Backward Stochastic Differential Equation

Majdi Rabia

January 23, 2017

Sommaire

Motivation

Example of a Stock Graph

American Option

Objective

- Let $t \in [0, T]$
- Payoff at time T:

$$V_T = (S_T - K)^+$$

- We can exercise at any stopping time between $[0, T]$
- Hence, we would like, with $g : x \to (x - K)^+$, to maximize $\mathbb{E}[g(S_t)]$ from today to maturity.

$$V_0 = \sup_{t \in \tau} \mathbb{E}[g(S_t)]$$

with τ the set of stopping times

Sommaire

Forward Diffusion

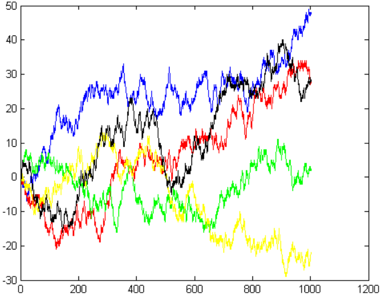
Model

$$dX_t = \mu(t, X_t)dt + \sigma(t, X_t)dB_t$$

Forward Diffusion

Model

$$dX_t = \mu(t, X_t)dt + \sigma(t, X_t)dB_t$$



Backward

Model

$$
\begin{aligned}
dX_t &= \mu(t, X_t)dt + \sigma(t, X_t)dB_t \\
-dY_t &= f(t, X_t, Y_t, Z_t)dt - Z_t dB_t \\
Y_T &= \xi(X_T)
\end{aligned}
$$

Backward

Model

$$
\begin{aligned}
dX_t &= \mu(t, X_t)dt + \sigma(t, X_t)dB_t \\
-dY_t &= f(t, X_t, Y_t, Z_t)dt - Z_t dB_t \\
Y_T &= \xi(X_T)
\end{aligned}
$$

Where :

- f is called the driver
- Z_t is a stochastic process related to $u(t, X_t) = Y_t$ by :

$$Z_t = \sigma(t, X_t) Du_x(t, X_t)$$

- ξ is a measurable function

Backward

Example : European Option

The driver for a European Option with a Geometric Brownian motion model for X_t and using risk neutral measure :

$$f(t, X_t, Y_t, Z_t) = -rY_t \tag{1}$$

Proof

$$
\begin{aligned}
dY_t &= \underbrace{\phi_t}_{\text{amount invested in stock}} \frac{dX_t}{X_t} + (Y_t - \phi_t) r dt \\
dY_t &= r\phi dt + \sigma dB_t + (Y_t - \phi_t) r dt \\
dY_t &= rY_t dt + \sigma \phi dB_t \\
-dY_t &= -rY_t dt - Z_t dB_t \qquad (2)
\end{aligned}
$$

Sommaire

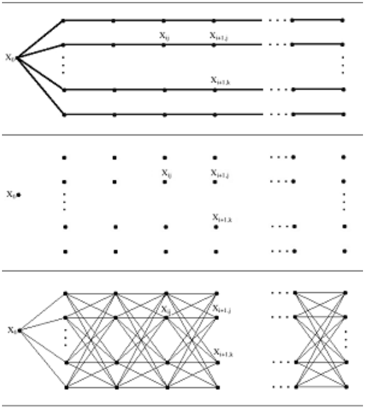
Discretization

$$Z_{t_i} = \frac{1}{\Delta t_i}\mathbb{E}[Y_{t_{i+1}}\Delta B_{t_i}|\mathcal{F}_{t_i}]$$

$$Y_{t_i} = \mathbb{E}[Y_{t_{i+1}}|\mathcal{F}_{t_i}] + f(t_i, S_{t_i}, Y_{t_{i+1}}, Z_{t_i})\Delta t_i$$

Sommaire

Mesh Method : Glasserman



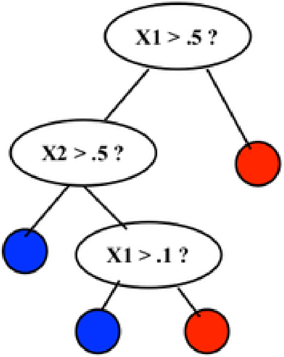
Mesh Method : Glasserman

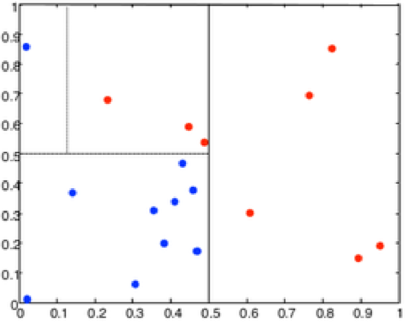
Propositon

$$\mathbb{E}[Y^j_{t_{i+1}}|(X_{t_{i+1}})] = \sum_{j=1}^{N} W^j_{i,k} Y^j_{t_{i+1}}$$

the weights being given by the properties of the diffusion

Tree Regression : Random Forest



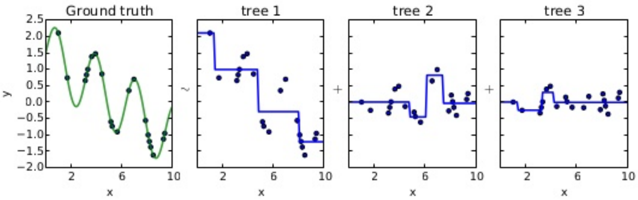


Tree Regression : Gradient Boosting

Statistical view on boosting

- $\Rightarrow$ Generalization of boosting to arbitrary loss functions

Residual fitting



Using BSDE properties

$Z_{t_i} = \frac{1}{\Delta t_i}\mathbb{E}[Y_{t_{i+1}}\Delta B_{t_i}|\mathcal{F}_{t_i}]$includes a very noisy process to approximate.

Using BSDE properties

$Z_{t_i} = \frac{1}{\Delta t_i}\mathbb{E}[Y_{t_{i+1}}\Delta B_{t_i}|\mathcal{F}_{t_i}]$includes a very noisy process to approximate.

We make use of : $Z_t = \sigma(t, X_t)Du_x(t, X_t)$ where $u(t, X_t) = Y_t$

Using BSDE properties

$Z_{t_i} = \frac{1}{\Delta t_i}\mathbb{E}[Y_{t_{i+1}}\Delta B_{t_i}|\mathcal{F}_{t_i}]$includes a very noisy process to approximate.

We make use of : $Z_t = \sigma(t, X_t)Du_x(t, X_t)$ where $u(t, X_t) = Y_t$

- 1 : $Z_{t_i} = \frac{1}{\Delta t_i}\mathbb{E}[Y_{t_{i+1}}\Delta B_{t_i}|\mathcal{F}_{t_i}]$ with one of previous methods
- 2 : $Y_{t_i} = \mathbb{E}[Y_{t_{i+1}}|\mathcal{F}_{t_i}] + f(t_i, S_{t_i}, Y_{t_{i+1}}, Z_{t_i})\Delta t_i$

We use then a Picard Iteration (considering this as a Markovian case):

- 3 : We approximate $Y_{t_i} = \hat{h}(t_i, X_{t_i})$
- 4 : $Z_{t_i} = \sigma(t_i, X_{t_i})\nabla\hat{h}(t_i, X_{t_i})$

This gives a more stable Z_t process at time t.

Sommaire

Bid-ask Spread Option

driver in this case

$$\begin{cases} f(t, Y_t, Z_t) = -Z_t\theta - rY + (R-r)(Y - \frac{Z_t}{\sigma})^- \\ \theta = \frac{\mu - r}{\sigma} \\ \xi(X_T) = (X_T - K_1) - 2(X_T - K_2) \end{cases}$$

European combination with different interest rates

We again consider a one dimensional Black-Scholes model with parameters

μ	σ	r	R	T	S_0	K_1	K_2
0.05	0.2	0.01	0.06	0.25	100	95	105

Gobet use of Hypercubes regression

Ex.1: bid-ask spread for interest rates

- Black-Scholes model and $\Phi(\mathbf{S}) = (S_T - K_1)_+ - 2(S_T - K_2)_+$.
- $f(t, x, y, z) = -\{yr + z\theta - (y - \frac{z}{\sigma})^-(R - r)\}$, $\theta = \frac{\mu - r}{\sigma}$.
- Parameters:

μ	σ	r	R	T	S_0	K_1	K_2
0.05	0.2	0.01	0.06	0.25	100	95	105

M	$N = 5$, $\delta = 5$ $D = [60, 140]$	$N = 20$, $\delta = 1$ $D = [60, 200]$	$N = 50$, $\delta = 0.5$ $D = [40, 200]$
128	3.05(**0.27**)	3.71(**0.95**)	3.69(**4.15**)
512	2.93(0.11)	3.14(0.16)	3.48(0.54)
2048	2.92(0.05)	3.00(0.03)	3.08(0.12)
8192	2.91(0.03)	2.96(0.02)	2.99(0.02)
32768	2.90(0.01)	**2.95**(0.01)	2.96(0.01)

Our Results

	stat parameter	values	
LSM	mean	2.9381	
LSM	std	0.0154	
LSM	95% confidence interval	[2.9366, 2.9396]	
LSM	min	2.9159	
LSM	max	2.9787	
Mesh	mean	2.841	
Mesh	std	0.0711	
Mesh	95% confidence interval	[2.834, 2.848]	N=1000 might be too small
Mesh	min	2.6986	
Mesh	max	2.9682	
Derivative	mean	2.9403	
Derivative	std	0.1441	
Derivative	95% confidence interval	[2.9121, 2.9685]	
Derivative	min	2.7534	
Derivative	max	3.2832	
RandomForest	mean	2.8007	
RandomForest	std	0.0508	
RandomForest	95% confidence interval	[2.7957, 2.8057]	Parameters of the method
RandomForest	min	2.7281	
RandomForest	max	2.8888	

Sommaire

RBSDE and American Option

Model

$$
\begin{aligned}
dX_t &= \mu(t, X_t)dt + \sigma(t, X_t)dB_t \\
-dY_t &= f(t, X_t, Y_t, Z_t)dt - Z_t dB_t + dL_t \\
Y_T &= \xi(X_T) \\
\int_0^T (Y_t - \xi(X_t))dL_t &= 0
\end{aligned}
$$

L_t controls Y to stay above the barrier ξ

Sommaire

- High dimension with Random Forest currently looks good, need to optimize the parameters and use multithreading to get competitive simulations and publish.

- High dimension with Random Forest currently looks good, need to optimize the parameters and use multithreading to get competitive simulations and publish.
- Adapt the Process to 2-BSDE, which requires more computation !

- High dimension with Random Forest currently looks good, need to optimize the parameters and use multithreading to get competitive simulations and publish.
- Adapt the Process to 2-BSDE, which requires more computation !
- Create a mapping non-linear PDE $\rightarrow$ to 2-BSDE

- High dimension with Random Forest currently looks good, need to optimize the parameters and use multithreading to get competitive simulations and publish.
- Adapt the Process to 2-BSDE, which requires more computation !
- Create a mapping non-linear PDE $\rightarrow$ to 2-BSDE
- Clean the code, comment and put on Github for open source.

Sommaire

Thanks to :
Alexandre Thiery (NUS, Department of Applied Probability and Statistics)
Zhou Chao (NUS, Department of Mathematics)
for supporting me throughout this research.