Ninomiya-Victoir scheme: strong convergence, antithetic version and application to multilevel estimators

A. Al Gerbi, B. Jourdain* and E. Clément†

March 3, 2020

Premia 22

1 Summary of the paper

In [2] , we were interested in the computation, by Monte Carlo methods, of the expectation $Y = \mathbb{E}\left[f\left(X_T\right)\right]$, where $f : \mathbb{R}^n \mapsto \mathbb{R}$ is a given function such that $\mathbb{E}\left[f\left(X_T\right)^2\right]$ is finite and X_T is the solution, at time $T \in \mathbb{R}_+^*$ of the stochastic differential equation of the form

$$\begin{cases} dX_t = b(X_t)dt + \sum\limits_{j=1}^{d} \sigma^j(X_t)dW_t^j, \ t \in [0,T], \\ X_0 = x. \end{cases} \tag{1.1}$$

Here, $x \in \mathbb{R}^n$ is the initial condition, $W = \left(W^1, \ldots, W^d\right)$ is a $d-$dimensional standard Brownian motion, $b : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ is the drift coefficient and $\sigma^j : \mathbb{R}^n \longrightarrow \mathbb{R}^n, j \in \{1, \ldots, d\}$, are the Brownian vector fields. We focused on minimizing the computational complexity subject to a given target error $\epsilon \in \mathbb{R}_+^*$. To measure the accuracy of an estimator $\hat{Y}$, we considered the root mean square error

$$RMSE\left(\hat{Y}, Y\right) = \mathbb{E}^{\frac{1}{2}}\left[\left|Y - \hat{Y}\right|^2\right].$$

1.1 The multilevel Monte Carlo estimator

The multilevel Monte Carlo method, introduced by Giles in [3], consists in combining multiple levels of discretization, using a geometric sequence of time steps $h_l = T/2^l, l \in \mathbb{N}$, for example. Denoting by X^{2^l} a numerical scheme, with time step h_l, the main idea of this technique is to use the following telescopic summation to control the bias

$$\mathbb{E}\left[f\left(X_T^{2^L}\right)\right] = \mathbb{E}\left[f\left(X_T^1\right)\right] + \sum_{l=1}^{L} \mathbb{E}\left[f\left(X_T^{2^l}\right) - f\left(X_T^{2^{l-1}}\right)\right].$$

*Université Paris-Est, Cermics (ENPC), INRIA, F-77455, Marne-la-Vallée, France e-mails: jourdain@cermics.enpc.fr, anis.al-gerbi@cermics.enpc.fr - This research benefited from the support of the "Chaire Risques Financiers", Fondation du Risque.

†Université Paris-Est, LAMA (UMR 8050), UPEMLV, UPEC, CNRS, F-77454, Marne-la-Vallée, France, e-mail: emmanuelle.clement@u-pem.fr.

Then, a generalized multilevel Monte Carlo estimator is built as follows

$$\hat{Y}_{MLMC} = \sum_{l=0}^{L} \frac{1}{M_l} \sum_{k=1}^{M_l} Z_k^l, \tag{1.2}$$

where $L \in \mathbb{N}^*$ is the last and finest level of discretization with time-step $T/2^L$, $(M_l)_{0\leq l\leq L} \in (\mathbb{N}^*)^{L+1}$ is the vector of sample sizes at each level, $\left(Z_k^l\right)_{0\leq l\leq L, 1\leq k\leq M_l}$ are independent random variables such that for, a given discretization level $l \in \{0,\dots,L\}$, the sequence $\left(Z_k^l\right)_{1\leq k\leq M_l}$ is identically distributed and satisfies

$$\mathbb{E}\left[Z^0\right] = \mathbb{E}\left[f\left(X_T^1\right)\right], \tag{1.3}$$

and

$$\forall l \in \{1,\dots,L\}, \mathbb{E}\left[Z^l\right] = \mathbb{E}\left[f\left(X_T^{2^l}\right) - f\left(X_T^{2^{l-1}}\right)\right]. \tag{1.4}$$

Assume that, for a given discretization level $l \in \{0,\dots,L\}$, the computational cost of simulating one sample Z^l is $C\lambda_l 2^l$, where $C \in \mathbb{R}_+$ is a constant, depending only on the discretization scheme and $\lambda_l \in \mathbb{Q}_+^*$ is a weight, depending only on l. The computational complexity of $\hat{Y}_{MLMC}$, denoted by $\mathcal{C}_{MLMC}$, is given by

$$\mathcal{C}_{MLMC} = C \sum_{l=0}^{L} M_l \lambda_l 2^l. \tag{1.5}$$

The natural choice for $Z^l, l \in \{0,\dots,L\}$, considered in [3] is

$$Z^0 = f\left(X_T^1\right), \tag{1.6}$$

$$\forall l \in \{1,\dots,L\}, Z^l = f\left(X_T^{2^l}\right) - f\left(X_T^{2^{l-1}}\right). \tag{1.7}$$

For this canonical choice, it is natural to take $\lambda_0 = 1$ and $\forall l \in \{1,\dots,L\}$, $\lambda_l = 3/2$. According to Theorem 3.1 in [3] the optimal complexity $\mathcal{C}^*_{MLMC}$, depends on the order α of weak convergence of the scheme and the order β of convergence to 0 of the variance of Z^l. Here, we recall this complexity theorem.

Theorem 1.1. *Assume that*

$$\mathbb{E}\left[f\left(X_T^{2^l}\right)\right] - Y = \frac{c_1}{2^{\alpha l}} + o\left(\frac{1}{2^{\alpha l}}\right), \tag{1.8}$$

and

$$\mathbb{V}\left(Z^l\right) = \frac{c_2}{2^{\beta l}} + o\left(\frac{1}{2^{\beta l}}\right), \tag{1.9}$$

for some constants $c_1 \in \mathbb{R}^*$ and $c_2 \in \mathbb{R}_+^*$ independent of l. Then, by choosing:

$$L^* = \left\lceil \frac{\log_2\left(\frac{\sqrt{2}|c_1|}{\epsilon}\right)}{\alpha} \right\rceil, \tag{1.10}$$

and

$$\forall l \in \{0,\dots,L^*\}, M_l^* = \left\lceil \frac{2}{\epsilon^2} \sqrt{\frac{\mathbb{V}\left(Z^l\right)}{\lambda_l 2^l}} \sum_{j=0}^{L^*} \sqrt{\lambda_j 2^j \mathbb{V}\left(Z^j\right)} \right\rceil, \tag{1.11}$$

we get an optimal computational complexity:

$$\begin{cases} \mathcal{C}^*_{MLMC} = O\left(\epsilon^{-2}\right) & \text{if } \beta > 1, \\ \mathcal{C}^*_{MLMC} = O\left(\epsilon^{-2}\left(\log\left(\frac{1}{\epsilon}\right)\right)^2\right) & \text{if } \beta = 1, \\ \mathcal{C}^*_{MLMC} = O\left(\epsilon^{-2+\frac{\beta-1}{\alpha}}\right) & \text{if } \beta < 1, \end{cases} \tag{1.12}$$

with $RMSE\left(\hat{Y}_{MLMC}, Y\right)$ *bounded by* ϵ. *Here,* $\lceil x \rceil$, *for* $x \in \mathbb{R}_+$, *denotes the unique* $n \in \mathbb{N}^*$ *satisfying* $n - 1 < x \leq n$.

To obtain the estimation (1.9), the key point is that the simulation of X^{2^l} and $X^{2^{l-1}}$ comes from the same Brownian path. We easily bound the variance convergence rate from below using the strong convergence rate γ of the numerical scheme, since in general, $\beta \geq 2\gamma$ for a smooth payoff. To attain $\gamma = 1$, one has in general to simulate iterated Brownian integrals involving Lévy areas, for which there is no known efficient method. To get around this difficulty, Giles and Szpruch introduced, in [4], a Milstein scheme without Lévy areas and its antithetic version by swapping the Brownian increments. In a multilevel Monte Carlo method, using the arithmetic average of the modified Milstein scheme and its antithetic version in the finest grid, and the modified Milstein scheme in the coarsest grid leads to $\beta = 2$ and $\alpha = 1$. By this way, Giles and Szpruch managed to improve the variance convergence rate without simulating the Lévy areas. To be more specific, they choose Z^l as follows

$$Z^0_{GS} = f\left(X_T^{GS,1}\right), \tag{1.13}$$

$$\forall l \in \{1, \ldots, L\}, Z^l_{GS} = \frac{1}{2}\left(f\left(\tilde{X}_T^{GS,2^l}\right) + f\left(X_T^{GS,2^l}\right)\right) - f\left(X_T^{GS,2^{l-1}}\right). \tag{1.14}$$

Here, $X^{GS,2^l}$ is the Giles and Szpruch scheme using a grid with time step $h_l = T/2^l$ and $\tilde{X}^{GS,2^l}$ is an antithetic discretization defined by swapping each successive pair of Brownian increments in the scheme. In [2], we proposed to adapt their technique to the Ninomiya-Victoir scheme, which is known to exhibit weak convergence with order $\alpha = 2$. We managed to reduce the constant in the computational complexity by decreasing the number of discretization levels.

1.2 The Ninomiya-Victoir scheme and its antithetic version

To discretize (1.1) we consider the Ninomiya-Victoir scheme introduced in [5]. To deal with the Ninomiya-Victoir scheme, it is more convenient to rewrite the stochastic differential equation (1.1) in Stratonovich form. Assuming $\mathcal{C}^1$ regularity for the Brownian vector fields, the Stratonovich form of (1.1) is given by:

$$\begin{cases} dX_t = \sigma^0(X_t)dt + \sum_{j=1}^{d} \sigma^j(X_t) \circ dW_t^j \\ X_0 = x \end{cases} \tag{1.15}$$

where $\sigma^0 = b - \frac{1}{2}\sum_{j=1}^{d} \partial\sigma^j\sigma^j$ and $\partial\sigma^j$ is the Jacobian matrix of σ^j defined as follows

$$\partial\sigma^j = \left(\partial_{x_k}\sigma^{ij}\right)_{i,k\in[\![1;n]\!]}.$$

Now, we introduce some notations to define the Ninomiya-Victoir scheme and its antithetic version.

- We consider two grids, a coarse grid with time step $h_{l-1} = T/2^{l-1}$ and a fine grid with time step $h_l = T/2^{l-1}$. The discretization times $(t_k)_{0\leq k\leq 2^{l-1}}$ and $\left(t_{k+\frac{1}{2}}\right)_{0\leq k\leq 2^{l-1}-1}$ are defined by $\forall k \in \left\{0,\dots,2^{l-1}\right\}, t_k = kh_{l-1}$, and $\forall k \in \left\{0,\dots,2^{l-1}-1\right\}, t_{k+\frac{1}{2}} = \left(k+\frac{1}{2}\right)h_{l-1}$.
- For $V : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ Lipschitz continuous, $\exp(tV)x_0$ denotes the solution, at time $t \in \mathbb{R}$, of the following ordinary differential equation in $\mathbb{R}^n$

$$\begin{cases} \frac{dx(t)}{dt} = V\left(x(t)\right) \\ x(0) = x_0. \end{cases} \tag{1.16}$$

- Let $\eta^{2^l} = (\eta_k)_{1\leq k\leq 2^l}$ be a sequence of independent, identically distributed Rademacher random variables independent of W.

On the coarsest grid, the Ninomiya-Victoir scheme $\left(X_{t_k}^{NV,2^{l-1},\eta^{2^l}}\right)_{k\in\left\{0,\dots,2^{l-1}\right\}}$ is defined inductively by $X_{t_0}^{NV,2^{l-1},\eta^{2^l}} = x$, and for $k \in \left\{0,\dots,2^{l-1}-1\right\}$,

- if $\eta_{2k+1} = 1$:

$$X_{t_{k+1}}^{NV,2^{l-1},\eta^{2^l}} = \exp\left(\frac{h_{l-1}}{2}\sigma^0\right)\exp\left(\Delta W_{t_{k+1}}^{d,c}\sigma^d\right)\dots\exp\left(\Delta W_{t_{k+1}}^{1,c}\sigma^1\right)\exp\left(\frac{h_{l-1}}{2}\sigma^0\right)X_{t_k}^{NV,2^{l-1},\eta^{2^l}}, \tag{1.17}$$

- and if $\eta_{2k+1} = -1$:

$$X_{t_{k+1}}^{NV,2^{l-1},\eta^{2^l}} = \exp\left(\frac{h_{l-1}}{2}\sigma^0\right)\exp\left(\Delta W_{t_{k+1}}^{1,c}\sigma^d\right)\dots\exp\left(\Delta W_{t_{k+1}}^{d,c}\sigma^1\right)\exp\left(\frac{h_{l-1}}{2}\sigma^0\right)X_{t_k}^{NV,2^{l-1},\eta^{2^l}}, \tag{1.18}$$

where $\Delta W_{t_{k+1}}^c = W_{t_{k+1}} - W_{t_k}$. Similarly, on the finest grid, the Ninomiya-Victoir scheme $\left(X_{t_k}^{NV,2^l,\eta^{2^l}}\right)_{k\in\left\{0,\dots,2^{l-1}\right\}}$ is defined inductively by $X_{t_0}^{NV,2^l,\eta^{2^l}} = x$, and for $k \in \left\{0,\dots,2^{l-1}-1\right\}$,

- if $\eta_{2k+1} = 1$:

$$X_{t_{k+\frac{1}{2}}}^{NV,2^l,\eta^{2^l}} = \exp\left(\frac{h_l}{2}\sigma^0\right)\exp\left(\Delta W_{t_{k+\frac{1}{2}}}^{d,f}\sigma^d\right)\dots\exp\left(\Delta W_{t_{k+\frac{1}{2}}}^{1,f}\sigma^1\right)\exp\left(\frac{h_l}{2}\sigma^0\right)X_{t_k}^{NV,2^l,\eta^{2^l}}, \tag{1.19}$$

- and if $\eta_{2k+1} = -1$:

$$X_{t_{k+\frac{1}{2}}}^{NV,2^l,\eta^{2^l}} = \exp\left(\frac{h_l}{2}\sigma^0\right)\exp\left(\Delta W_{t_{k+\frac{1}{2}}}^{1,f}\sigma^d\right)\dots\exp\left(\Delta W_{t_{k+\frac{1}{2}}}^{d,f}\sigma^1\right)\exp\left(\frac{h_l}{2}\sigma^0\right)X_{t_k}^{NV,2^l,\eta^{2^l}}, \tag{1.20}$$

- if $\eta_{2k+2} = 1$:

$$X_{t_{k+1}}^{NV,2^l,\eta^{2^l}} = \exp\left(\frac{h_l}{2}\sigma^0\right)\exp\left(\Delta W_{t_{k+1}}^{d,f}\sigma^d\right)\dots\exp\left(\Delta W_{t_{k+1}}^{1,f}\sigma^1\right)\exp\left(\frac{h_l}{2}\sigma^0\right)X_{t_{k+\frac{1}{2}}}^{NV,2^l,\eta^{2^l}}, \tag{1.21}$$

- and if $\eta_{2k+2} = -1$:

$$X_{t_{k+1}}^{NV,2^l,\eta^{2^l}} = \exp\left(\frac{h_l}{2}\sigma^0\right)\exp\left(\Delta W_{t_{k+1}}^{1,f}\sigma^d\right)\ldots\exp\left(\Delta W_{t_{k+1}}^{d,f}\sigma^1\right)\exp\left(\frac{h_l}{2}\sigma^0\right)X_{t_{k+\frac{1}{2}}}^{NV,2^l,\eta^{2^l}}, \tag{1.22}$$

where $\Delta W_{t_{k+\frac{1}{2}}}^f = W_{t_{k+\frac{1}{2}}} - W_{t_k}$, $\Delta W_{t_{k+1}}^f = W_{t_{k+1}}^f - W_{t_{k+\frac{1}{2}}}^f$. The antithetic scheme is defined by the same iterative equations, except that the Brownian increments $\Delta W_{t_{k+\frac{1}{2}}}^f$ and $\Delta W_{t_{k+1}}^f$ are swapped. More precisely, the antithetic version of the Ninomiya-Victoir scheme $\left(\tilde{X}_{t_k}^{NV,2^{l-1},\eta^{2^l}}\right)_{k\in\left\{0,\ldots,2^{l-1}\right\}}$, with respect to $\left(X_{t_k}^{NV,2^l,\eta^{2^l}}\right)_{k\in\left\{0,\ldots,2^{l-1}\right\}}$, is defined inductively by $\tilde{X}_{t_0}^{NV,2^{l-1},\eta^{2^l}} = x$, and for $k \in \left\{0,\ldots,2^{l-1}-1\right\}$,

- if $\eta_{2k+1} = 1$:

$$X_{t_{k+1}}^{NV,2^l,\eta^{2^l}} = \exp\left(\frac{h_l}{2}\sigma^0\right)\exp\left(\Delta W_{t_{k+\frac{1}{2}}}^{d,f}\sigma^d\right)\ldots\exp\left(\Delta W_{t_{k+1}}^{1,f}\sigma^1\right)\exp\left(\frac{h_l}{2}\sigma^0\right)X_{t_k}^{NV,2^l,\eta^{2^l}}, \tag{1.23}$$

- and if $\eta_{2k+1} = -1$:

$$X_{t_{k+\frac{1}{2}}}^{NV,2^l,\eta^{2^l}} = \exp\left(\frac{h_l}{2}\sigma^0\right)\exp\left(\Delta W_{t_{k+1}}^{1,f}\sigma^d\right)\ldots\exp\left(\Delta W_{t_{k+1}}^{d,f}\sigma^1\right)\exp\left(\frac{h_l}{2}\sigma^0\right)X_{t_k}^{NV,2^l,\eta^{2^l}}, \tag{1.24}$$

- if $\eta_{2k+2} = 1$:

$$X_{t_{k+1}}^{NV,2^l,\eta^{2^l}} = \exp\left(\frac{h_l}{2}\sigma^0\right)\exp\left(\Delta W_{t_{k+\frac{1}{2}}}^{d,f}\sigma^d\right)\ldots\exp\left(\Delta W_{t_{k+\frac{1}{2}}}^{1,f}\sigma^1\right)\exp\left(\frac{h_l}{2}\sigma^0\right)X_{t_{k+\frac{1}{2}}}^{NV,2^l,\eta^{2^l}}, \tag{1.25}$$

- and if $\eta_{2k+2} = -1$:

$$X_{t_{k+1}}^{NV,2^l,\eta^{2^l}} = \exp\left(\frac{h_l}{2}\sigma^0\right)\exp\left(\Delta W_{t_{k+\frac{1}{2}}}^{1,f}\sigma^d\right)\ldots\exp\left(\Delta W_{t_{k+\frac{1}{2}}}^{d,f}\sigma^1\right)\exp\left(\frac{h_l}{2}\sigma^0\right)X_{t_{k+\frac{1}{2}}}^{NV,2^l,\eta^{2^l}}. \tag{1.26}$$

With regards to the multilevel Monte Carlo estimator, we propose to choose

$$Z_{NV}^0 = f\left(X_T^{NV,1,\eta^1}\right), \tag{1.27}$$

and, $\forall l \in \{1,\ldots,L\}$,

$$\begin{aligned} Z_{NV}^l = \frac{1}{4}&\left(f\left(\tilde{X}_T^{NV,2^l,\eta^{2^l}}\right) + f\left(\tilde{X}_T^{NV,2^l,-\eta^{2^l}}\right) + f\left(X_T^{NV,2^l,\eta^{2^l}}\right) + f\left(X_T^{NV,2^l,-\eta^{2^l}}\right)\right) \\ &- \frac{1}{2}\left(f\left(X_T^{NV,2^{l-1},\eta^{2^l}}\right) + f\left(X_T^{NV,2^{l-1},-\eta^{2^l}}\right)\right). \end{aligned} \tag{1.28}$$

Under some regularity assumptions, in [2], we proved that the order of convergence to 0 of the variance of Z_{NV}^l is $\beta = 2$, which leads to an optimal complexity $O\left(\epsilon^{-2}\right)$. More precisely, we proved the following result.

Proposition 1.2. *Assume that $f \in \mathcal{C}^2(\mathbb{R}^n, \mathbb{R})$ with bounded first and second order derivatives, $b \in \mathcal{C}^2(\mathbb{R}^n, \mathbb{R}^n)$ with bounded first and second order derivatives, $\forall j \in \{1, \dots, d\}$, $\sigma^j \in \mathcal{C}^3(\mathbb{R}^n, \mathbb{R}^n)$ with bounded first and second order derivatives and with polynomially growing third order derivatives, and that $\forall j, m \in \{1, \dots, d\}$, $\partial\sigma^j\sigma^m$ has bounded first order derivatives. Then:*

$$\forall p \geq 1, \exists c \in \mathbb{R}_+^*, \forall l \in \mathbb{N}^*, \ \mathbb{E}\left[\left|Z_{NV}^l\right|^{2p}\right] \leq \frac{c}{2^{2pl}}$$

where Z_{NV}^l is defined by (1.28).

1.3 Practical procedure

Now we present the practical procedure used to implement the multilevel estimators. Putting together the elements already discussed, the algorithm that we used for the multilevel Monte Carlo with the Ninomiya-Victoir scheme is as follows. We begin by estimating the weak error constant c_1 in (1.8), the constant c_2 which comes from the variance estimation (1.9) and checking the orders of weak and strong convergence. When the asymptotic behavior (1.8) of the bias of the scheme is satisfied, one has

$$\mathbb{E}\left[Z_{NV}^l\right] \sim \frac{c_1\left(1 - 2^{\alpha}\right)}{2^{\alpha l}}. \tag{1.29}$$

Using a regression with few values of $\left(l, \left|\mathbb{E}\left[Z_{NV}^l\right]\right|\right)$, we estimate c_1 and check the order α of weak convergence. In the same way, we estimate c_2 and check the strong order β of variance convergence to 0, using a regression in (1.9). Then we estimate $\mathbb{V}\left(Z_{NV}^0\right)$ using a standard Monte Carlo estimator $\hat{V}_0$. After that, for a given ϵ we define L^* using (1.10) then we set

$$M_0^* = \left\lceil \frac{2}{\epsilon^2}\sqrt{\frac{\hat{V}^0}{\lambda_0}}\left(\sqrt{\lambda_0\hat{V}^0} + \sum_{j=1}^{L^*}\sqrt{c_2\lambda_j 2^{j(1-\beta)}}\right)\right\rceil \tag{1.30}$$

and

$$\forall l \in \{1, \dots, L^*\}, M_l^* = \left\lceil \frac{2}{\epsilon^2}\sqrt{\frac{c_2}{\lambda_l 2^{l(\beta+1)}}}\left(\sqrt{\lambda_0\hat{V}^0} + \sum_{j=1}^{L^*}\sqrt{c_2\lambda_j 2^{j(1-\beta)}}\right)\right\rceil. \tag{1.31}$$

With regards to the wights $(\lambda_l)_{0 \leq l \leq L^*}$, a reasonable choice is to take $\lambda_0 = 1$ and $\forall l \in \{1, \dots, L^*\}, \lambda_l = 5$. Finally, we compute the multilevel Monte Carlo estimator with the Ninomiya-Victoir scheme

$$\hat{Y}_{MLMC}^{NV} = \sum_{l=0}^{L^*}\frac{1}{M_l^*}\sum_{k=1}^{M_l^*} Z_{NV}^{l,k}.$$

1.4 Application to the Heston model

The Heston model is an asset price model which assumes that the implied volatility, denoted by V, evolves according to an autonomous Cox-Ingersoll-Ross SDE:

$$\begin{cases} dU_t = (r - \delta - \frac{1}{2}V_t)dt + \sqrt{V_t}dW_t^1 \\ dV_t = \kappa(\theta - V_t)dt + \sigma\sqrt{V_t}\left(\rho dW_t^1 + \sqrt{1-\rho^2}dW_t^2\right), \end{cases} \tag{1.32}$$

where

- $\theta \in \mathbb{R}_+^*$ is the long implied variance, or long run average price variance; as t tends to infinity, the expected value of V_t tends to θ,
- $\kappa \in \mathbb{R}_+^*$ is the rate at which V_t reverts to θ,
- $\sigma \in \mathbb{R}_+^*$ is the volatility of the implied volatility and determines the variance of V_t,
- $r \in \mathbb{R}$ the annualized risk-free interest rate, continuously compounded,
- $\delta \in \mathbb{R}_+^*$ is the annualized continuous yield dividend,
- $\rho \in]-1, 1[$ is the correlation between the two Brownian motion (ie stock price and implied volatility).

The asset price S is given by $S_t = \exp(U_t)$. To ensure that the zero boundary is not attainable for the volatility process, one has to assume that $2\kappa\theta \geq \sigma^2$. The main difficulty is located in 0, where the square root is not Lipschitz. In this $2-$dimensional model, the Brownian vector fields are given by $\sigma^1 \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \sqrt{v} \\ \rho\sigma\sqrt{v} \end{pmatrix}$, $\sigma^2 \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} 0 \\ \sigma\sqrt{1-\rho^2}\sqrt{v} \end{pmatrix}$ and the drift coefficient is $b \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} r - \delta - \frac{1}{2}v \\ \kappa(\theta - v) \end{pmatrix}$. The Stratonovich drift is given by $\sigma^0 = b - \frac{1}{2}\left(\partial\sigma^1\sigma^1 + \partial\sigma^2\sigma^2\right)$:

$$\begin{aligned}
\sigma^0 \begin{pmatrix} u \\ v \end{pmatrix} &= \begin{pmatrix} r - \delta - \frac{1}{2}v \\ \kappa(\theta - v) \end{pmatrix} - \frac{1}{2}\left(\begin{pmatrix} 0 & \frac{1}{2\sqrt{v}} \\ 0 & \frac{\rho\sigma}{2\sqrt{v}} \end{pmatrix} \begin{pmatrix} \sqrt{v} \\ \rho\sigma\sqrt{v} \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & \frac{\sigma\sqrt{1-\rho^2}}{2\sqrt{v}} \end{pmatrix} \begin{pmatrix} 0 \\ \sigma\sqrt{1-\rho^2}\sqrt{v} \end{pmatrix} \right) \\
&= \begin{pmatrix} r - \delta - \frac{1}{2}v - \frac{1}{4}\rho\sigma \\ \kappa(\theta - v) - \frac{\sigma^2}{4} \end{pmatrix}.
\end{aligned}$$

Therefore, setting $\xi = \theta - \frac{\sigma^2}{4\kappa}$, the Ninomiya-Victoir scheme, for a given uniform grid, is inductively defined by:

1st step:

$$\bar{U}^0_{t_{k+1}} = U^{NV,\eta}_{t_k} + \frac{1}{2}\left(r - \delta - \frac{1}{2}\rho\sigma - \frac{1}{2}\xi\right)t_1 + \frac{1}{2\kappa}(v - \xi)\left(\exp\left(-\frac{1}{2}\kappa t_1\right) - 1\right),$$

$$\bar{V}^0_{t_{k+1}} = \left(V^{NV,\eta}_{t_k} - \xi\right)\left(\exp\left(-\frac{1}{2}\kappa t_1\right) - 1\right) + \xi.$$

2nd step:
If $\eta_{k+1} = 1$:

$$\bar{U}^{1,\eta}_{t_{k+1}} = \bar{U}^0_{t_{k+1}} + \sqrt{\bar{V}^0_{t_{k+1}}}\Delta W^1_{t_{k+1}} + \frac{1}{4}\rho\sigma\left(\Delta W^1_{t_{k+1}}\right)^2,$$

$$\bar{V}^{1,\eta}_{t_{k+1}} = \left(\sqrt{\bar{V}^0_{t_{k+1}}} + \frac{1}{2}\sigma\rho\Delta W^1_{t_{k+1}}\right)^2,$$

$$\bar{U}^{2,\eta}_{t_{k+1}} = \bar{U}^{1,\eta}_{t_{k+1}},$$

$$\bar{V}^{2,\eta}_{t_{k+1}} = \left(\sqrt{\bar{V}^{1,\eta}_{t_{k+1}}} + \frac{1}{2}\sigma\sqrt{1-\rho^2}\Delta W^2_{t_{k+1}}\right)^2.$$

If $\eta_{k+1} = -1$:

$$\bar{U}^{1,\eta}_{t_{k+1}} = \bar{U}^{0,\eta}_{t_{k+1}},$$

$$\bar{V}^{1,\eta}_{t_{k+1}} = \left(\sqrt{\bar{V}^{0}_{t_{k+1}}} + \frac{1}{2}\sigma\sqrt{1-\rho^2}\Delta W^{2}_{t_{k+1}}\right)^2,$$

$$\bar{U}^{2,\eta}_{t_{k+1}} = \bar{U}^{1,\eta}_{t_{k+1}} + \sqrt{\bar{V}^{1,\eta}_{t_{k+1}}}\Delta W^{1}_{t_{k+1}} + \frac{1}{4}\rho\sigma\left(\Delta W^{1}_{t_{k+1}}\right)^2,$$

$$\bar{V}^{2,\eta}_{t_{k+1}} = \left(\sqrt{\bar{V}^{1,\eta}_{t_{k+1}}} + \frac{1}{2}\sigma\rho\Delta W^{1}_{t_{k+1}}\right)^2.$$

3rd step:

$$U^{NV,\eta}_{t_{k+1}} = \bar{U}^{2,\eta}_{t_{k+1}} + \frac{1}{2}\left(r-\delta-\frac{1}{2}\rho\sigma-\frac{1}{2}\xi\right)t_1 + \frac{1}{2\kappa}(v-\xi)\left(\exp\left(-\frac{1}{2}\kappa t_1\right)-1\right),$$

$$V^{NV,\eta}_{t_{k+1}} = \left(\bar{V}^{2,\eta}_{t_{k+1}}-\xi\right)\left(\exp\left(-\frac{1}{2}\kappa t_1\right)-1\right)+\xi.$$

Assuming $\xi \geq 0$, the Ninomiya-Victoir scheme is well defined and the volatility process is always positive (see [1]), whereas the Giles-Szpruch scheme and usual schemes such as the Euler scheme are not well defined since they can lead to negative values of the volatility process for which the square root is not defined at the next step. That is why it is preferable to use the Ninomiya-Victoir scheme to discretize the Heston model. In addition, for $\xi < 0$, in section 3.1 of [1], Alfonsi proposed a modification of the Ninomiya-Victoir scheme preserving the positivity of the volatility and the weak order two.

2 Code in C/C++

In this section, we provide a documentation for our code written in C/C++. The aim is to price an European put option , in the Heston model, $\mathbb{E}\left[\exp\left(-rT\right)\left(K-S_T\right)_+\right]$, for a given maturity T and strike K, using our multilevel Monte Carlo technique.

The routine **void Brownian(double* W,int l,double T)** simulates Brownian increments on a grid, with time step $h_l = T/2^l$. Arguments:

- the pointer double* W is an array with size 2^l,
- the integer int l corresponds to the level of the grid,
- the real double T corresponds to the final time.

It writes the result in the array W.

The routine **void Antithetic_Brownian(double* W,int l)** swaps the two consecutive Brownian increments of the already existing Brownian increments W on a grid, with time step $h_l = T/2^l$. Arguments:

- the pointer double* W is an array with size 2^l, which contains Brownian increments,
- the integer int l corresponds to the level of the grid.

It writes the result in the array W.

The routine **void Coarse_Brownian(double* W,int l)** sums each consecutive Brownian increments of the already existing Brownian increments W on a grid, with time step $h_l = T/2^l$. Arguments:

- the pointer double* W is an array with size 2^l, which contains Brownian increments,
- the integer int l corresponds to the level of the grid.

It writes the result in the sub-array $\Big(W[0], \ldots, W[2^{l-1} - 1]\Big)$.

The routine **void Bernoulli(bool* Eta,int l)** simulates independent Bernoulli random variables on a grid with 2^l steps. Arguments:

- the pointer bool* Eta is an array with size 2^l,
- the integer int l corresponds to the level of the grid.

It writes the result in the array Eta.

The routine **void Coarse_Bernoulli(bool* Eta,int l)** extracts, from the array Eta, with size 2^l, the sub-array $\Big(Eta[0], Eta[2] \ldots, Eta[2^{l-1}]\Big)$. Arguments:

- the pointer bool* Eta is an array with size 2^l,
- the integer int l corresponds to the level of the grid.

It writes the result in the sub-array $\Big(Eta[0], \ldots, Eta[2^{l-1} - 1]\Big)$.

The routine **void Antithetic_Bernoulli(bool* Eta,int l)** flips each component of the boolean array Eta, with size 2^l. Arguments:

- the pointer bool* Eta is an array with size 2^l,
- the integer int l corresponds to the level of the grid.

It routine writes the result in the array Eta.

In the following:

- the real double T corresponds to the maturity, in years, of the European put option,
- the real double K corresponds the strike, of the European put option,
- the real double theta corresponds to the parameter θ, in the Heston model,
- the real double kappa corresponds to the parameter κ, in the Heston model,
- the real double sigma corresponds to the parameter σ, in the Heston model,
- the real double r corresponds to the parameter r, in the Heston model,
- the real double delta corresponds to the parameter δ, in the Heston model,
- the real double rho corresponds to the parameter ρ, in the Heston model,

and we assume that $\theta \geq \frac{\sigma^2}{4\kappa}$.

The routines **void Fine_NV_Scheme**, **void Antithetic_NV_Scheme** and **void Coarse_NV_Scheme** simulate, respectively the Ninomiya-Victoir schemes, $X_T^{NV,2^l,\eta^{2^l}}$, $\tilde{X}_T^{NV,2^l,\eta^{2^l}}$ and $X_T^{NV,2^{l-1},\eta^{2^l}}$ in the framework of the Heston model. The three routines takes the same arguments:

double* W1,double* W2,bool* Eta,int l,double T,double &x,double &v,double kappa,double theta,double sigma,double r,double delta, double rho.

- The pointer double* W1 is an array with size 2^l, which contains Brownian increments on a grid with time step $h_l = T/2^l$.
- the pointer double* W2 is an array with size 2^l, which contains Brownian increments on a grid with time step $h_l = T/2^l$ and independent of the ones in W1.
- the pointer bool* Eta is an array with size 2^l, which contains independent Bernoulli random variables.
- the integer int l corresponds to the level of the grid,
- the real double &x corresponds to the logarithm of the initial value of the stock price at time 0, ie $\log(S_0)$,
- the real double &v corresponds to the initial value of the implied volatility at time 0.

The results, which are $\log\left(S_T^{NV}\right)$ and V_T^{NV} are respectively placed in x and v.

The routines **double Put(double x,double K,double T,double r)** computes exercise value of an European put option. Arguments:

- the double x corresponds to the logarithm of the stock price, ie $x = log(S_T)$.

It returns exercise value, ie $f(S_T) = \exp(-rT)(K - S_T)_+$.

In the following:

- the real double x corresponds to the logarithm of the initial value of the stock price at time 0, ie $\log(S_0)$,
- the real double v corresponds to the initial value of the implied volatility at time 0,
- the real double epsilon corresponds to the target error ϵ.

The routines **Crude_Monte_Carlo_NV** and , **Antithetic_Monte_Carlo_NV** compute, using a standard Monte Carlo method, respectively the expectations $\mathbb{E}\left[f\left(S_T^{NV,2^l,\eta^{2^l}}\right)\right]$ and $\mathbb{E}\left[Z_{NV}^l\right]$ together with their associated variance. Here f denotes the payoff of a put option and Z_{NV}^l is defined by (1.28). Both routines write the results in the variable mean and the variable variance (see below), and take the same arguments:

(double &mean,double &variance,int l,long M,double K,double T,double x,double v,double kappa,double theta,double sigma,double r,double delta, double rho).

- the integer int l corresponds to the level of the grid,
- the integer long M corresponds to the sample size of the Monte Carlo method,

The routine **Intermediate_Step(double &c1,double &c2,double &alpha,double &beta,int Lmax,long M,double K,double T,double x, double v,double kappa, double theta, double sigma, double r,double delta, double rho)** estimates the parameters c_1 ,c_2, α and β as described in section 1.3. It writes the results in the variables c1, c2, alpha, beta respectively. Arguments:

- the integer int Lmax corresponds to the number of levels used to compute the regression (we advise to take Lmax = 4),
- the integer long M corresponds to the sample size of the Monte Carlo method (we advise to take M between 10^4 and 10^5).

The routine **Last_Level(int &L,double epsilon,double c1,double alpha)** computes the optimal finest level of discretization L^* defined by (1.10). It writes the result in the variable L. Arguments:

- the real double epsilon corresponds to the target error of the multilevel Monte Carlo method,
- the real double c1 corresponds to the previous estimation of the constant c_1,
- the real double alpha corresponds to the previous estimation of the constant α.

The routine **Optimal_Parameters(long* Ml,int L,double epsilon,double c2,double beta,double Var0)** computes the optimal sample size $(M_l^*)_{0\leq l\leq L^*}$ defined by (1.30) and (1.31). Arguments:

- The pointer long* Ml is an array with size $L+1$, which will contain, at the end of the procedure, the optimal sample size.
- the integer int L corresponds to the previous optimal finest level of discretization,
- the real double epsilon corresponds to the target error of the multilevel Monte Carlo method,
- the real double c2 corresponds to the previous estimation of the constant c_2,
- the real double beta corresponds to the previous estimation of the constant β,
- the real double Var0 corresponds to an estimation of the variance $\mathbb{V}\left[Z_{NV}^0\right]$.

The routine **Antithetic_Multi_Level_NV(double &price_MC,double &variance_MC,double K,double T,double S, double v,double kappa, double theta, double sigma, double r,double delta, double rho,double epsilon,int Lmax=4, long M=100000)** implement the multilevel Monte Carlo method, for the pricing of an European put option. It writes the result in the double price_MC,double, together with its associated variance in variance_MC. The arguments Lmax and long M correspond to the ones in the routine **Intermediate_Step**.

References

[1] A. Alfonsi. Affine Diffusions and Related Processes: Simulation, Theory and Applications, ISBN-13: 9783319052205, 2015. 8

[2] A.Al Gerbi, B. Jourdain, E. Clément, Ninomiya-Victoir scheme: strong convergence, antithetic version and application to multilevel estimators, arXiv:1508.06492v2, 2015. 1, 3, 5

[3] M.B. Giles. Multi-level Monte Carlo path simulation. Operations Research, 56(3):607-617, 2008. 1, 2

[4] M.B. Giles, L. Szpruch. Antithetic multilevel Monte Carlo estimation for multi-dimensional SDEs without Lévy area simulation, Annals of Applied Probability, 24(4):1585-1620, 2014. 3

[5] S. Ninomiya, N. Victoir. Weak approximation of stochastic differential equations and application to derivative pricing. Applied Mathematical Finance 15, 107-121, 2008. 3