A probabilistic numerical method for optimal multiple switching problem and application to investments in electricity generation

René Aïd, Luciano Campi, Nicolas Langrené, Huyên Pham

Univ. Paris Diderot - Sorbonne Paris Cité, Univ. Paris Nord, LPMA, FiME

Paris, 28th Oct. 2012

Motivation

Investments in power generation

- Capital intensive
- Very long-term returns
- Many technologies
- Many random factors :
  - Demand
  - Outages
  - Fuel prices
  - Inflows

Natural modelling : Stochastic control

- High-dimensional
- Infinite horizon

Motivation

More precisely : **Optimal multiple switching**

Outline

1. Numerical scheme
   - Convergence rate
   - Complexity analysis
2. Application
   - Detailed modelling
   - Numerical solution

Outline

1 Formulation

2 Numerical scheme

3 Complexity

4 Memory reduction

5 Application

6 Conclusion

Problem

$$v(t,x,i) = \sup_{\alpha \in \mathcal{A}_{t,i}} \mathbb{E}\left[\int_t^{\infty} f\left(s, X_s^{t,x}, I_s^{\alpha}\right) ds - \sum_{\tau_n \geq t} k\left(\tau_n, \zeta_n\right)\right]$$

- $X^{t,x}$ Markovian diffusion in $\mathbb{R}^d$, $X_t = x$
- I^{α} piecewise constant in a finite set $\mathbb{I}_q = \{i_1, i_2, \ldots, i_q\}$ ($\Rightarrow$ **optimal switching**)
- $\alpha = (\tau_n, \iota_n)_{n \in \mathbb{N}}$ impulse control strategy, τ_n increasing stopping times, ι_n random variables in $\mathbb{I}_q$, $\zeta_n = \iota_n - \iota_{n-1}$

$$I_s^{\alpha} = \iota_0 \mathbf{1}\{0 \leq s < \tau_0\} + \sum_{n \in \mathbb{N}} \iota_n \mathbf{1}\{\tau_n \leq s < \tau_{n+1}\}$$

- $\mathcal{A}_{t,i}$ set of admissible strategies s.t. $I_t^{\alpha} = i$

Assumptions

Diffusion coefficients

Lipschitz continuity & Linear growth

Reward function f

Lipschitz continuity & Linear growth

Exponential discount $e^{-\rho t}$

Cost function k

Lipschitz continuity & Linear growth

Exponential discount $e^{-\rho t}$

Minimum fixed cost

Triangular condition : $k(t;i,j) + k\left(t;j,k\right) > k\left(t;i,k\right)$

Properties

- $v_i = v(.,.,i)$ solutions of a system of Hamilton-Jacobi-Bellman Quasi-Variational Inequalities
- $v(t, X_t, i)$ solution of a Reflected Backward Stochastic Differential Equation
- **Dynamic Programming Principle** : $\forall \tau \geq t$ stopping time,

$$v(t,x,i) = \sup_{\alpha \in \mathcal{A}_{t,i}} \mathbb{E}\left[\int_t^{\tau} f\left(s, X_s^{t,x}, I_s^{\alpha}\right) ds - \sum_{t \leq \tau_n \leq \tau} k(\tau_n, \zeta_n) + v\left(\tau, X_{\tau}^{t,x}, I_{\tau}^{\alpha}\right)\right]$$

Outline

1 Formulation

2 Numerical scheme

3 Complexity

4 Memory reduction

5 Application

6 Conclusion

Finite time horizon

$$v_T(t,x,i) = \sup_{\alpha \in \mathcal{A}_{t,i}^T} \mathbb{E}\left[\int_t^T f(s,X_s^{t,x},I_s^\alpha)\,ds - \sum_{t \leq \tau_n \leq T} k(\tau_n,\zeta_n) + g\left(T,X_T^{t,x},I_T^\alpha\right)\right]$$

$$g\,(T,x,i) = \mathbb{E}\left[\int_T^\infty f\left(s,X_s^{T,x},i\right)ds\right]$$

OR any Lipschitz fct s.t. $|g(T,x,i)| \leq Ce^{-\rho T}\,(1+|x|)$

Approximation error

$$|v\,(t,x,i) - v_T\,(t,x,i)| \leq \mathbf{C}\,(\mathbf{1}+|\mathbf{x}|)\,\mathbf{e}^{-\bar{\rho}\mathbf{T}}$$

Time discretization

$$\bar{v}_{\Pi}(t,x,i) = \sup_{\alpha \in \mathcal{A}_{t,i}^{\Pi}} \mathbb{E}\left[\int_t^T f\left(\pi(s), \bar{X}_s^{t,x}, I_s^{\alpha}\right) ds - \sum_{t \leq \tau_n \leq T} k(\tau_n, \iota_{n-1}, \iota_n) + g\left(T, \bar{X}_T^{t,x}, I_T^{\alpha}\right)\right]$$

$\Pi = \{t_0 = 0 < t_1 < \ldots < t_N = T\}$, step h

$d\bar{X}_s = b\left(\pi(s), \bar{X}_{\pi(s)}\right) ds + \sigma\left(\pi(s), \bar{X}_{\pi(s)}\right) dW_s, \ 0 \leq s \leq T, \ \bar{X}_0 = x_0$

Approximation error

$$\left|v_T(0,x,i) - \bar{v}_{\Pi}(0,x,i)\right| \leq \mathbf{C}\left(\mathbf{1} + |\mathbf{x}|^{\frac{3}{2}}\right)\sqrt{\mathbf{h}}$$

cf. [Gassiat et al., 2011]

Space localization

$$\bar{X}_t \in \mathbb{R}^d \to \mathcal{P}^\varepsilon\left(\bar{X}_t\right) \in \mathcal{D}^\varepsilon$$

$\mathcal{D}^\varepsilon$ s.t. $\mathbb{E}\left[\left|\bar{X}_t - \mathcal{P}^\varepsilon\left(\bar{X}_t\right)\right|\right] \leq \varepsilon \ \forall 0 \leq t \leq T$

$\mathcal{D}^\varepsilon$ bounded domain : $\forall x \in \mathcal{D}^\varepsilon$, $|x| \leq C\left(T, \varepsilon\right)$

Approximation error

$$\left|\bar{v}_\Pi\left(0, x, i\right) - \bar{v}_\Pi^\varepsilon\left(0, x, i\right)\right| \leq \mathbf{C}\varepsilon$$

Conditional expectation approximation (1/2)

Dynamic programming principle

$$\bar{v}_\Pi(T,x,i) = g(T,x,i)$$

$$\bar{v}_\Pi(t_n,x,i) = \max_{j\in\mathbb{I}_q}\left\{hf(t_n,x,j) - k(t_n,i,j) + \Phi_j^{t_n,x}(\bar{v}_\Pi)\right\}$$

$$\Phi_j^{t_n,x}(\varphi) = \mathbb{E}\left[\varphi\left(t_{n+1},\bar{X}_{t_{n+1}},j\right)\middle|\bar{X}_{t_n}=x\right]$$

Approximation : Least-squares regression

$$\Phi_j^{t_n,x}(\varphi) \rightsquigarrow \tilde{\Phi}_j^{t_n,x}(\varphi) = \min_{\lambda\in\mathbb{R}^K}\mathbb{E}\left[\left(\varphi\left(t_{n+1},\bar{X}_{t_{n+1}},i\right) - \sum_{k=1}^K\lambda_k e_k\left(\bar{X}_{t_n}\right)\right)^2\right]$$

Conditional expectation approximation (1/2)

Dynamic programming principle

$$\bar{v}_\Pi\left(T,x,i\right)=g\left(T,x,i\right)$$

$$\bar{v}_\Pi\left(t_n,x,i\right)=\max_{j\in\mathbb{I}_q}\left\{hf\left(t_n,x,j\right)-k\left(t_n,i,j\right)+\Phi_j^{t_n,x}\left(\bar{v}_\Pi\right)\right\}$$

$$\Phi_j^{t_n,x}\left(\varphi\right)=\mathbb{E}\left[\varphi\left(t_{n+1},\bar{X}_{t_{n+1}},j\right)\middle|\bar{X}_{t_n}=x\right]$$

Approximation : **Least-squares regression**

$$\Phi_j^{t_n,x}(\varphi)\rightsquigarrow\tilde{\Phi}_j^{t_n,x}(\varphi)=\min_{\lambda\in\mathbb{R}^K}\mathbb{E}\left[\left(\varphi\left(t_{n+1},\bar{X}_{t_{n+1}},i\right)-\sum_{k=1}^{K}\lambda_k e_k\left(\bar{X}_{t_n}\right)\right)^2\right]$$

Conditional expectation approximation (1/2)

Local basis

$$e_k(x) = \mathbf{1}\{x \in B_k\}$$

$(B_k)_{1 \le k \le K}$ hypercubes $\subset \mathbb{R}^d$, partition of $\mathcal{D}^\varepsilon$, edges of length $\boldsymbol{\delta}$

cf. [Gobet et al., 2005], [Bouchard and Warin, 2011]

Approximation error

$$\left|\bar{v}_\Pi(0,x,i) - \tilde{v}_\Pi(0,x,i)\right| \le \mathbf{C}\frac{\boldsymbol{\delta}}{\mathbf{h}}$$

Conditional expectation approximation (2/2)

Sample approximation

$$\tilde{\Phi}_j^{t_n,x}(\varphi) \rightsquigarrow \hat{\Phi}_j^{t_n,x}(\varphi) = \min_{\lambda \in \mathbb{R}^K} \sum_{m=1}^{M} \left[\left(\varphi\left(t_{n+1}, \bar{X}_{t_{n+1}}^m, i\right) - \sum_{k=1}^{K} \lambda_k e_k\left(\bar{X}_{t_n}^m\right) \right)^2 \right]$$

Lemma

$$\forall p \geq 1\,,\ \left\| \hat{\Phi}_j^{t_n,x}(\varphi) - \tilde{\Phi}_j^{t_n,x}(\varphi) \right\|_p \leq \frac{C_p}{\sqrt{M}} \frac{\Gamma^{t_n,x}(\varphi) + \bar{\varphi}^{t_n}}{\mathbb{P}\left(\bar{X}_{t_n} \in B(x)\right)^{1-\frac{1}{p \vee 2}}}$$

- Extension of [Tan, 2011]
- Makes use of $\left\| \frac{1}{M} \sum_{m=1}^{M} X_m \right\|_p \leq \frac{C_p}{\sqrt{M}} \|X\|_{p \vee 2}$ (Marcinkiewicz-Zygmund + Jensen)

Conditional expectation approximation (2/2)

Sample approximation

$$\tilde{\Phi}_j^{t_n,x}(\varphi) \rightsquigarrow \hat{\Phi}_j^{t_n,x}(\varphi) = \min_{\lambda\in\mathbb{R}^K} \sum_{m=1}^{M} \left[\left(\varphi\left(t_{n+1}, \bar{X}_{t_{n+1}}^m, i\right) - \sum_{k=1}^{K} \lambda_k e_k\left(\bar{X}_{t_n}^m\right) \right)^2 \right]$$

Lemma

$$\forall p \geq 1\,, \quad \left\| \hat{\Phi}_j^{t_n,x}(\varphi) - \tilde{\Phi}_j^{t_n,x}(\varphi) \right\|_p \leq \frac{C_p}{\sqrt{M}} \frac{\Gamma^{t_n,x}(\varphi) + \bar{\varphi}^{t_n}}{\mathbb{P}\left(\bar{X}_{t_n} \in B(x)\right)^{1-\frac{1}{p\vee 2}}}$$

- Extension of [Tan, 2011]
- Makes use of $\left\| \frac{1}{M} \sum_{m=1}^{M} X_m \right\|_p \leq \frac{C_p}{\sqrt{M}} \left\| X \right\|_{p\vee 2}$ (Marcinkiewicz-Zygmund + Jensen)

Rate of convergence

Approximation error

$$\left\| \tilde{v}_\Pi (0,x,i) - \hat{v}_\Pi (0,x,i) \right\|_p \leq \frac{\mathbf{C_p}}{\mathbf{h\sqrt{M}}} \frac{\mathbf{1+C\,(T,\varepsilon)+\sqrt{h}}}{\mathbf{p\,(T,\delta,\varepsilon)^{1-\frac{1}{p\vee 2}}}}$$

$$p\,(T,\delta,\varepsilon) = \min_{t_n\in\Pi,\, B_k\subset\mathcal{D}^\varepsilon} \mathbb{P}\left(\bar{X}_{t_n} \in B_k\right)$$

Full rate of convergence

$$\left\| v (0,x,i) - \hat{v}_\Pi (0,x,i) \right\|_p \leq$$

$$C_p\left\{(1+|x|)e^{-\bar{\rho}T}+\left(1+|x|^{\frac{3}{2}}\right)\sqrt{h}+\varepsilon+\frac{\delta}{h}+\frac{C_p}{h\sqrt{M}}\frac{1+C(T,\varepsilon)+\sqrt{h}}{p(T,\delta,\varepsilon)^{1-\frac{1}{p\vee 2}}}\right\}$$

Rate of convergence

Approximation error

$$\left\| \tilde{v}_\Pi(0,x,i) - \hat{v}_\Pi(0,x,i) \right\|_p \leq \frac{C_p}{h\sqrt{M}} \frac{1 + C(T,\varepsilon) + \sqrt{h}}{p(T,\delta,\varepsilon)^{1-\frac{1}{p\vee 2}}}$$

$p(T,\delta,\varepsilon) = \min_{t_n \in \Pi,\, B_k \subset \mathcal{D}^\varepsilon} \mathbb{P}\left(\bar{X}_{t_n} \in B_k\right)$

Full rate of convergence

$$\left\| v(0,x,i) - \hat{v}_\Pi(0,x,i) \right\|_p \leq$$

$$C_p \left\{ (1+|x|)e^{-\bar{\rho}T} + \left(1+|x|^{\frac{3}{2}}\right)\sqrt{h} + \varepsilon + \frac{\delta}{h} + \frac{C_p}{h\sqrt{M}} \frac{1 + C(T,\varepsilon) + \sqrt{h}}{p(T,\delta,\varepsilon)^{1-\frac{1}{p\vee 2}}} \right\}$$

Example

$X_t = W_t$ d-dim. B.M.

$$C(T,\varepsilon) < \sqrt{T \log\left(\frac{8T}{\pi\varepsilon^{\frac{2}{d}}}\right)}$$

$$p(T,\varepsilon,\delta) \gg \frac{\varepsilon}{(4T)^d}\delta^d$$

One-parameter rate of convergence

$$\left\| v(0,x,i) - \hat{v}_{\Pi}(0,x,i) \right\|_p \leq C_p \left(1 + |x|^{\frac{3}{2}}\right) \sqrt{h}$$

- $T = \frac{1}{2\bar{\rho}} \ln\left(\frac{1}{h}\right)$, $\varepsilon = \sqrt{h}$, $\delta = h^{\frac{3}{2}}$
- $M = \frac{1}{2\bar{\rho}} \left(\frac{2}{\bar{\rho}}\right)^{2d} \left(\ln\left(\frac{1}{h}\right)\right)^{2d+1} \ln\left(\frac{4}{\pi\bar{\rho}} \frac{\ln\left(\frac{1}{h}\right)}{h^{\frac{1}{d}}}\right) \frac{1}{h^{3(d+1)}}$ ...

Example

$X_t = W_t$ d-dim. B.M.

$$C(T,\varepsilon) < \sqrt{T \log\left(\frac{8T}{\pi\varepsilon^{\frac{2}{d}}}\right)}$$

$$p(T,\varepsilon,\delta) \gg \frac{\varepsilon}{(4T)^d}\delta^d$$

One-parameter rate of convergence

$$\left\| v(0,x,i) - \hat{v}_\Pi(0,x,i) \right\|_p \leq C_p \left(1 + |x|^{\frac{3}{2}}\right)\sqrt{h}$$

- $T = \frac{1}{2\bar{\rho}} \ln\left(\frac{1}{h}\right)$, $\varepsilon = \sqrt{h}$, $\delta = h^{\frac{3}{2}}$
- $M = \frac{1}{2\bar{\rho}}\left(\frac{2}{\bar{\rho}}\right)^{2d}\left(\ln\left(\frac{1}{h}\right)\right)^{2d+1} \ln\left(\frac{4}{\pi\bar{\rho}}\frac{\ln\left(\frac{1}{h}\right)}{h^{\frac{1}{d}}}\right)\frac{1}{h^{3(d+1)}}$ . . .

Outline

1 Formulation

2 Numerical scheme

3 Complexity

4 Memory reduction

5 Application

6 Conclusion

Computational complexity

Dynamic programming

$$\hat{v}_\Pi\left(T, x_m, i\right) = g\left(T, x_m, i\right)$$
$$\hat{v}_\Pi\left(t_n, x_m, i\right) = \max_{j \in \mathbb{I}_q} \left\{ hf\left(t_n, x_m, j\right) - k\left(t_n, i, j\right) + \hat{\Phi}_j^{t_n, x_m}\left(\hat{v}_\Pi\right) \right\}$$

Complexity

$$\mathcal{O}\left(q^2 \times N \times M \log\left(M\right)\right)$$

- q =number of switches
- N =number of time steps
- M =number of Monte Carlo trajectories

or $\mathcal{O}\left(q \times N \times M \log\left(M\right)\right)$ under suitable conditions

Memory Complexity : Naive implementation

Euler scheme : **Forward**

$$
\begin{aligned}
x_{t_{n+1}} &= x_{t_n} + b\left(t_n, x_{t_n}\right) h + \sigma\left(t_n, x_{t_n}\right) \varepsilon_i \sqrt{h} \\
\varepsilon_i &\sim \mathcal{N}(0,1)
\end{aligned}
$$

Dynamic programming : **Backward**

$$
\hat{v}_\Pi(T, x_m, i) = g(T, x_m, i)
$$

$$
\hat{v}_\Pi(t_n, x_m, i) = \max_{j \in \mathbb{I}_q} \left\{ hf(t_n, x_m, j) - k(t_n, i, j) + \hat{\mathbb{E}}\left[\hat{v}_\Pi\left(t_{n+1}, \bar{X}_{t_{n+1}}^{t_n, x_m}, j\right)\right] \right\}
$$

$\Rightarrow$ storage of Monte Carlo sample $\Rightarrow$ $\mathcal{O}(N \times M)$

Outline

1. Formulation
2. Numerical scheme
3. Complexity
4. Memory reduction
5. Application
6. Conclusion

How to avoid the sample storage ?

Memory reduction method

[Chan et al., 2004] 1-dim geometric B.M.

[Chan et al., 2006] d-dim geometric B.M.

[Chan and Wu, 2011] exponential Lévy

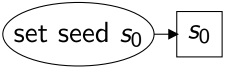
$\hookrightarrow$ limited to additive processes

Our contribution

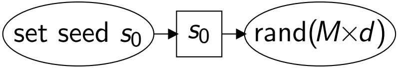
Extension to any Markovian process

(Pseudo)Random number generators

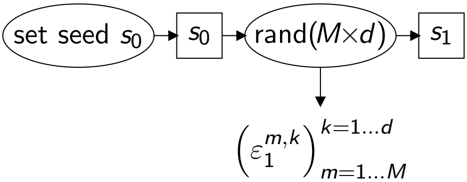
(Pseudo)Random number generators



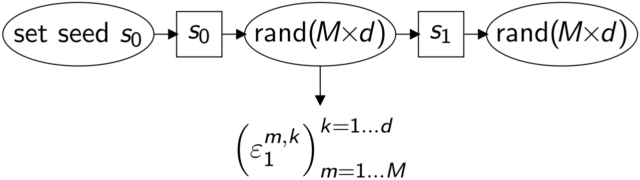
(Pseudo)Random number generators



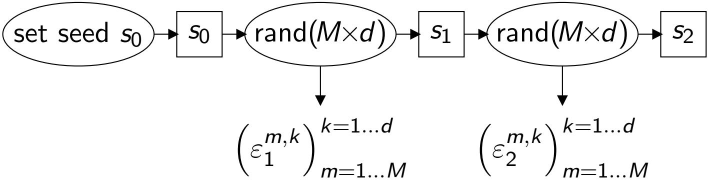
(Pseudo)Random number generators



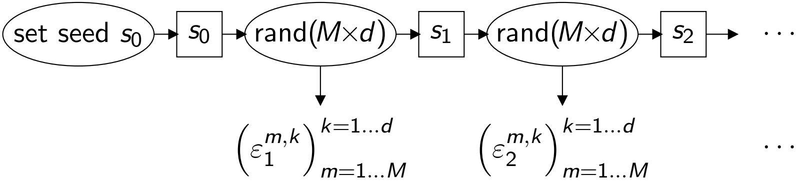
(Pseudo)Random number generators



(Pseudo)Random number generators

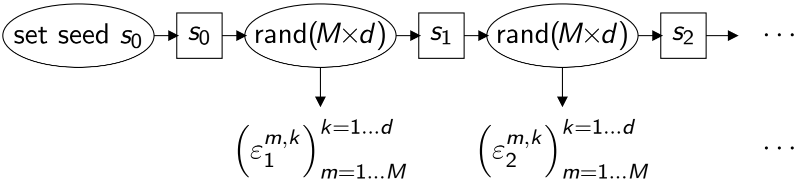


(Pseudo)Random number generators



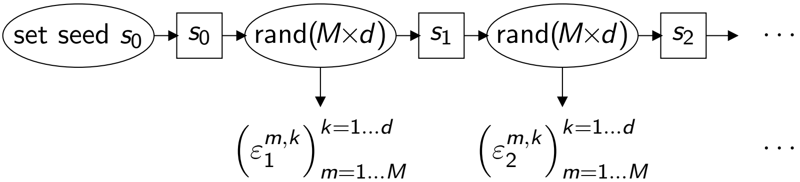
- The seeds s_n can be set and stored
- The i.i.d. sequence $\varepsilon_n^{m,k}$ can be used for the Euler scheme

(Pseudo)Random number generators



- The seeds s_n can be set and stored
- The i.i.d. sequence $\varepsilon_n^{m,k}$ can be used for the Euler scheme

(Pseudo)Random number generators



- The seeds s_n can be set and stored
- The i.i.d. sequence $\varepsilon_n^{m,k}$ can be used for the Euler scheme

Euler scheme

Case of a diffusion

$$
\begin{aligned}
x_{t_{n+1}} &= F\left(x_{t_n}, \varepsilon_i\right) \\
F\left(x, \varepsilon\right) &= x + b\left(t_n, x\right) h + \sigma\left(t_n, x\right) . \varepsilon \sqrt{h} \\
\varepsilon_i &\sim \mathcal{N}\left(0, 1\right)
\end{aligned}
$$

Euler scheme

Case of a diffusion

$$\begin{aligned} x_{t_{n+1}} &= F\left(x_{t_n}, \varepsilon_i\right) \\ F\left(x, \varepsilon\right) &= x + b\left(t_n, x\right) h + \sigma\left(t_n, x\right) . \varepsilon \sqrt{h} \\ \varepsilon_i &\sim \mathcal{N}\left(0, 1\right) \end{aligned}$$

Suppose that $\exists F^{-1}(x, \varepsilon)$ s.t. $\forall \varepsilon$, $F\left(F^{-1}(x, \varepsilon), \varepsilon\right) = x$

General memory reduction method (v1)

Euler scheme

Initialize $X[m]\,,\ m = 1..M$
for $i = 1..N-1$
  $S[i] \leftarrow \mathrm{getseed}\,()$
  for $m = 1..M$
    $\varepsilon \leftarrow \mathrm{rand}\,(d)$
    $X[m] \leftarrow F\,(X[m]\,,\varepsilon)$
  end
end
$S[N] \leftarrow \mathrm{getseed}\,()$

Inverse Euler scheme

Final values in $X[m]\,,\ m=1..M$
for $i = N-1..1$
  $\mathrm{setseed}\,(S[i])$
  for $m = 1..M$
    $\varepsilon \leftarrow \mathrm{rand}\,(d)$
    $X[m] \leftarrow F^{-1}\,(X[m]\,,\varepsilon)$
  end
end
$\mathrm{setseed}\,(S[N])$

- Doubles the number of calls to $\mathrm{rand}\,()$
- But memory complexity down to $\mathcal{O}\,(M+N)$

General memory reduction method (v1)

Euler scheme

Initialize $X[m]$, $m = 1..M$
for $i = 1..N-1$
  $S[i] \leftarrow \text{getseed}()$
  for $m = 1..M$
    $\varepsilon \leftarrow \text{rand}(d)$
    $X[m] \leftarrow F(X[m], \varepsilon)$
  end
end
$S[N] \leftarrow \text{getseed}()$

Inverse Euler scheme

Final values in $X[m]$, $m = 1..M$
for $i = N-1..1$
  $\text{setseed}(S[i])$
  for $m = 1..M$
    $\varepsilon \leftarrow \text{rand}(d)$
    $X[m] \leftarrow F^{-1}(X[m], \varepsilon)$
  end
end
$\text{setseed}(S[N])$

- Doubles the number of calls to $\text{rand}()$
- But memory complexity down to $\mathcal{O}(M+N)$

General memory reduction method (v1)

Euler scheme

Initialize $X[m]$, $m = 1..M$
for $i = 1..N-1$
 $S[i] \leftarrow \text{getseed}()$
 for $m = 1..M$
  $\varepsilon \leftarrow \text{rand}(d)$
  $X[m] \leftarrow F(X[m], \varepsilon)$
 end
end
$S[N] \leftarrow \text{getseed}()$

Inverse Euler scheme

Final values in $X[m]$, $m = 1..M$
for $i = N-1..1$
 $\text{setseed}(S[i])$
 for $m = 1..M$
  $\varepsilon \leftarrow \text{rand}(d)$
  $X[m] \leftarrow F^{-1}(X[m], \varepsilon)$
 end
end
$\text{setseed}(S[N])$

- Doubles the number of calls to $\text{rand}()$
- But memory complexity down to $\mathcal{O}(M+N)$

General memory reduction method (v1)

Euler scheme

Initialize $X[m]$, $m = 1..M$
for $i = 1..N-1$
  $S[i] \leftarrow \text{getseed}()$
  for $m = 1..M$
    $\varepsilon \leftarrow \text{rand}(d)$
    $X[m] \leftarrow F(X[m], \varepsilon)$
  end
end
$S[N] \leftarrow \text{getseed}()$

Inverse Euler scheme

Final values in $X[m]$, $m = 1..M$
for $i = N-1..1$
  $\text{setseed}(S[i])$
  for $m = 1..M$
    $\varepsilon \leftarrow \text{rand}(d)$
    $X[m] \leftarrow F^{-1}(X[m], \varepsilon)$
  end
end
$\text{setseed}(S[N])$

- Doubles the number of calls to $\text{rand}()$
- But memory complexity down to $\mathcal{O}(M+N)$

General memory reduction method (v2)

Euler scheme

for $i = 1..N-1$
  $S[i] \leftarrow \mathrm{getseed}\,()$
  for $m = 1..M$
    $\varepsilon \leftarrow \mathrm{rand}\,(d)$
  end
end
Initialize $X[m]\,,\ m = 1..M$
for $i = 1..N-1$
  $\mathrm{setseed}\,(\mathrm{S[N-i+1]})$
  for $m = 1..M$
    $\varepsilon \leftarrow \mathrm{rand}\,(d)$
    $X[m] \leftarrow F\,(X[m]\,,\varepsilon)$
  end
end

Final values in $X[m]\,,\ m = 1..M$
$\mathrm{setseed}\,(S[0])$
$\mathrm{free}\,(S)$
for $i = N-1..1$
  for $m = 1..M$
    $\varepsilon \leftarrow \mathrm{rand}\,(d)$
    $X[m] \leftarrow F^{-1}\,(X[m]\,,\varepsilon)$
  end
end

- No seed change nor storage

General memory reduction method (v2)

Euler scheme

for $i = 1..N-1$
 $S[i] \leftarrow \text{getseed}()$
 for $m = 1..M$
  $\varepsilon \leftarrow \text{rand}(d)$
 end
end
Initialize $X[m]\,,\ m = 1..M$
for $i = 1..N-1$
 $\text{setseed}(\text{S}[\text{N}-\text{i}+1])$
 for $m = 1..M$
  $\varepsilon \leftarrow \text{rand}(d)$
  $X[m] \leftarrow F(X[m], \varepsilon)$
 end
end

Inverse Euler scheme

Final values in $X[m]\,,\ m=1..M$
$\text{setseed}(S[0])$
$\text{free}(S)$
for $i = N-1..1$
 for $m = 1..M$
  $\varepsilon \leftarrow \text{rand}(d)$
  $X[m] \leftarrow F^{-1}(X[m], \varepsilon)$
 end
end

- No seed change nor storage

General memory reduction method (v2)

Euler scheme

for $i = 1..N-1$
  $S[i] \leftarrow \text{getseed}\,()$
  for $m = 1..M$
    $\varepsilon \leftarrow \text{rand}\,(d)$
  end
end
Initialize $X[m]\,,\; m = 1..M$
for $i = 1..N-1$
  $\text{setseed}\,(\mathrm{S[N-i+1]})$
  for $m = 1..M$
    $\varepsilon \leftarrow \text{rand}\,(d)$
    $X[m] \leftarrow F\,(X[m]\,,\varepsilon)$
  end
end

Inverse Euler scheme

Final values in $X[m]\,,\, m{=}1..M$
$\text{setseed}\,(S[0])$
$\text{free}\,(S)$
for $i = N-1..1$
  for $m = 1..M$
    $\varepsilon \leftarrow \text{rand}\,(d)$
    $X[m] \leftarrow F^{-1}\,(X[m]\,,\varepsilon)$
  end
end

- No seed change nor storage

General memory reduction method (v2)

Euler scheme

for $i = 1..N-1$
    $S[i] \leftarrow \text{getseed}()$
    for $m = 1..M$
        $\varepsilon \leftarrow \text{rand}(d)$
    end
end
Initialize $X[m]\,,\ m = 1..M$
for $i = 1..N-1$
    $\text{setseed}(\mathrm{S[N-i+1]})$
    for $m = 1..M$
        $\varepsilon \leftarrow \text{rand}(d)$
        $X[m] \leftarrow F(X[m], \varepsilon)$
    end
end

Inverse Euler scheme

Final values in $X[m]\,,\ m = 1..M$
$\text{setseed}(S[0])$
$\text{free}(S)$
for $i = N-1..1$
    for $m = 1..M$
        $\varepsilon \leftarrow \text{rand}(d)$
        $X[m] \leftarrow F^{-1}(X[m], \varepsilon)$
    end
end

- No seed change nor storage

Stability ?

$\mathbf{f}\left(\mathbf{f}^{-1}(\mathbf{x})\right) = \mathbf{x}$?

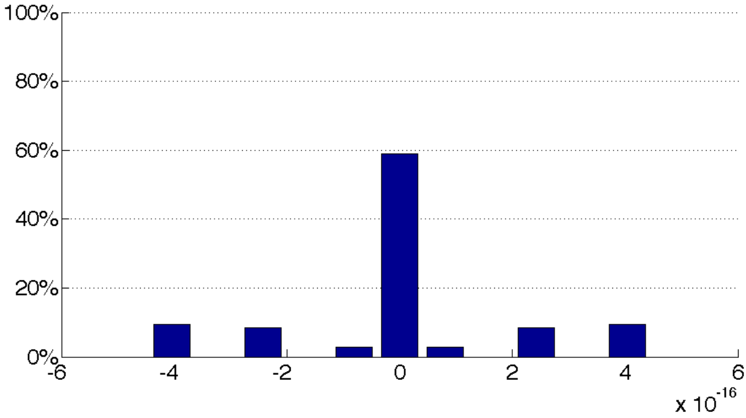


Figure: $f\left(f^{-1}(x)\right) - x$ for $f(x) = 2x + 3$ on a sample of 10^7 pts in $[0, 1]$

Stability ?

$$f\left(f^{-1}(x)\right) = x + \epsilon Z\,,\ \epsilon > 0\,,\ Z \text{ r.v.}$$

Examples

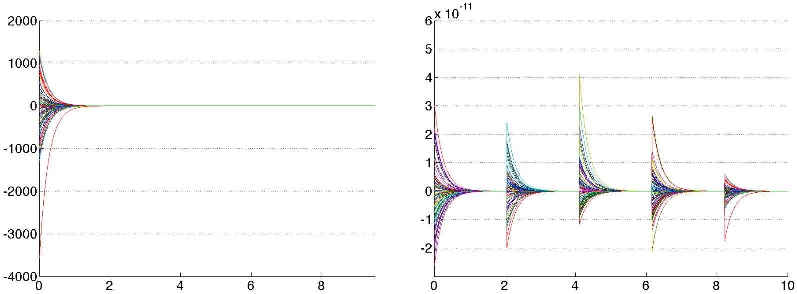
Arithmetic B.M. $\quad \tilde{x}_{t_0} - x_{t_0} = \epsilon \sum_{i=0}^{N-1} z_i$

$\hookrightarrow$ tame

Ornstein-Uhlenbeck $\quad \tilde{x}_{t_0} - x_{t_0} = \epsilon \sum_{i=0}^{N-1} \frac{1}{(1-\alpha h)^i} z_i$

$\hookrightarrow$ problem if $T > \frac{1}{\alpha} \ln\left(\frac{1}{\epsilon}\right)$

Stabilizing correction



(a) Without intermediate saves (b) With intermediate saves

Figure: Compound rounding error, Ornstein-Uhlenbeck

Outline

1 Formulation

2 Numerical scheme

3 Complexity

4 Memory reduction

5 Application

6 Conclusion

Application to investments in power generation

Features

Application to investments in power generation

Features

Not included

- Multiple agents / Strategic behaviour
- Multizone / Network
- Dynamic constraints / Time to build

Application to investments in power generation
Features

Not included

- Multiple agents / Strategic behaviour
- Multizone / Network
- Dynamic constraints / Time to build

Application to investments in power generation

Features

Not included

- Multiple agents / Strategic behaviour
- Multizone / Network
- Dynamic constraints / Time to build

Included

- Multiple factors :

Application to investments in power generation

Features

Not included

- Multiple agents / Strategic behaviour
- Multizone / Network
- Dynamic constraints / Time to build

Included

- Multiple factors :
- Power price structural model

Application to investments in power generation

Features

Not included

- Multiple agents / Strategic behaviour
- Multizone / Network
- Dynamic constraints / Time to build

Included

- Multiple factors :
  - Demand
  - Installed capacities
- Power price structural model :

Application to investments in power generation
Features

Not included
- Multiple agents / Strategic behaviour
- Multizone / Network
- Dynamic constraints / Time to build

Included
- Multiple factors :
  - Demand
  - Installed capacities
  - Random outages
  - Energy prices
- Power price structural model :

Application to investments in power generation

Features

Not included

- Multiple agents / Strategic behaviour
- Multizone / Network
- Dynamic constraints / Time to build

Included

- Multiple factors :
  - Demand
  - Installed capacities
  - Random outages
  - Energy prices
- Power price structural model :

Application to investments in power generation

Features

Not included

- Multiple agents / Strategic behaviour
- Multizone / Network
- Dynamic constraints / Time to build

Included

- Multiple factors :
  - Demand
  - Installed capacities
  - Random outages
  - Energy prices
- Power price structural model :

Application to investments in power generation

Features

Not included

- Multiple agents / Strategic behaviour
- Multizone / Network
- Dynamic constraints / Time to build

Included

- Multiple factors :
  - Demand
  - Installed capacities
  - Random outages
  - Energy prices
- Power price structural model :
  - Peaks / Scarcity

Application to investments in power generation

Features

Not included

- Multiple agents / Strategic behaviour
- Multizone / Network
- Dynamic constraints / Time to build

Included

- Multiple factors :
  - Demand
  - Installed capacities
  - Random outages
  - Energy prices
- Power price structural model :
  - Peaks / Scarcity
  - Investments feedback

Application to investments in power generation

Features

Not included

- Multiple agents / Strategic behaviour
- Multizone / Network
- Dynamic constraints / Time to build

Included

- Multiple factors :
  - Demand
  - Installed capacities
  - Random outages
  - Energy prices
- Power price structural model :
  - Peaks / Scarcity
  - **Investments feedback**

Application to investments in power generation

Features

Not included

- Multiple agents / Strategic behaviour
- Multizone / Network
- Dynamic constraints / Time to build

Included

- Multiple factors :
  - Demand
  - Installed capacities
  - Random outages
  - Energy prices
- Power price structural model :
  - Peaks / Scarcity
  - Investments feedback

Application to investments in power generation
Features

Not included

- Multiple agents / Strategic behaviour
- Multizone / Network
- Dynamic constraints / Time to build

Included

- Multiple factors :
  - Demand
  - Installed capacities
  - Random outages
  - Energy prices
- Power price structural model :
  - Peaks / Scarcity
  - **Investments feedback**

Factors modelling

Demand D_t

deterministic seasonalities + O.U. process

Capacities C_t

Controlled installed capacities : $I_t = I_{0-} + \sum_{\tau_n \leq t} \zeta^n$

Available capacities : $C_t = I_t \times A_t$

A_t = availability rate (outages)

$A_t = \mathcal{T}\,(\mathrm{O.U.} + \mathrm{seasonalities})$

$\mathcal{T} : \mathbb{R} \mapsto]0,1[$ (cf. [Wagner, 2012])

Factors modelling

Demand D_t

deterministic seasonalities + O.U. process

Capacities C_t

Controlled installed capacities : $I_t = I_{0-} + \sum_{\tau_n \leq t} \zeta^n$
Available capacities : $C_t = I_t \times A_t$
A_t = availability rate (outages)
$A_t = \mathcal{T}\left(\text{O.U.} + \text{seasonalities}\right)$
$\mathcal{T} : \mathbb{R} \mapsto]0,1[$ (cf. [Wagner, 2012])

Factors modelling

Fuels and CO_2 prices

$S_t^0 = \mathrm{CO}_2$ price
S_t^k =fuel price for tech. $k \geq 1$

$$dS_t = \Xi S_t dt + \Sigma S_t dW_t^S$$

$\Rightarrow$ **Cointegrated geometric B.M.**

- Non-stationary individual prices
- Stationary linear combinations

(provided $\mathrm{rank}(\Xi)$ not full, cf. [Benmenzer et al., 2007])

Full fuel price in €/MWh

$$\tilde{S}_t^k = h_k^0 S_t^0 + h_k S_t^k$$

Factors modelling

Fuels and CO_2 prices

$S_t^0 = \mathrm{CO}_2$ price
S_t^k =fuel price for tech. $k \geq 1$

$$dS_t = \Xi S_t dt + \Sigma S_t dW_t^S$$

$\Rightarrow$ **Cointegrated geometric B.M.**

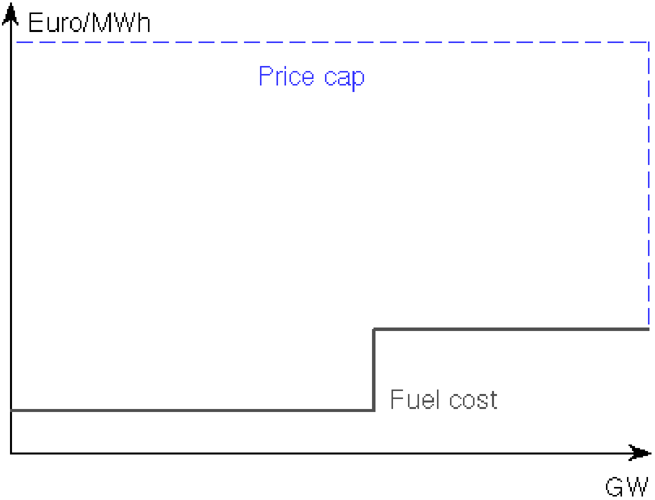
- Non-stationary individual prices
- Stationary linear combinations

(provided $\mathrm{rank}(\Xi)$ not full, cf. [Benmenzer et al., 2007])

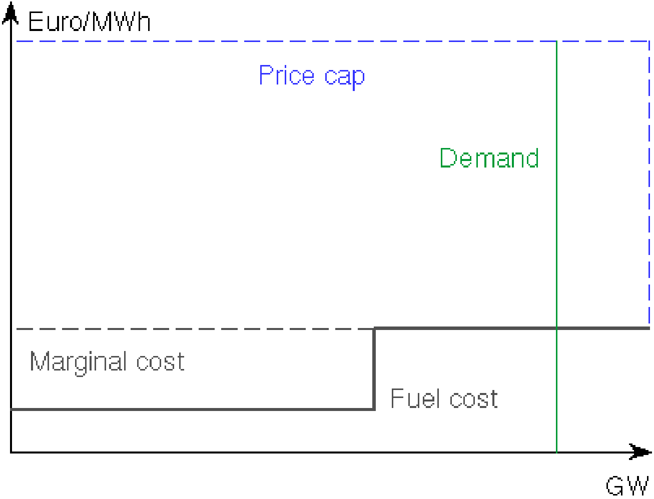
Full fuel price in €/MWh

$$\tilde{S}_t^k = h_k^0 S_t^0 + h_k S_t^k$$

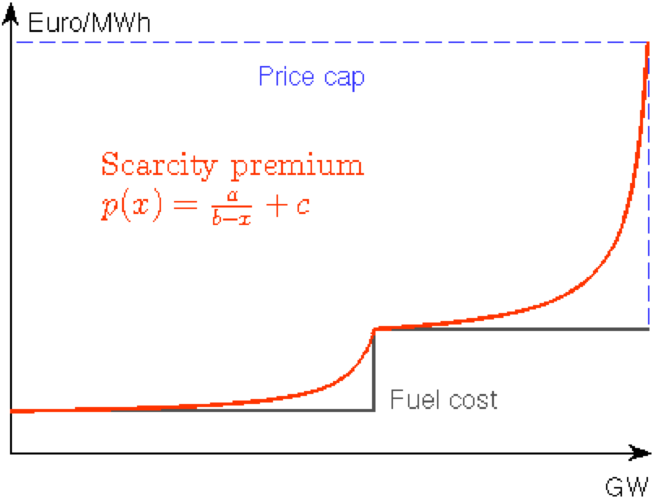
Power price



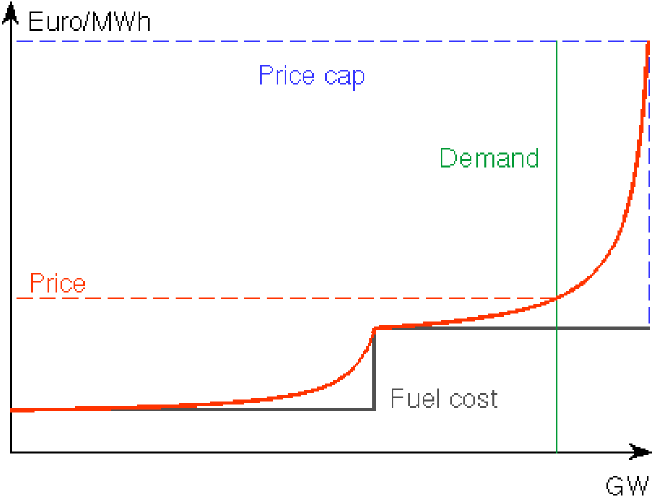
Power price



Power price


Euro/MWh
Price cap
Scarcity premium
$p(x) = \frac{a}{b-x} + c$
Fuel cost
GW

Power price



Objective function

New plant ζ^j of type j

Cost $\kappa^j\left(\zeta^j\right) = \kappa_f^j + \zeta^j \times \kappa_p^j$

Output $O_t^j = \min\left(\zeta^j \times A_t^j, \left(D_t - \overline{C}_t^{(j-1)}\right)^+\right)$

Revenue $\left(P_t - \tilde{S}_t^j\right)^+ - \kappa_m^j$

Total Gain $\int_0^\infty e^{-\rho s}\left(O_s^j\left(P_s - \tilde{S}_s^j\right)^+ - \kappa_m^j\right) ds - \kappa^j\left(\zeta^j\right)$

$$v(t,x,i) = \sup_{\alpha \in \mathcal{A}_{t,i}} \mathbb{E}\left[\sum_{j=1}^{d'} \int_t^\infty e^{-\rho s}\left(O_s^j\left(P_s - \tilde{S}_s^j\right)^+ - \kappa_m^j\right) ds - \sum_{\tau_n \geq t} e^{-\rho \tau_n} \kappa\left(\zeta^j\right)\right]$$

Objective function

New plant ζ^j of type j

Cost $\kappa^j\left(\zeta^j\right) = \kappa_f^j + \zeta^j \times \kappa_p^j$

Output $O_t^j = \min\left(\zeta^j \times A_t^j, \left(D_t - \overline{C}_t^{(j-1)}\right)^+\right)$

Revenue $\left(P_t - \tilde{S}_t^j\right)^+ - \kappa_m^j$

Total Gain $\int_0^\infty e^{-\rho s}\left(O_s^j\left(P_s - \tilde{S}_s^j\right)^+ - \kappa_m^j\right) ds - \kappa^j\left(\zeta^j\right)$

$$v(t,x,i) = \sup_{\alpha \in \mathcal{A}_{t,i}} \mathbb{E}\left[\sum_{j=1}^{d'} \int_t^\infty e^{-\rho s}\left(O_s^j\left(P_s - \tilde{S}_s^j\right)^+ - \kappa_m^j\right) ds - \sum_{\tau_n \geq t} e^{-\rho \tau_n} \kappa\left(\zeta^j\right)\right]$$

Objective function

New plant ζ^j of type j

Cost $\kappa^j\left(\zeta^j\right) = \kappa_f^j + \zeta^j \times \kappa_p^j$

Output $O_t^j = \min\left(\zeta^j \times A_t^j, \left(D_t - \overline{C}_t^{(j-1)}\right)^+\right)$

Revenue $\left(P_t - \tilde{S}_t^j\right)^+ - \kappa_m^j$

Total Gain $\int_0^\infty e^{-\rho s}\left(O_s^j\left(P_s - \tilde{S}_s^j\right)^+ - \kappa_m^j\right) ds - \kappa^j\left(\zeta^j\right)$

Value function

$$v(t,x,i) = \sup_{\alpha \in \mathcal{A}_{t,i}} \mathbb{E}\left[\sum_{j=1}^{d'} \int_t^\infty e^{-\rho s}\left(O_s^j\left(P_s - \tilde{S}_s^j\right)^+ - \kappa_m^j\right) ds - \sum_{\tau_n \geq t} e^{-\rho \tau_n} \kappa\left(\zeta^j\right)\right]$$

Objective function

New plant ζ^j of type j

Cost $\kappa^j\left(\zeta^j\right) = \kappa_f^j + \zeta^j \times \kappa_p^j$

Output $O_t^j = \min\left(\zeta^j \times A_t^j, \left(D_t - \overline{C}_t^{(j-1)}\right)^+\right)$

Revenue $\left(P_t - \tilde{S}_t^j\right)^+ - \kappa_m^j$

Total Gain $\int_0^\infty e^{-\rho s}\left(O_s^j\left(P_s - \widetilde{S}_s^j\right)^+ - \kappa_m^j\right) ds - \kappa^j\left(\zeta^j\right)$

Value function

$$v(t,x,i) = \sup_{\alpha \in \mathcal{A}_{t,i}} \mathbb{E}\left[\sum_{j=1}^{d'} \int_t^\infty e^{-\rho s}\left(O_s^j\left(P_s - \widetilde{S}_s^j\right)^+ - \kappa_m^j\right) ds - \sum_{\tau_n \geq t} e^{-\rho \tau_n} \kappa\left(\zeta^j\right)\right]$$

Objective function

New plant ζ^j of type j

Cost $\kappa^j\left(\zeta^j\right) = \kappa_f^j + \zeta^j \times \kappa_p^j$

Output $O_t^j = \min\left(\zeta^j \times A_t^j, \left(D_t - \overline{C}_t^{(j-1)}\right)^+\right)$

Revenue $\left(P_t - \tilde{S}_t^j\right)^+ - \kappa_m^j$

Total Gain $\int_0^\infty e^{-\rho s}\left(O_s^j\left(P_s - \tilde{S}_s^j\right)^+ - \kappa_m^j\right) ds - \kappa^j\left(\zeta^j\right)$

Value function

$$v(t,x,i) = \sup_{\alpha \in \mathcal{A}_{t,i}} \mathbb{E}\left[\sum_{j=1}^{d'} \int_t^\infty e^{-\rho s}\left(O_s^j\left(P_s - \tilde{S}_s^j\right)^+ - \kappa_m^j\right) ds - \sum_{\tau_n \geq t} e^{-\rho \tau_n} \kappa\left(\zeta^j\right)\right]$$

Objective function

New plant ζ^j of type j

Cost $\kappa^j\left(\zeta^j\right)=\kappa_f^j+\zeta^j\times\kappa_p^j$

Output $O_t^j=\min\left(\zeta^j\times A_t^j,\left(D_t-\overline{C}_t^{(j-1)}\right)^+\right)$

Revenue $\left(P_t-\tilde{S}_t^j\right)^+-\kappa_m^j$

Total Gain $\int_0^\infty e^{-\rho s}\left(O_s^j\left(P_s-\widetilde{S}_s^j\right)^+-\kappa_m^j\right)ds-\kappa^j\left(\zeta^j\right)$

Value function

$$v(t,x,i)=\sup_{\alpha\in\mathcal{A}_{t,i}}\mathbb{E}\left[\sum_{j=1}^{d'}\int_t^\infty e^{-\rho s}\left(O_s^j\left(P_s-\widetilde{S}_s^j\right)^+-\kappa_m^j\right)ds-\sum_{\tau_n\geq t}e^{-\rho\tau_n}\kappa\left(\zeta^j\right)\right]$$

Numerical example

2 technologies

"Base" tech. $\tilde{S}_0^1 = 40€/\text{MWh}$, vol.5%,
$I_0^1 = 67\text{GW}$, $\kappa_p^1 = 2.00\,10^9€/\text{GW}$

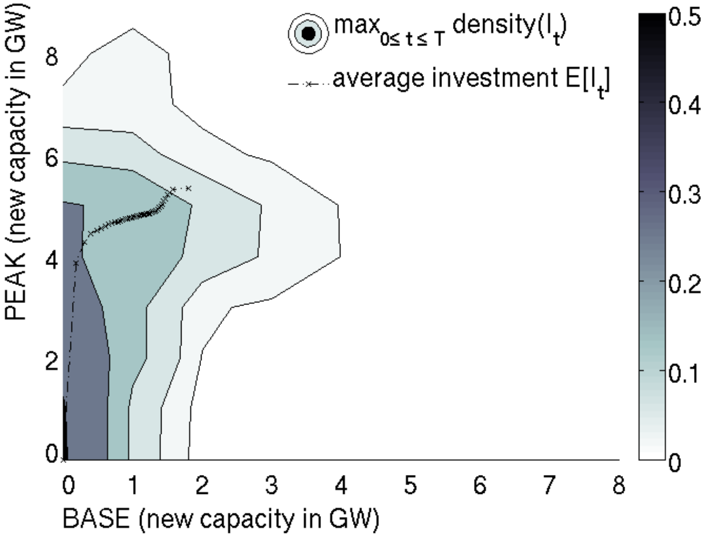
"Peak" tech. $\tilde{S}_0^2 = 80€/\text{MWh}$, vol.15%,
$I_0^1 = 33\text{GW}$, $\kappa_p^2 = 0.24\,10^9€/\text{GW}$

$\tilde{S}_t^2 - 2\tilde{S}_t^1$ stationary, $D_0 = 70\text{GW}$

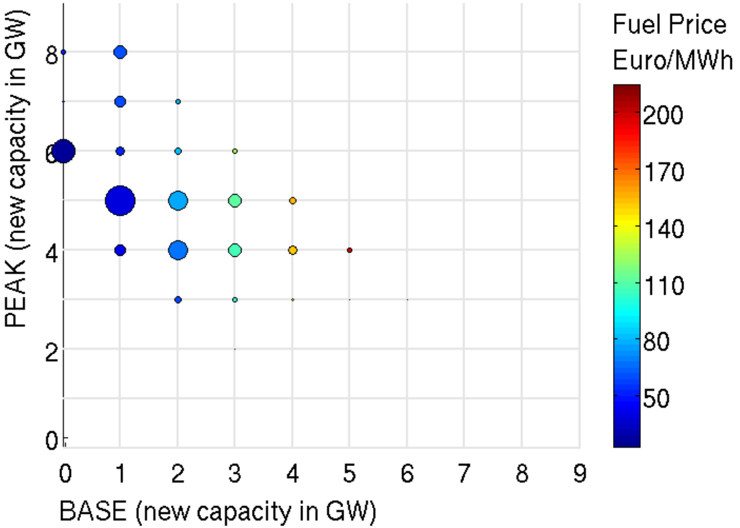
Numerical parameters

$T = 40$ years (+20 years for terminal values)
$h = 1/730$ (2 steps per day), but investments only once a year
$b = 2^6 = 64$ base functions, piecewise linear
$M = 5000$ M.C. trajectories

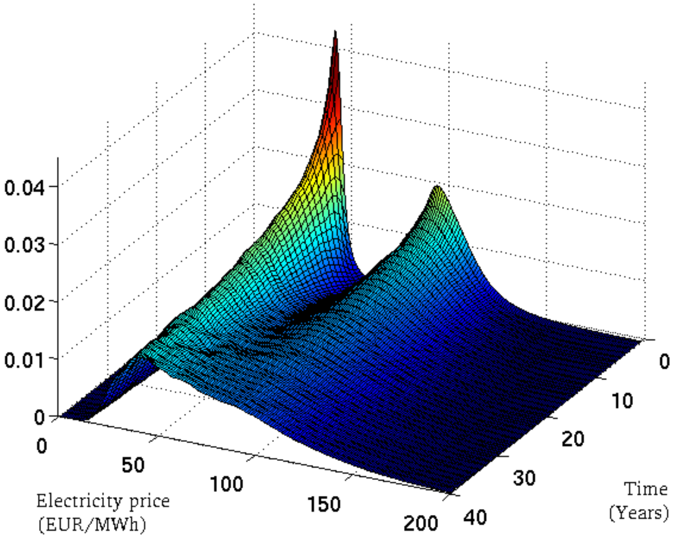
Optimal investments



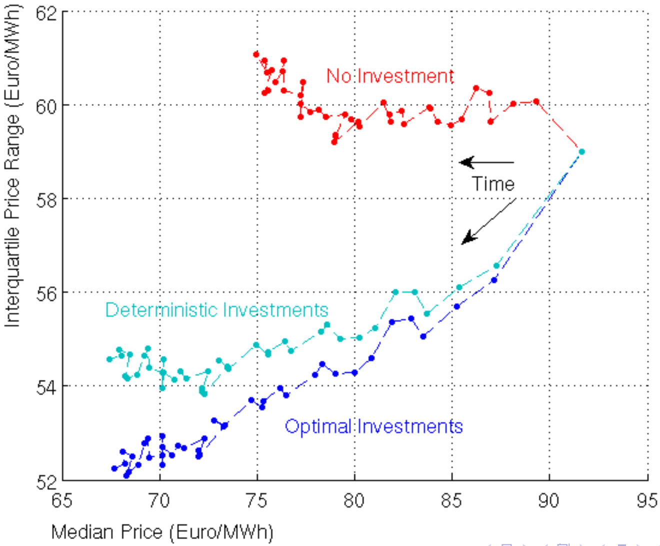
Optimal investments



Price densities



Strategies comparison



Extensions

- More technologies ⇒ more dimensions
- Asymptotic confidence intervals ⇒ second algorithm
- Multiple agents

Extensions

- More technologies $\Rightarrow$ more dimensions
- Asymptotic confidence intervals $\Rightarrow$ second algorithm
- Multiple agents

Extensions

- More technologies $\Rightarrow$ more dimensions
- Asymptotic confidence intervals $\Rightarrow$ second algorithm
- Multiple agents

Extensions

- More technologies ⇒ more dimensions
- Asymptotic confidence intervals ⇒ second algorithm
- Multiple agents

Outline

1 Formulation

2 Numerical scheme

3 Complexity

4 Memory reduction

5 Application

6 Conclusion

Conclusion

- Numerical scheme for optimal switching
  - Rate of convergence
  - Memory reduction method
- Application to investments in electricity generation
  - Structural power price model
  - Numerical resolution
- Extensions

Conclusion

- Numerical scheme for optimal switching
  - Rate of convergence
  - Memory reduction method
- Application to investments in electricity generation
  - Structural power price model
  - Numerical resolution
- Extensions

Conclusion

- Numerical scheme for optimal switching
  - Rate of convergence
  - Memory reduction method
- Application to investments in electricity generation
  - Structural power price model
  - Numerical resolution
- Extensions

Questions

References I

G. Benmenzer, E. Gobet and C. Jérusalem (2007)
Arbitrage free cointegrated models in gas and oil future markets

B. Bouchard and X. Warin (2011)
Monte Carlo valorisation of American options : facts and new algorithms to improve existing methods

R. Chan, Y. Chen, and K.-M. Yeung (2004)
A memory reduction method in pricing American options

R. Chan, C.-Y. Wong, and K.-M. Yeung (2006)
Pricing multi-asset American-style options by memory reduction Monte Carlo methods

R. Chan, and T. Wu (2011)
Memory-reduction method for pricing American-style options under exponential Lévy processes

References II

P. Gassiat, I. Kharroubi and H. Pham (2011)
Time discretisation and quantization methods for optimal multiple switching problem

E. Gobet, J.-P. Lemor and X. Warin (2005)
A regression-based Monte Carlo method to solve BSDEs

X. Tan (2011)
A splitting method for fully nonlinear degenerate parabolic PDEs

A. Wagner (2012)
Residual demand modeling and application to electricity pricing