Stratified Nested Regression Monte-Carlo scheme with large scale parallelization

emmanuel.gobet@polytechnique.edu

With the support of La FONDATION du RISQUE

Joint work with J. Salas (U. da Coruña), P. Turkedjiev (EP), C. Vasquez (UdC).

In minor revision for *SIAM Scientific Computing.*

Structure of the talk

1. BSDE setting (could be extended to other dynamic programming)
2. Usual Regression Monte Carlo methods **[G'-Turkedjiev, Math Comp 2015]**
   - ✓ Algorithm (**parallelization not available**)
   - ✓ Error estimates
   - ✓ Strongest implementation constraint: **memory !!**
3. Stratified version, **parallelization on basis functions not on simulations**
   - ✓ Randomization and norms equivalence
   - ✓ Error estimates
   - ✓ Complexity and memory analysis
4. Numerical tests on GPU
5. Data driven version with non-intrusive stratified resampler (with Liu-Zubelli)

1) BSDE setting

Standard BSDE with *fixed terminal time* T:

$$\mathbf{Y_t} = \xi + \int_{\mathbf{t}}^{\mathbf{T}} \mathbf{f}(\mathbf{s}, \mathbf{Y_s}, \mathbf{Z_s})\mathrm{d}\mathbf{s} - \int_{\mathbf{t}}^{\mathbf{T}} \mathbf{Z_s}\mathrm{d}\mathbf{W_s}$$

- ✓ driving noise = Brownian Motion W
- ✓ Lipschitz driver f, terminal condition $\xi \in L_2$
- ✓ Markovian BSDE: $f(s, \omega, y, z) = f(s, X_s, y, z)$ and $\xi = g(X_T)$ for a diffusion X with coefficients (b, σ)
- ✓ Reaction-diffusion equations, neuroscience, non-linear pricing in finance

Multidimensional unknown: $X \in \mathbb{R}^d$, $Y \in \mathbb{R}$, $Z \in \mathbb{R}^q$.

Markovian BSDE: $\mathbf{Y_t} = \mathbf{u}(\mathbf{t}, \mathbf{X_t}), \mathbf{Z_t} = \sigma \nabla \mathbf{u}(\mathbf{t}, \mathbf{X_t}), \partial_{\mathbf{t}} \mathbf{u} + \mathcal{L}\mathbf{u} + \mathbf{f}(\mathbf{u}, \sigma \nabla \mathbf{u}) = \mathbf{0}$

Approximation/simulation in 2 stages:

1. time-discretization (numerous works under rather general settings)
2. solving the dynamic programming equation (nested cond. expect., few works)

Time discretization of $Y_t = \xi + \int_t^T f(s, Y_s, Z_s)\mathrm{d}s - \int_t^T Z_s \mathrm{d}W_s$

Discretization along *equidistant* time grid $\pi := \{0 = t_0 < \ldots < t_N = T\}$:

- ✓ $(i+1)$-th time-step is $\Delta_i = t_{i+1} - t_i = T/N$;
- ✓ related Brownian motion increments $\Delta W_i := W_{t_{i+1}} - W_{t_i}$.

Heuristic derivation

From $Y_{t_i} = Y_{t_{i+1}} + \int_{t_i}^{t_{i+1}} f(s, X_s, Y_s, Z_s)\mathrm{d}s - \int_{t_i}^{t_{i+1}} Z_s \mathrm{d}W_s$, we derive

$$\begin{aligned}
\mathbf{Y_{t_i}} &= \mathbb{E}(Y_{t_{i+1}} + \int_{t_i}^{t_{i+1}} f(s, X_s, Y_s, Z_s)\mathrm{d}s|\mathcal{F}_{t_i}) \\
&\approx \mathbb{E}(\mathbf{Y_{t_{i+1}}} + \mathbf{f(t_i, X_{t_i}, Y_{t_{i+1}}, Z_{t_i})\ \Delta_i}|\mathcal{F}_{\mathbf{t_i}}),
\end{aligned}$$

$$\begin{aligned}
\mathbf{Z_{t_i}\Delta_i} &\approx \mathbb{E}(\int_{t_i}^{t_{i+1}} Z_s \mathrm{d}s|\mathcal{F}_{t_i}) = \mathbb{E}([Y_{t_{i+1}} + \int_{t_i}^{t_{i+1}} f(s, X_s, Y_s, Z_s)\mathrm{d}s]\Delta W_i^\top|\mathcal{F}_{t_i}) \\
&\approx \mathbb{E}(\mathbf{Y_{t_{i+1}}\Delta W_i^\top}|\mathcal{F}_{\mathbf{t_i}}) \qquad \text{(where } {}^\top \text{ denotes the transpose).}
\end{aligned}$$

Dynamic programming equations

★ **O**ne-step forward **D**ynamic **P**rogramming equation

$$\begin{cases} Y_i & = \mathbb{E}_i\left(Y_{i+1} + f_i(Y_{i+1}, Z_i)\Delta_i\right), \quad 0 \le i < N, \qquad Y_N = \xi. \\ \Delta_i Z_i & = \mathbb{E}_i\left(Y_{i+1}\Delta W_i^\top\right), \quad 0 \le i < N. \end{cases} \qquad (\textbf{ODP})$$

✓ X could be approximated by a path-wise approximation (e.g. Euler scheme)

✓ For f and g Lipschitz, the L_2-error is of order $N^{-\frac{1}{2}}$

★ **M**ulti-Step forward **D**ynamic **P**rogramming equation:

$$\begin{cases} Y_i & = \mathbb{E}_i\left(\xi + \sum_{k=i}^{N-1} f_k(Y_{k+1}, Z_k)\Delta_k\right), \\ \Delta_i Z_i & = \mathbb{E}_i\left([\xi + \sum_{k=i+1}^{N-1} f_k(Y_{k+1}, Z_k)\Delta_k]\Delta W_i^\top\right). \end{cases} \qquad (\textbf{MDP})$$

✓ Without extra approximation, **ODP⟺MDP.**

✓ Differences occur when conditional expectations are approximated: **MDP $>$ ODP**

2) Usual Regression Monte Carlo method

✓ Markovian representations: $Y_i = y_i(X_i)$ and $Z_i = z_i(X_i)$

✓ Computations of y and z on approximation spaces $\mathcal{F}_i^Y, \mathcal{F}_i^Z$ (finite dimensional vector spaces: global/local polynomials, Fourier basis, wavelets. . .)

✓ N independent learning samples: at time i, $[(X_j^{i,m})_{0\le j\le N}, \Delta W_i^{i,m}]_{1\le m\le M}$.

→ Initialization : for $i = N$ take $y_N^{\mathcal{F},M}(\cdot) = g(\cdot)$.

→ Iteration : for $i = N-1, \cdots, 0$, solve the empirical least-squares problems

$$z_i^{\mathcal{F},M} = \operatorname*{arginf}_{\varphi\in\mathcal{F}_i^Z} \sum_{m=1}^{M} \left|\left[g(X_N^{i,m}) + \sum_{j\ge i+1} f(t_j, X_j^{i,m}, y_{j+1}^{\mathcal{F},M}(X_{j+1}^{i,m}), z_j^{\mathcal{F},M}(X_j^{i,m}))\Delta_j\right]\frac{\Delta W_i^{i,m}}{\Delta_i} - \varphi(X_i^{i,m})\right|^2,$$

$$y_i^{\mathcal{F},M} = \operatorname*{arginf}_{\varphi\in\mathcal{F}_i^Y} \sum_{m=1}^{M} \left|g(X_N^{i,m}) + \sum_{j\ge i} f(t_j, X_j^{i,m}, y_{j+1}^{\mathcal{F},M}(X_{j+1}^{i,m}), z_j^{\mathcal{F},M}(X_j^{i,m}))\Delta_j - \varphi(X_i^{i,m})\right|^2.$$

✓ Apply soft thresholding with explicit constants.

Theorem (Non asymptotic error estimates). $\exists\, C$ (explicit) s.t.

$$\mathbb{E}\Big[\|y_i^{\mathcal{F},M}(\cdot)-y_i(\cdot)\|_{i,M}^2\Big]\le C\inf_{\varphi\in\mathcal{F}_i^Y}\mathbb{E}|\varphi(X_i)-y_i(X_i)|^2+C\frac{\dim(\mathcal{F}_i^Y)}{M}+C\sum_{j=i}^{N-1}\mathcal{E}(j)\Delta_j,$$

$$\sum_{j=i}^{N-1}\mathbb{E}\Big[\|z_j^{\mathcal{F},M}(\cdot)-z_j(\cdot)\|_{j,M}^2\Big]\Delta_j\le C\sum_{j=i}^{N-1}\mathcal{E}(j)\Delta_j,$$

$$\mathcal{E}(j):=\inf_{\varphi\in\mathcal{F}_j^Y}\mathbb{E}|\varphi(X_j)-y_j(X_j)|^2+\inf_{\varphi\in\mathcal{F}_j^Z}\mathbb{E}|\varphi(X_j)-z_j(X_j)|^2+\left(\dim(\mathcal{F}_j^Y)+\frac{\dim(\mathcal{F}_j^Z)}{\Delta_j}\right)\frac{\log(M)}{M}.$$

- Estimates are sharp: **approximation error** + **statistical error**
- Explicit error bounds, robust w.r.t. the model and the basis
- Simulation effort: $\mathbf{M}\ge\mathbf{\Delta_i^{-1}}\max(\mathbf{N}\dim(\mathcal{F}_\mathbf{i}^\mathbf{Z}),\dim(\mathcal{F}_\mathbf{i}^\mathbf{Y}))$
- Memory effort: $\max\left(\sum_{\mathbf{i=1}}^{\mathbf{N}}\dim(\mathcal{F}_\mathbf{i}^\mathbf{Z})+\dim(\mathcal{F}_\mathbf{i}^\mathbf{Y}),\mathbf{NM}\right)=\mathbf{NM}$
- In this form, no clear parallelization

✓ **Optimal parameters**: L_2-error $=$ `Computational Cost`$^{-\frac{1}{8+\frac{\text{dimension}}{\text{smoothness of } z}}}$.

3) Stratification

Two objectives:

✓ Relaxing the requirement on M

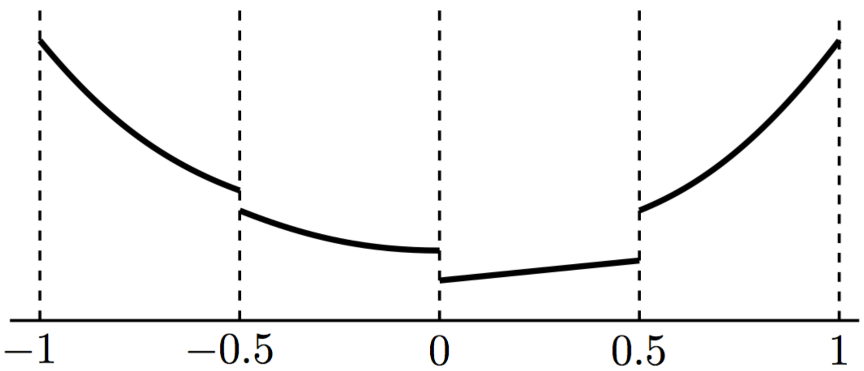
✓ Allowing parallel computations **w.r.t. the basis functions**

First choice: local approximations

✓ partition of the state space $\mathbb{R}^d$ in `strata` ➠ finite number of disjoints sets $(\mathcal{H}_k)_k$

✓ on each set $\mathcal{H}_k$, (local) polynomial

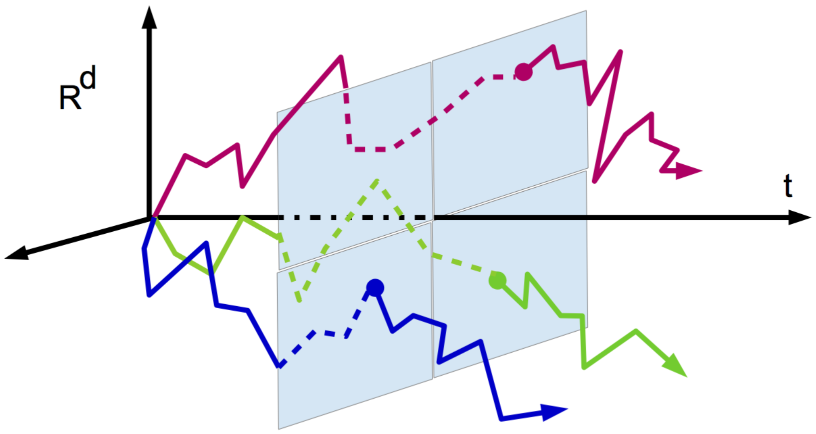
- **LP0**: piecewise constant approximation
- **LP1**: linear approximation



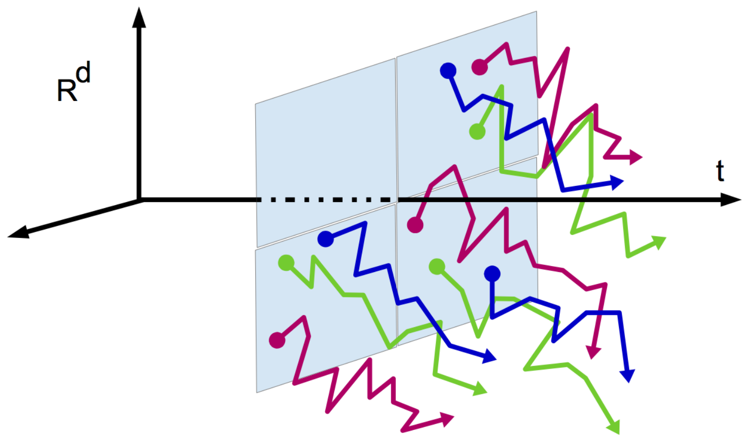
✓ function spaces $\mathcal{L}_{Y,k}, \mathcal{L}_{Z,k}$ of dimension 1 or $d+1$

✓ to get a statistical error of order N^{-1}, only N^2 simulations in $\mathcal{H}_k$ are required

Second choice: stratified simulations and regressions



Samples for usual RMC



Samples for Stratified RMC

✓ ν = probability distribution on $\mathbb{R}^d$

✓ ν_k = restriction of ν to $\mathcal{H}_k$

one should be able to simulate according to ν_k

✓ In our test: take $\mathcal{H}_k$ as hypercube and ν with independent coordinates, having the logistic distribution (1d-CDF is $F_\mu(x) = e^{\mu x}/(1 + e^{\mu x})$)

✓ At each date t_i and each stratum $\mathcal{H}_k$, draw M simulations according to ν_k and start independent M diffusion/Euler scheme from these M points.

$$
\begin{aligned}
z_i^{\mathcal{F},M}\Big|_{\mathcal{H}_k} = \operatorname*{arginf}_{\varphi\in\mathcal{L}_{Z,k}} \sum_{m=1}^{M} \Big| \Big[g(X_N^{i,k,m}) + \sum_{j\geq i+1} f(t_j, X_j^{i,k,m}, y_{j+1}^{\mathcal{F},M}(X_{j+1}^{i,k,m}), z_j^{\mathcal{F},M}(X_j^{i,k,m}))\Delta_j \Big] \\
\times \frac{\Delta W_i^{i,k,m}}{\Delta_i} - \varphi(X_i^{i,k,m}) \Big|^2,
\end{aligned}
$$

$$
y_i^{\mathcal{F},M}\Big|_{\mathcal{H}_k} = \operatorname*{arginf}_{\varphi\in\mathcal{L}_{Y,k}} \sum_{m=1}^{M} \Big| g(X_N^{i,k,m}) + \sum_{j\geq i} f(t_j, X_j^{i,k,m}, y_{j+1}^{\mathcal{F},M}(X_{j+1}^{i,k,m}), z_j^{\mathcal{F},M}(X_j^{i,k,m}))\Delta_j - \varphi(X_i^{i,k,m}) \Big|^2.
$$

- This can be done in parallel on different processors
- As many processors as the number of cubes $\mathcal{H}_k$
- Information on value functions must be shared by all the processors

Convergence analysis

To allow the control of errors propagation, one should wonder whether

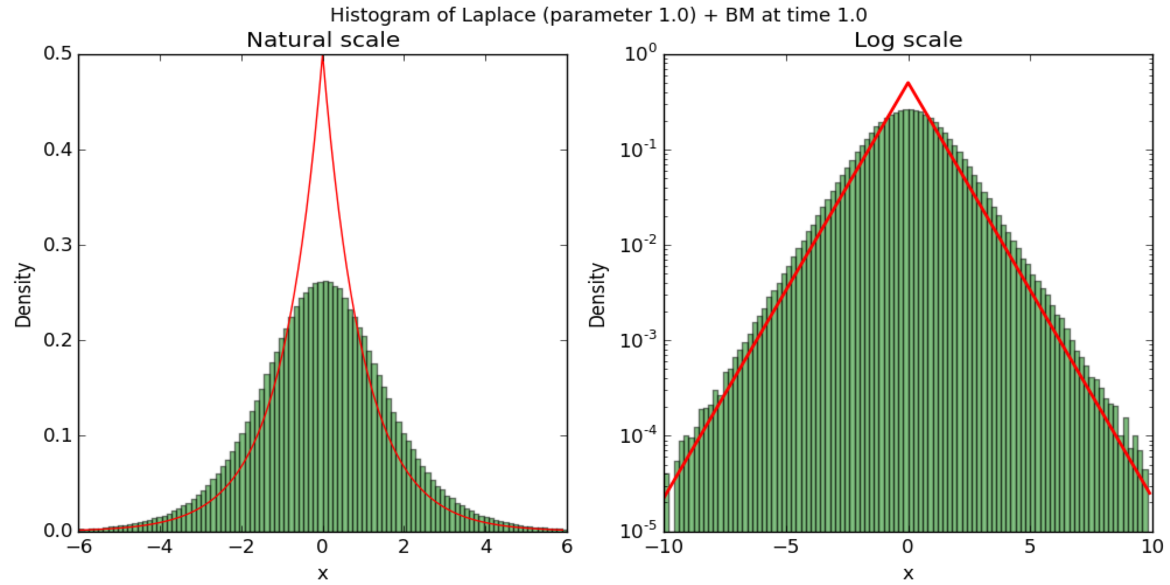
$$X_j^{i,\nu} \stackrel{d}{=} X_j^{j,\nu} (= \nu)?$$

- ✓ In general NO, since ν is not a stationary distribution and X is not ergodic
- ✓ But, we have the **BM equivalence property**: under mild assumptions on b and σ,

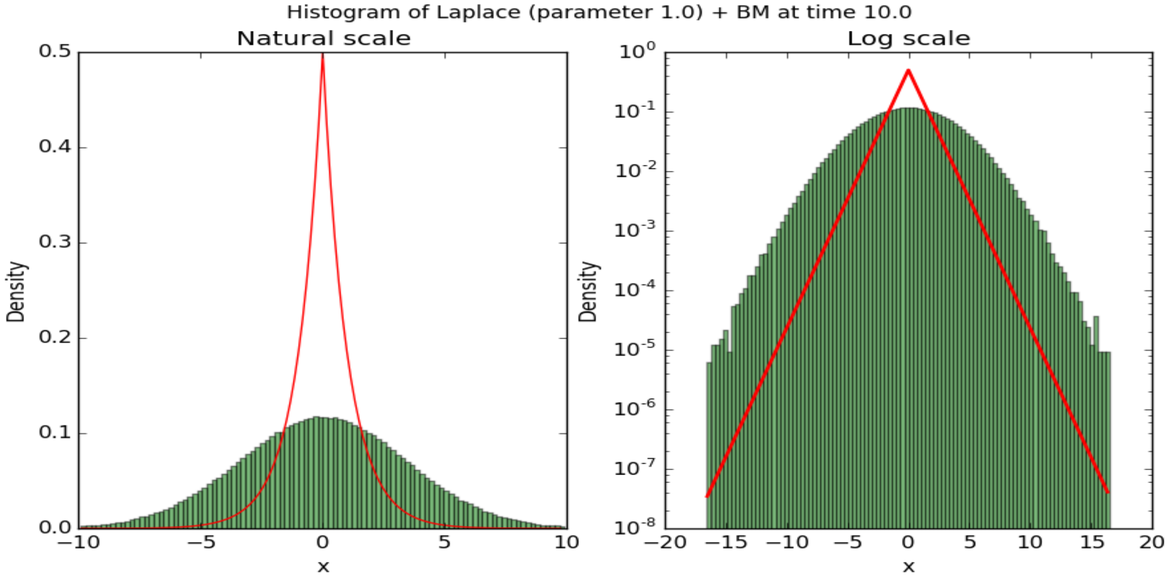
$$\mathbb{E}\left(|\mathbf{h}(\mathbf{X}_{\mathbf{j}}^{\mathbf{i},\nu})|^{\mathbf{2}}\right) \leqslant_{\mathbf{c}} \int_{\mathbb{R}^{\mathbf{d}}} |\mathbf{h}(\mathbf{x})|^{\mathbf{2}} \nu(\mathrm{d}\mathbf{x}), \quad \text{for any } \mathbf{h},$$

  with a constant c uniform in $0 \leq i \leq j \leq N$.
- ✓ Satisfied for **distributions with Sub Exponential tails** (like logistic distribution)

Example (Laplace distribution $\nu(\mathrm{d}x) = \frac{1}{2}e^{-|x|}\mathrm{d}x$).



Distribution of $X_0 + W_1$ with $X_0 \overset{\text{dist.}}{=} \nu$



Distribution of $X_0 + W_{10}$ with $X_0 \overset{\text{dist.}}{=} \nu$

Theorem (Error estimates for LP0 and LP1 spaces). For some explicit constant C, one has

$$\mathbb{E}\Big[\int_{\mathbb{R}^d} |y_i^{\mathcal{F},M}(x) - y_i(x)|^2 \nu(\mathrm{d}x)\Big] \le C\mathcal{E}(i) + C\sum_{j=i}^{N-1} \mathcal{E}(j)\Delta_j,$$

$$\sum_{j=i}^{N-1} \mathbb{E}\Big[\int_{\mathbb{R}^d} |z_j^{\mathcal{F},M}(x) - z_j(x)|^2 \nu(\mathrm{d}x)\Big]\Delta_j \le C\sum_{j=i}^{N-1} \mathcal{E}(j)\Delta_j,$$

$$\mathcal{E}(j) := \sum_k \nu(\mathcal{H}_k) \inf_{\varphi\in\mathcal{L}_{Y,k}} \int_{\mathcal{H}_k} |\varphi(x) - y_j(x)|^2 \nu_k(\mathrm{d}x)$$
$$+ \sum_k \nu(\mathcal{H}_k) \inf_{\varphi\in\mathcal{L}_{Z,k}} \int_{\mathcal{H}_k} |\varphi(x) - z_j(x)|^2 \nu_k(\mathrm{d}x) + \frac{\log(\mathbf{M})}{\boldsymbol{\Delta}_{\mathbf{j}}\mathbf{M}}.$$

Better dependency on M.

Stratified algorithm (SRMDP) vs non-stratified (LSMDP)

Algorithm	Number of simulations		Computational cost	
	LP0	**LP1**	**LP0**	**LP1**
SRMDP	N^2	N^2	$N^{4+d/2}$	$N^{4+d/4}$
LSMDP	$N^{2+d/2}$	$N^{2+d/4}$	$N^{4+d/2}$	$N^{4+d/4}$

Comparison of numerical parameters as a function of N.

Algorithm	**LP0**	**LP1**
SRMDP	$N^{1+d/2}$	$N^{1+d/4} \vee N^2$
LSMDP	$N^{2+d/2}$	$N^{2+d/4}$

Comparison of shared memory requirement as a function of N.

Recall that LSMDP can not take advantage of parallel architecture.

4) Numerical tests

We perform numerical experiments on the BSDE with data

- ✓ $g(x) = \omega(T,x)(1+\omega(T,x))^{-1}$ with $\omega(t,x) = \exp(t + \sum_{j=1}^{d} x_j)$.
- ✓ $f(t,x,y,z) = \left(\sum_{j=1}^{d} z_j\right)\left(y - \frac{2+d}{2d}\right)$
- ✓ Tests up to dimension $d = 19$

Explicit solution:

$$y_i(x) = \omega(t_i,x)(1+\omega(t_i,x))^{-1}, \qquad z_{j,i}(x) = \omega(t_i,x)(1+\omega(t_i,x))^{-2}.$$

Computer:

- ✓ GPU GeForce GTX TITAN Black with 6 GBytes of global memory
- ✓ Intel Xeon CPU E5-2620 v2 clocked at 2.10 GHz with 62 GBytes of RAM, CentOS Linux, NVIDIA CUDA SDK 7.0 and GNU C compiler 4.8.2.
- ✓ 256×64 threads configuration

★ $d = 4$, **LP0**

Δ_t	#CUBES	K	M	$MSE_{Y,\max}$	$MSE_{Y,\mathrm{av}}$	$MSE_{Z,\mathrm{av}}$	CPU	GPU
0.2	8	4096	25	−3.712973	−3.774071	−0.964842	0.23	2.00
0.1	12	20736	100	−4.066741	−4.303750	−1.607104	5.23	2.20
0.05	17	83521	400	−4.337988	−4.698645	−2.302092	171.92	12.39
0.02	28	614656	2500	−4.472564	−4.988069	−3.225411	58066.33	3070.92

★ $d = 6$, **LP0**

Δ_t	#CUBES	K	M	$MSE_{Y,\max}$	$MSE_{Y,\mathrm{av}}$	$MSE_{Z,\mathrm{av}}$	CPU	GPU
0.2	2	64	25	−2.392320	−2.451332	−0.431059	0.21	1.99
0.1	3	729	100	−2.440274	−2.500775	−1.096603	0.47	2.05
0.05	4	4096	400	−2.829757	−2.905192	−1.687142	17.21	3.15
0.02	7	117649	2500	−3.235130	−3.539011	−2.557686	13930.70	874.25

5) Data driven version with non-intrusive stratified resampler

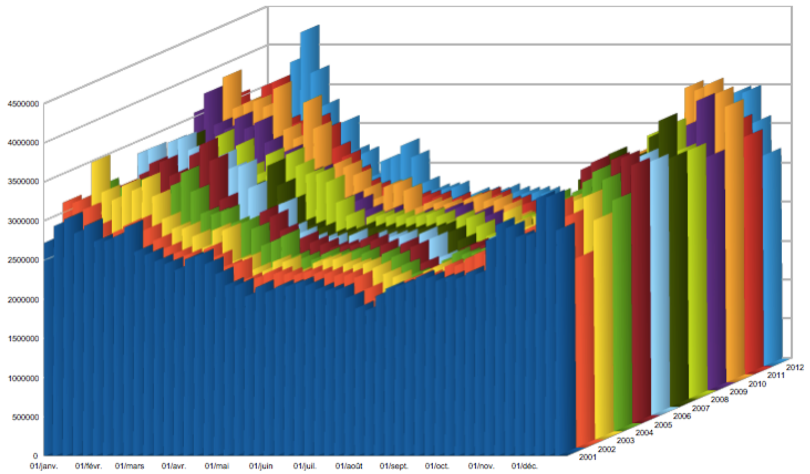
Framework.

✓ **Root sample:** M given observations of X on the period $[0, N]$.

✓ M **small:** impossible to calibrate accurately the model

✓ **Example.** Electricity consumption.

- France, weekly data, 2001-2012
- Seasonality trend
- Time-varying volatility

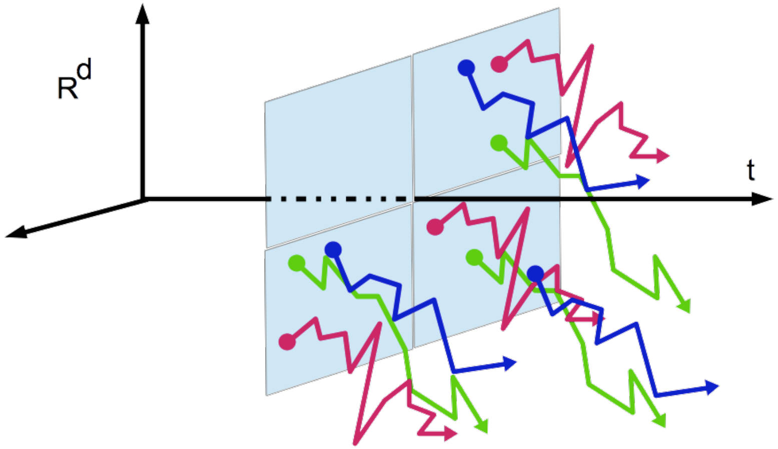


✓ Structure assumption:

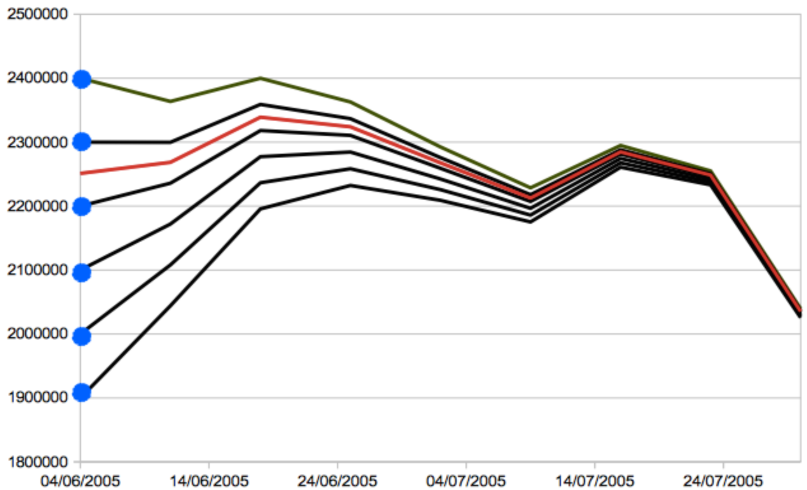
$$\mathbf{X_t} = \mathbf{x_0} - \int_0^t \mathbf{A}(\mathbf{X_s} - \bar{\mathbf{X}}_\mathbf{s})\mathrm{d}\mathbf{s} + \int_0^t \boldsymbol{\Sigma}_\mathbf{s}\mathrm{d}\mathbf{W_s} + \mathbf{L_t}$$

with A known (**only**).

Non-intrusive stratified resampler (NISR):



✓ **Example (on electricity consumptions).**



Resampling one path from any point

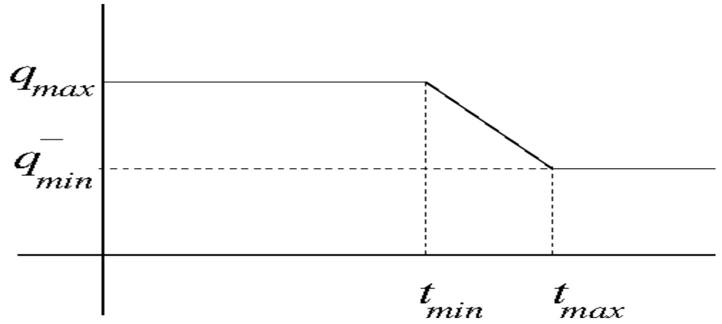
Resampling from the path 2005, between June and July, from different levels

⚠ At most, M paths can be resampled, from any point and any time (we loose independency)

▷ Optimal stopping example

A travel agency wants to launch a promotion, its profit is affected by the temperature and the exchange rate. We want to compute $v(X_0^1, X_0^2)$ defined by

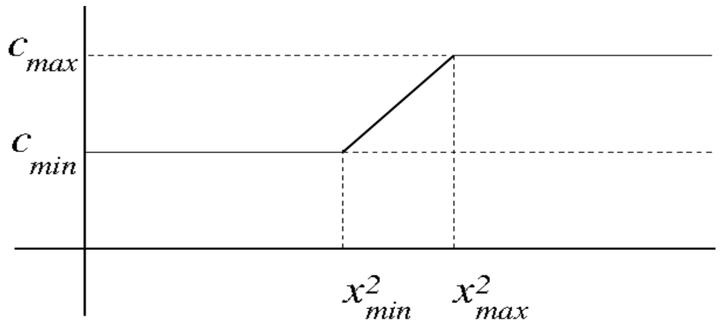
$$\operatorname*{ess\,sup}_{\tau\in\mathcal{T}} \mathbb{E}\Big[q((\tau-0.25)^2\times 240 + X_\tau^1)\, e^{-|\tau-1/6|}\Big(\underline{c} - c(e^{X^2_{\tau+1\,\text{month}}})\Big)\Big]$$



Campaign effectiveness q

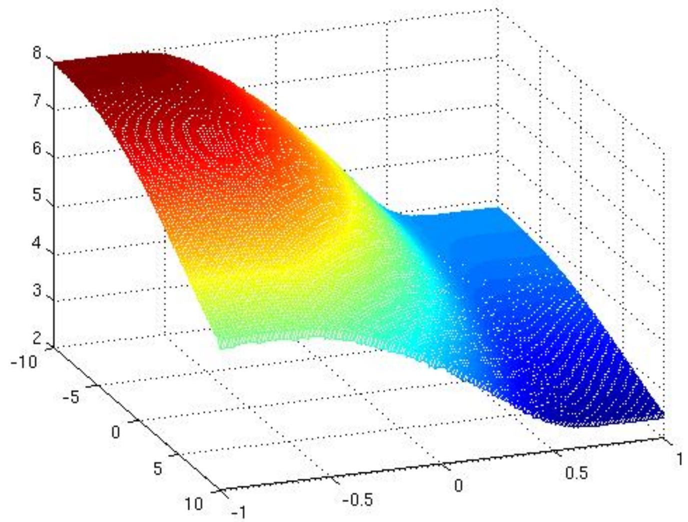
where $\mathcal{T} = \{\frac{k}{48}, k = 0, 1, \cdots, 24\}$ and

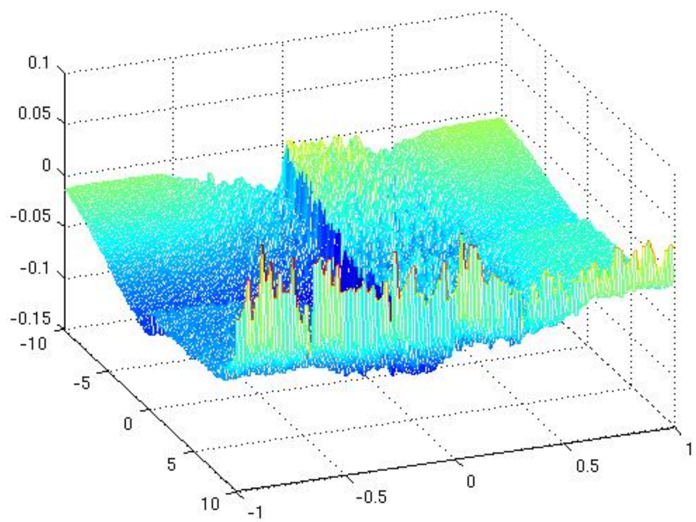
- ✓ $t = 0$ = 1st october
- ✓ $t = 1/6$ = 1st december
- ✓ $t = 0.25$ = 1st january



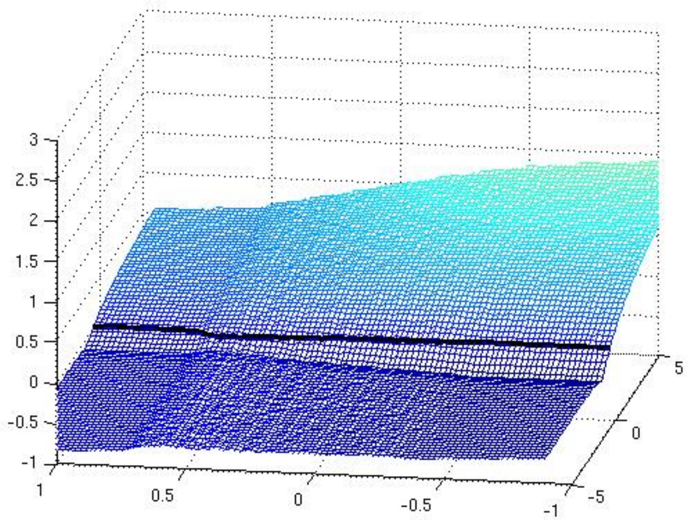
Cost function c

Model for X: X^1=OU process, X^2=ABM.





Approximation (left), error (right) at $t = 0$, $\#cubes = 100^2$, $M = 40$.



Continuation value function and exercice boundary at 8th week