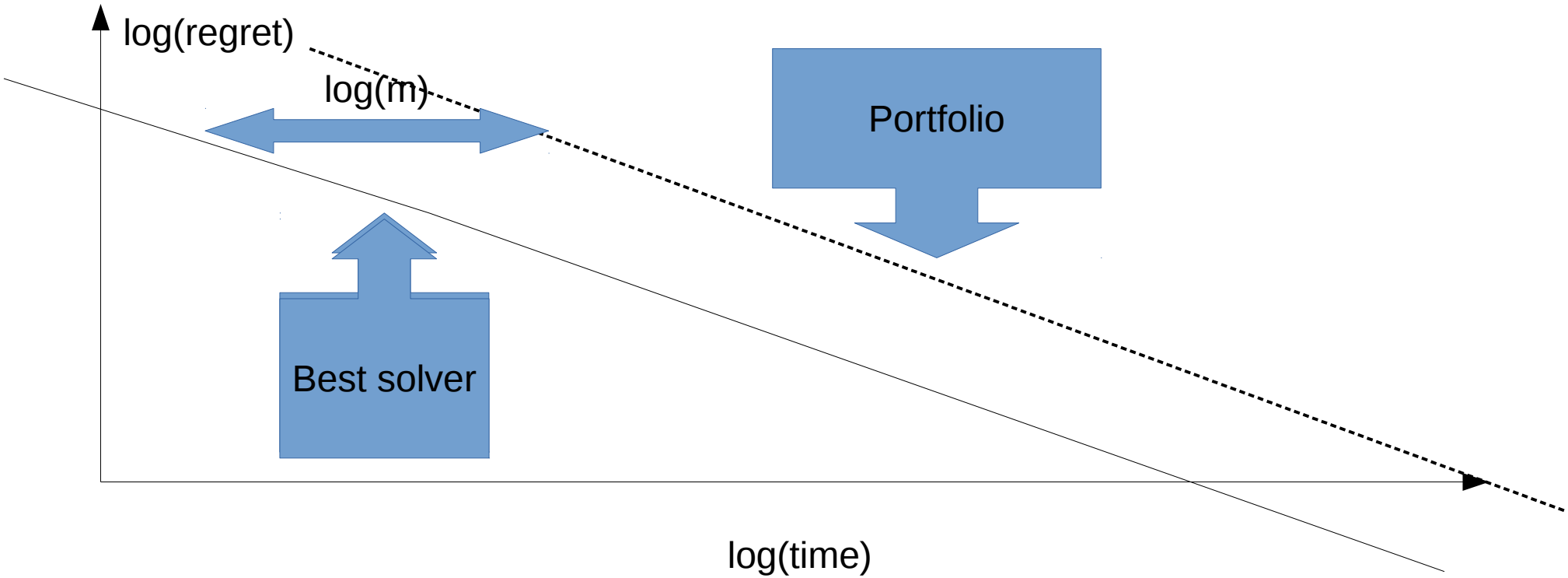
then,

- NOPA (noisy opt. portfolio alg.) has the log(m) shift ;
- and INOPA (improved Nopa) has the log(m') shift.

Readable version: NOPA

There exist universal parameters, such that

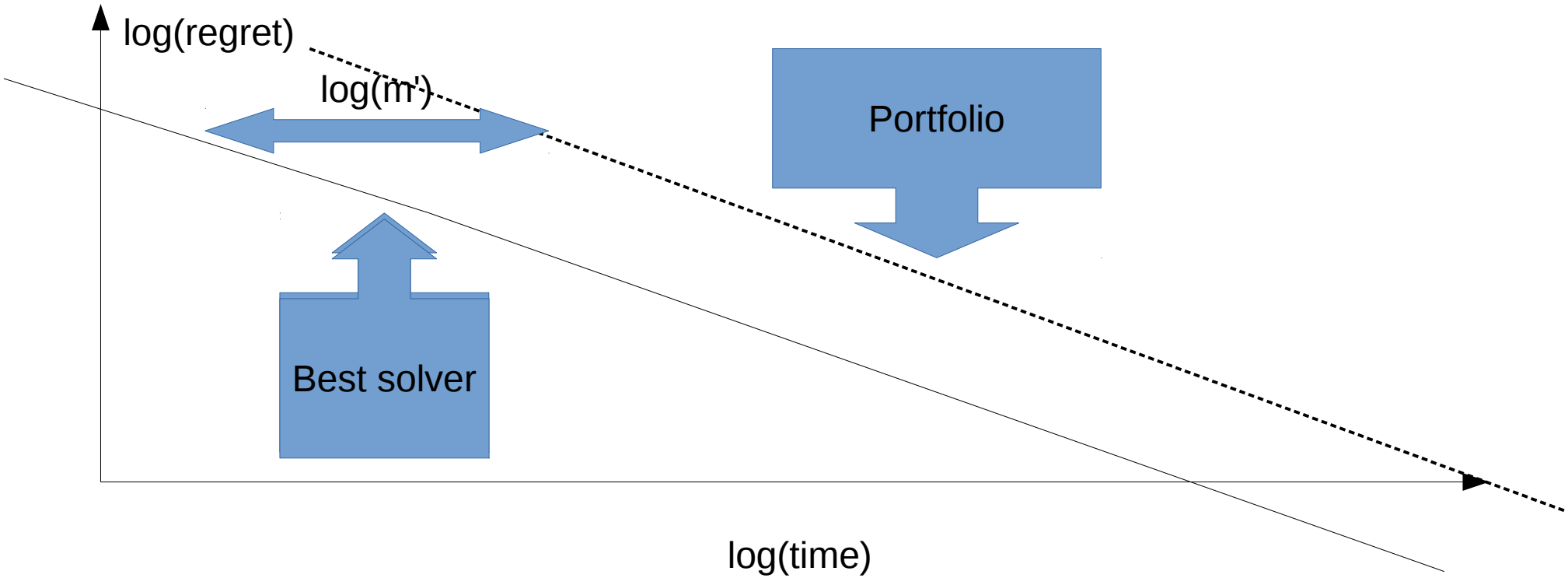
- if rates are $f(x_n)-f(x^*) \sim 1/n^{\alpha}$
- then NOPA just has a log(m) shift *(m=#solvers)*



Readable version: INOPA

There exist universal parameters, such that

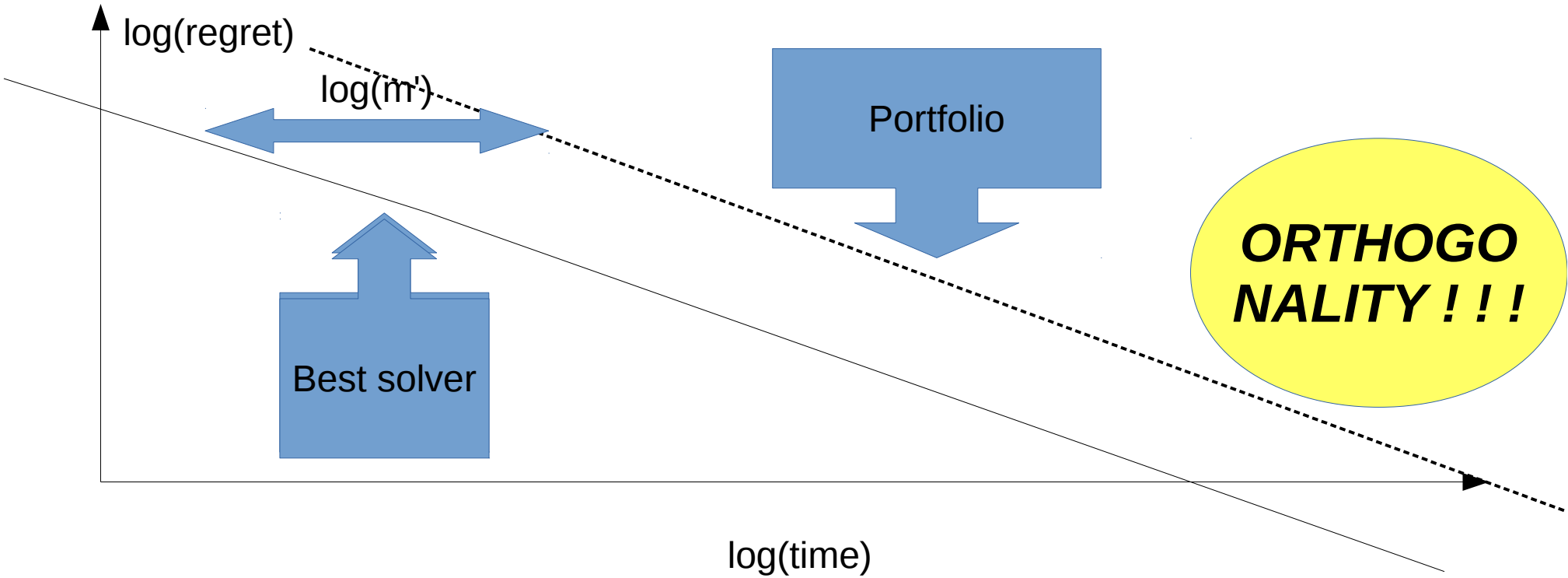
- if rates are $f(x_n)-f(x^*) \sim 1/n^{\alpha}$
- then INOPA has a log(m') shift *(m'=#optimal solvers)*



Readable version: INOPA

There exist universal parameters, such that

- if rates are $f(x_n)-f(x^*) \sim 1/n^{\alpha}$
- then INOPA has a log(m') shift *(m'=#optimal solvers)*



Conclusions

- Portfolio = revolution in comb. optim.
  - Simple
  - Compliant with parallelism
  - Robust (even against bug!)
  - Winning many competitions
- In noisy optimization
  - Can be more tricky, because comparing is expensive
  - But great for robustness, ML is so variable!

Power systems and uncertainties

Bias correction: a different path to robustness; easy to apply, but expensive (parallel); at least, check your bias (cross-validation)

Scenarios: Nash = more complicated, but natural and selecting scenarios. But random recommendation :-)

Noisy optimization: a new application for portfolio methods.

Energy transition debate

MEDIAPART
MAR.17 NOVEMBRE 2015 – ÉDITION DE LA MI-JOURNÉE
LE JOURNAL | INTERNATIONAL | FRANCE | ÉCONOMIE | CULTURE | *ENGLISH* | *ESPAÑOL*

CLIMAT : LE SOMMET PARIS 2015 — ENQUÊTE

Energie : le rapport caché sur une France 100% renouvelable

08 AVRIL 2015 | PAR CHRISTOPHE GUEUGNEAU ET JADE LINDGAARD

Mediapart s'est procuré le rapport commandé par l'Ademe sur une France 100 % renouvelable en 2050 : le potentiel énergétique est colossal, et ne coûterait pas beaucoup plus cher que de maintenir le nucléaire. Mais visiblement, il dérange puisque sa publication a été repoussée. Nous le publions en intégralité pour

Energy transition debate

Nucléaire : **EDF pourrait associer des partenaires au renouvellement du parc français**

ANNE FEITZ / JOURNALISTE | LE 23/10 À 17:27, MIS À JOUR À 17:53

Le PDG d'EDF, Jean-Bernard Levy /Credit:NICOLAS MESSYASZ/SIPA - **SIPA**

1 / 1

A l'horizon 2050, un parc de 30 ou 40 EPR Nouveau Modèle aura été construit, imagine EDF. L'électricien estime qu'il

ÉNERGIE - ENVIRONNEMENT

Le gaz et les carburants vont payer pour le solaire

Electricité : Bruxelles surveille les projets de la France

L'exécutif baisse la facture des industriels électro-intensifs

Power systems / energy transition

- Public debate is very dirty
- Claims based on proprietary codes/studies (or no study at all...)
- It's so easy to cheat and get the numbers you want

==> ***duty:*** prefer open source / open data