Source | Model | Option
| Model_Option | Help on mc methods | Archived Tests

mc_fps

Input parameters

- Number of iterations N
- Generator type
- Increment inc
- Confidence Value
- Volatility of volatility

Output parameters

- Price P
- Error price σ_P
- Delta δ
- Error delta σ_{delta}
- Price Confidence Interval: ICp [Inf Price,Sup Price]
- Delta Confidence Interval: ICp [Inf Delta,Sup Delta]

Description

Le but est de donner le prix et le delta d'un put européen sous le modèle à volatilité stochastique suivant [?]:

$$\begin{cases} dX_t = rX_t\,dt + \sigma_t X_t\,dW_t \\ \sigma_t = f(Y_t) \\ dY_t = -\alpha Y_t\,dt + \beta\,dZ_t \end{cases}$$

Ainsi dans ce modèle, le prix du sous-jacent et la volatilité sont des variables stochastiques.

Dans le système ci-dessus,

- X représente le sous-jacent, X_t son cours à la date t,
- r est le taux instantané, supposé constant,
- σ_t est la valeur à la date t de la volatilité du cours du sous-jacent; c'est un processus stochastique, fonction déterministe du processus Y ; la fonction f est définie sur $\mathbb{R}$ et à valeurs strictement positives,
- Y est un processus d'Ornstein-Uhlenbeck, de moyenne à long terme nulle et de variance à long terme $\nu^2 = \frac{\beta^2}{2\alpha}$,
- W et Z sont deux mouvements browniens corrélés, avec une corrélation constante $\rho \in]-1, 1[$, telle que $d\langle W, Z\rangle_t = \rho\, dt$. Nous prendrons deux mouvements browniens indépendants $(W_t^1)_{t\geq 0}$ et $(W_t^2)_{t\geq 0}$ de sorte que le modèle puisse être décrit par l'équation différentielle stochastique suivante :

$$\left\{\begin{array}{c} dX_t = rX_t\, dt + \sigma_t X_t \left(\rho\, dW_t^2 + \sqrt{1-\rho^2} dW_t^1\right), \qquad X_0 = x \\ \sigma_t = f(Y_t) \\ dY_t = -\alpha Y_t\ dt + \beta\, dW_t^2, \qquad Y_0 = y, \end{array}\right.$$

Pour résoudre le système ci-dessus, on exploitera plusieurs propriétés:

1. La dynamique de Y est autonome, i.e. elle ne dépend pas de la dynamique de X; on peut donc simuler d'abord Y, puis X sachant Y.

2. On peut simuler Y exactement, car Y est un processus gaussien. Cependant, ayant simulé Y uniquement, il n'est pas possible de simuler le processus X. Il est nécessaire de simuler le couple (Y,W^2), qui est également un processus gaussien.

3. Afin de simuler la loi de (Y,W^2) en limitant le nombre de calculs, on exploitera le caractère markovien de ce processus.

4. On en déduira l'expression de X_T, puisqu'on connait son expression en fonction de Y, W^1 et W^2:

$$X_T = xe^{rT}e^{\rho\int_0^T f(Y_t)\,dW_t^2+\sqrt{1-\rho^2}\int_0^T f(Y_t)\,dW_t^1-\frac{1}{2}\int_0^T f(Y_t)^2dt}.$$

1 Simulation du couple

Pour simuler la loi du couple (Y,W^2), on utilise le résultat suivant:

- Soit $M \in \mathbb{N}$, $M \geq 1$, et $0 < t_1 < t_2 < \cdots < t_M = T$.

  Alors le vecteur $V = \left(Y_{t_1}, W_{t_1}^2, Y_{t_2}, W_{t_2}^2, Y_{t_3}, W_{t_3}^2, \ldots, Y_{t_M}, W_{t_M}^2\right)$ est un vecteur gaussien.

  Preuve: Soit $(\lambda_i)_{1\leq i\leq M} \in \mathbb{R}^M$ et $(\mu_i)_{1\leq i\leq M} \in \mathbb{R}^M$. Il s'agit de montrer que la variable aléatoire réelle

$$G = \sum_{i=1}^{M}\left(\lambda_i Y_{t_i} + \mu_i W_{t_i}^2\right)$$

  suit une loi gaussienne. Pour ce faire, posons $R_t = e^{\alpha t}Y_t$; comme $dR_t = \beta e^{\alpha t}dW_t^2$,

$$R_t = y + \beta\int_0^t e^{\alpha s}dW_s^2.$$

  Il est alors facile de trouver des réels c, λ_i' et μ_i' tels que

$$\begin{aligned} G &= c + \sum_{i=1}^{M}\left(\lambda_i'\int_{t_{i-1}}^{t_i} e^{\alpha s}dW_s^2 + \mu_i'\left(W_{t_i}^2 - W_{t_{i-1}}^2\right)\right) \\ &= c + \sum_{i=1}^{M}\int_{t_{i-1}}^{t_i}\left(\lambda_i' e^{\alpha s} + \mu_i'\right)dW_s^2. \end{aligned}$$

Or, pour tout i, $I_i = \int_{t_{i-1}}^{t_i} (\lambda_i' e^{\alpha s} + \mu_i')\, dW_s^2$ est une variable gaussienne - car limite dans L^2 d'une suite de variables gaussiennes ; de plus les I_i sont indépendantes. G est donc la somme de v.a. gaussiennes indépendantes, c'est donc elle-même une gaussienne.

Méthode 1 On génère la matrice de variance-covariance Γ du vecteur gaussien V, son vecteur des moyennes m, on calcule une racine carrée A de Γ par la méthode de Cholesky, et on génère un vecteur G de $2M$ variables aléatoires gaussiennes centrées réduites indépendantes. Alors le vecteur $AG + m$ a même loi que V.

Pour calculer Γ, il suffit de connaître, pour $1 \leq i \leq j \leq M$:

$$\begin{aligned}
\mathrm{cov}(Y_{t_i}, Y_{t_j}) &= \nu^2 \left(e^{-\alpha(t_i+t_j)} - e^{-\alpha(t_j-t_i)}\right), \\
\mathrm{cov}(W_{t_i}^2, W_{t_j}^2) &= t_i, \\
\mathrm{cov}(Y_{t_i}, W_{t_j}^2) &= \frac{\beta}{\alpha}\left(1 - e^{-\alpha t_i}\right), \\
\mathrm{cov}(Y_{t_j}, W_{t_i}^2) &= e^{-\alpha(t_j-t_i)}\mathrm{cov}(Y_{t_i}, W_{t_j}^2).
\end{aligned}$$

Pour calculer m, il suffit de connaître

$$\begin{aligned}
\mathbb{E}^*\left[Y_{t_i}\right] &= y e^{-\alpha t_i}, \\
\mathbb{E}^*\left[W_{t_i}^2\right] &= 0.
\end{aligned}$$

L'inconvénient de cette méthode est que l'on utilise des matrices de taille $2M$.

Méthode 2 Ici on exploite le caractère markovien - *en plus* du caractère gaussien - du couple $(Y_t, W_t^2)_{0 \leq t \leq T}$. Pour cela, on se donne une subdivision régulière $0 = t_0 < t_1 < t_2 < \cdots < t_M = T$ de $[0, T]$ de pas Δt et on remarque que pour $1 \leq n \leq M$,

$$\begin{aligned}
Y_{t_n} &= e^{-\alpha \Delta t}\left(Y_{t_{n-1}} + \beta e^{-\alpha t_{n-1}} \int_{t_{n-1}}^{t_n} e^{\alpha s} dW_s^2\right), \\
W_{t_n}^2 &= W_{t_{n-1}}^2 + \left(W_{t_n}^2 - W_{t_{n-1}}^2\right),
\end{aligned}$$

si bien que si l'on pose :

$$\begin{aligned}
V_n &= (Y_{t_n}, W^2_{t_n})^t \\
U_n &= \left(\beta e^{-\alpha t_{n-1}} \int_{t_{n-1}}^{t_n} e^{\alpha s} dW^2_s,\ W^2_{t_n} - W^2_{t_{n-1}}\right)^t \\
g\left(\begin{pmatrix} y \\ w \end{pmatrix}, \begin{pmatrix} u^1 \\ u^2 \end{pmatrix}\right) &= \begin{pmatrix} e^{-\alpha \Delta t}\left(y + u^1\right) \\ w + u^2 \end{pmatrix}
\end{aligned}$$

on a alors pour pour $n \geq 1$

$$\begin{aligned}
g(V_{n-1}, U_n) &= \begin{pmatrix} e^{-\alpha \Delta t}\left(Y_{t_{n-1}} + \beta e^{-\alpha t_{n-1}} \int_{t_{n-1}}^{t_n} e^{\alpha s} dW^2_s\right) \\ W^2_{t_{n-1}} + W^2_{t_n} - W^2_{t_{n-1}} \end{pmatrix}. \\
&= V_n.
\end{aligned}$$

Or on note que $(U_n)_{1 \leq n \leq M}$ est une suite de vecteurs aléatoires gaussiens indépendants de dimension 2 de même loi $\mathcal{N}_2(0, \Gamma)$, où

$$\Gamma = \begin{pmatrix} \nu^2 \left(e^{2\alpha \Delta t} - 1\right) & \frac{\beta}{\alpha}\left(e^{\alpha \Delta t} - 1\right) \\ \frac{\beta}{\alpha}\left(e^{\alpha \Delta t} - 1\right) & \Delta t \end{pmatrix},$$

et indépendante de V_0. Par conséquent, $(V_n)_{0 \leq n \leq M}$ est une chaîne de Markov qu'on peut simuler facilement en générant M lois $\mathcal{N}_2(0, \Gamma)$ indépendantes. Pour ce faire, on applique la méthode 1 - mais *en dimension 2* seulement. Cette méthode est plus rapide, car la méthode de Cholesky en dimension d nécessite de l'ordre de d^3 opérations.

2 Simulation de X

On rappelle que

$$X_T = x e^{rT} \exp\left(\rho \int_0^T f(Y_t)\, dW^2_t + \sqrt{1 - \rho^2} \int_0^T f(Y_t)\, dW^1_t - \frac{1}{2} \int_0^T f(Y_t)^2 dt\right)$$

et que l'on a déjà simulé le couple (Y, W^2) aux dates $t_n = \frac{nT}{M}$. On décide d'approximer respectivement $I_0 = \int_0^T f(Y_t)^2 dt$, $I_1 = \int_0^T f(Y_t)\, dW^1_t$ et $I_2 =$

$\int_0^T f(Y_t)\, dW_t^2$ par

$$\begin{aligned} I_0^{\Delta t} &= \Delta t \sum_{i=0}^{M-1} f(Y_{t_i})^2, \\ I_1^{\Delta t} &= \sum_{i=0}^{M-1} f(Y_{t_i}) \left(W_{t_{i+1}}^1 - W_{t_i}^1\right), \\ I_2^{\Delta t} &= \sum_{i=0}^{M-1} f(Y_{t_i}) \left(W_{t_{i+1}}^2 - W_{t_i}^2\right). \end{aligned}$$

Les valeurs des Y_{t_i} et $W_{t_i}^2$ sont connues, il suffit donc à cette étape de générer une suite $W_{t_{i+1}}^1 - W_{t_i}^1$ de gaussiennes indépendantes centrées et de variance Δt. On approxime alors X_T par

$$\overline{X}_T^{\Delta t} = xe^{rT} \exp\left(\rho I_2^{\Delta t} + \sqrt{1-\rho^2} I_1^{\Delta t} - \frac{1}{2} I_0^{\Delta t}\right).$$

3 Méthode de Monte-Carlo et extrapolation de Romberg (fonction *principal*)

Pour une valeur Δt du pas de temps, on a ainsi obtenu une expression explicite de $\overline{X}_T^{\Delta t}$, qui est une approximation de X_T. On note h la fonction représentant le payoff. Dans notre cas, il s'agit de

$$h : x \longrightarrow h(x) = (K - x)_+ ,$$

où K représente la valeur du strike, et $(.)_+ = \max(., 0)$.

On souhaite calculer la valeur en $t = 0$ de l'option dont le payoff est $h(X_T)$. On l'approxime alors $\mathbb{E}[h(X_T)]$ par

$$\frac{1}{N} \sum_{i=1}^{N} h(\overline{X}_T^{(i)}),$$

où N est le nombre de simulations.

Donc la valeur en $t = 0$ est approximéé par:

$$e^{-rT} \frac{1}{N} \sum_{i=1}^{N} h(\overline{X}_T^{(i)}).$$

3.1 Erreur statistique

D'après le théorème de la limite centrale,

$$\frac{\sqrt{N}}{\sigma}\left(\frac{1}{N}\sum_{i=1}^{N} h(\overline{X}_T^{(i)}) - \mathbb{E}\left[h\left(\overline{X}_T\right)\right]\right) \underset{N\longrightarrow+\infty}{\overset{en\ loi}{\longrightarrow}} \mathcal{N}(0,1),$$

où $\sigma^2 = \mathrm{Var}\left[\overline{X}_T\right]$.

L'erreur statistique est alors donnée par

$$\frac{\sigma}{\sqrt{N}}.$$

On réduira la variance grâce à l'utilisation de la méthode des variables antithétiques, qui permet de réduire le nombre d'appel au générateur de lois uniformes.

3.2 Erreur temporelle

Il faut également considérer l'erreur de discrétisation temporelle

$$\left|\mathbb{E}\left[h\left(\overline{X}_T^{\Delta t}\right)\right] - \mathbb{E}\left[h\left(X_T\right)\right]\right|,$$

d'ordre Δt.

Pour la suite, on admettra le résultat suivant:

$$\mathbb{E}\left[h\left(\overline{X}_T^{\Delta t}\right)\right] - \mathbb{E}\left[h\left(X_T\right)\right] = C\Delta t + O\left(\Delta t^2\right),$$

où C est une constante indépendante de Δt.

3.3 Méthode d'extrapolation de Romberg

Si l'on a bien le développement précédent, alors on peut mettre en oeuvre la méthode d'extrapolation de Romberg : on estime $\mathbb{E}\left[h\left(\overline{X}_T^{\Delta t}\right)\right]$ pour deux valeurs Δt et $\Delta t/2$ du pas de temps, et on en déduit une estimation de

$$Z_T^{\Delta t} = 2\mathbb{E}\left[h\left(\overline{X}_T^{\Delta t/2}\right)\right] - \mathbb{E}\left[h\left(\overline{X}_T^{\Delta t}\right)\right].$$

$Z_T^{\Delta t}$ est une approximation d'ordre 2 en temps de $\mathbb{E}\left[h\left(X_T\right)\right]$ car:

$$\begin{aligned}
Z_T^{\Delta t} - \mathbb{E}\left[h\left(X_T\right)\right] &= 2\mathbb{E}\left[h\left(\overline{X}_T^{\Delta t/2}\right)\right] - 2\mathbb{E}\left[h\left(\overline{X}_T^{\Delta t}\right)\right] - \mathbb{E}\left[h\left(X_T\right)\right] \\
&= \underbrace{2\left(\mathbb{E}\left[h\left(\overline{X}_T^{\Delta t}\right)\right] - \mathbb{E}\left[h\left(X_T\right)\right]\right)} - \underbrace{\left(\mathbb{E}\left[h\left(\overline{X}_T^{\Delta t}\right)\right] - \mathbb{E}\left[h\left(X_T\right)\right]\right)} \\
&= 2C\frac{\Delta t}{2} - C\Delta t + O(\\
&= O\left(\Delta t^2\right).
\end{aligned}$$

Pour mesurer numériquement cette erreur, il est essentiel de pouvoir estimer dans le code informatique la constante en facteur du Δt^2. Pour ce faire, *supposons* qu'on dispose d'un développement limité

$$\mathbb{E}\left[h\left(\overline{X}_T^{\Delta t}\right)\right] - \mathbb{E}\left[h\left(X_T\right)\right] = C_1'\Delta t + C_2'\Delta t^2 + o\left(\Delta t^2\right).$$

Alors il existe C_2 tel que

$$Z_T^{\Delta t} - \mathbb{E}\left[h\left(X_T\right)\right] = C_2\Delta t^2 + o\left(\Delta t^2\right),$$

et

$$Z_T^{\Delta t} - Z_T^{\Delta t/2} = \frac{3}{4}C_2\Delta t^2 + o\left(\Delta t^2\right).$$

Ceci permet d'estimer *a posteriori* l'erreur due à la discrétisation temporelle $C_2\Delta t^2$ par $C_2^{\Delta t}\Delta t^2 = \frac{4}{3}\left(Z_T^{\Delta t} - Z_T^{\Delta t/2}\right)$.

On souhaite obtenir une erreur temporelle de l'ordre de l'erreur statistique. Pour ce faire, on commence avec un pas de temps $\Delta t = T$, i.e. $M = 1$. On divise Δt par 2 (en pratique dans le programme cela revient à multiplier le nombre de pas de temps M par 2),et on recalcule cette erreur temporelle. On s'arrête lorsqu'elle se situe dans l'intervalle $\left[\pm\frac{3\epsilon_s^{\Delta t}}{2}\right]$, où $\epsilon_s^{\Delta t}$ est l'erreur statistique calculée grâce à la méthode de Monte-Carlo.

Au final, on a donc un pas de temps Δt et on a confiance en l'estimation

$$Z_N^{\Delta t} \pm 2\epsilon_s^{\Delta t}$$

pour le put.

4 Détail du programme

On donne ici la correspondance entre les variables que l'on utlise pour notre modèle et celles effectivement utilisées dans le code du programme:

N	*double N*
x	*double X*
r	*double r*
σ_f	*double sigmaf*
Y_0	*double Y0*
α	*double alpha*
ν	*double nu*
ρ	*double rho*
T	*double T*
M	*double M*

Dans ce qui suit, on décrit les principales fonctions.

- fonction *V*. la valeur retournée est de type *static double ***. Il s'agit du vecteur
$$V = \left(Y, W^2\right).$$
- fonction *XT*. la valeur retournée est de type *static double **. C'est un tableau à deux cases. La première contient $\overline{X}_T^{\Delta t}$, et la deuxième $\overline{X}_T^{\frac{\Delta t}{2}}$.
- fonction *prix_es_delta*. la valeur retournée est de type *static int*. Danc cette fonction on écrit dans les pointeurs suivants:

  **p1* contient le prix du put.

  **delta1* contient le delta.

  **error_price1* contient l'erreur comise en calculant le prix avec notre méthode.

  **error_delta1* contient l'erreur comise en calculant le delta avec notre méthode.

References