

SIMULATIONS DES VARIABLES ALEATOIRES

 Ibrahima Dione (Ph.D.) & Van Son Lai (Ph.D., CFA)

 Simulations Stochastiques et Applications en Finance

- Génération d'une Variable Uniforme
- Génération des Variables Aléatoires Discrètes
- Simulation de Variables Aléatoires Continues
- Simulation de Vecteurs Aléatoires
- Méthode de Génération par Rejet
- Méthode Monte Carlo et Chaînes de Markov (MCMC)

Génération d'une Variable Uniforme

- ▷ Dans MATLAB, pour générer une variable aléatoire uniforme entre $[0, 1]$ notée $U(0, 1)$, on utilise la commande *rand*.
- ▷ La commande *rand(m, n)* génère $m \times n$ variables aléatoires uniformément reparties entre $[0, 1]$ et rangées sous forme d'une matrice de m lignes et n colonnes.

Note: Une variable aléatoire uniformément repartie entre $[a, b]$ peut être générée à partir de la variable $U(0, 1)$ via la relation

$$U(a, b) = (b - a)U(0, 1) + a \quad (1)$$

Génération des Variables Aléatoires Discrètes

- ▷ Soit la variable aléatoire X dont l'espace des épreuves est l'ensemble fini $\{x_1, x_2, \dots, x_n\}$ ayant les probabilités respectives p_1, p_2, \dots, p_n . On peut simuler X avec une variable uniforme $U(0, 1)$ en posant

$$X = x_1 \text{ si } U(0, 1) \leq p_1$$

$$X = x_2 \text{ si } p_1 < U(0, 1) \leq p_1 + p_2$$

⋮

$$X = x_n \text{ si } p_1 + \dots + p_{n-1} < U(0, 1) \leq p_1 + \dots + p_n = 1$$

(2)

- ▷ Cette méthode revient simplement à subdiviser l'intervalle $[0, 1]$ en n sous-intervalles dont les longueurs correspondent aux p_1, p_2, \dots, p_n .

Définition

Une **variable aléatoire binomiale** $X = B(N, p)$ est la somme de N variables aléatoires binaires B_i telles que $X = B_1 + B_2 + \dots + B_n$ où

$$P(B_i = 0) = 1 - p \text{ et } P(B_i = 1) = p$$

- ▷ Lorsque N est relativement petit (par exemple inférieur à 30), on peut simuler $B(N, p)$ en générant N variables binaires B_i dont chacune est obtenue à partir d'une variable uniforme $U(0, 1)$.



- Soit une variable aléatoire de **Poisson X** de paramètre λ , c'est à dire

$$P(X = k) = \frac{e^{-\lambda} \lambda^k}{k!}, \quad k = 0, 1, 2, \dots, +\infty \quad (3)$$

- Pour simuler la variable X , on peut utiliser la technique précédente:

- ★ En générant une suite de variables aléatoires indépendantes U_i , uniformément reparties sur $[0, 1]$;
- ★ Et en définissant la variable de Poisson X comme suit (où $p = e^{-\lambda}$)

$$\begin{aligned} U_1 < p &\longrightarrow X = 0 \\ \left. \begin{array}{ll} U_1 & \geq p \\ U_1 U_2 & < p \end{array} \right\} &\longrightarrow X = 1 \\ &\vdots \\ &\left. \begin{array}{ll} U_1 & \geq p \\ U_1 U_2 & \geq p \\ \vdots & \\ U_1 U_2 \cdots U_k & \geq p \\ U_1 U_2 \cdots U_{k+1} & < p \end{array} \right\} \longrightarrow X = k \end{aligned} \quad (4)$$



- On peut démontrer que la probabilité d'un événement associée à la variable construite à partir de l'algorithme en (4) vérifie

$$\begin{aligned} P(U_1 \geq p, U_1 U_2 \geq p, \dots, U_1 U_2 \cdots U_k \geq p, U_1 U_2 \cdots U_{k+1} < p) \\ = \frac{p}{k} (-1)^k (\log(p))^k \end{aligned}$$

- Et remplaçant p par sa valeur $e^{-\lambda}$, cette probabilité devient

$$\begin{aligned} \frac{p}{k} (-1)^k (\log(p))^k &= \frac{e^{-\lambda}}{k} (-1)^k \left(\log(e^{-\lambda}) \right)^k \\ &= \frac{e^{-\lambda}}{k} (-1)^k (-\lambda)^k \\ &= \frac{e^{-\lambda} \lambda^k}{k!} \end{aligned}$$

- Cet algorithme nous génère effectivement une variable de Poisson de probabilité donnée en (3).

Simulation de Variables Aléatoires Continues



Propriété(s) : Si F une distribution cumulative continue donnée et U une variable aléatoire uniforme sur $[0, 1]$, alors la variable aléatoire définie par

$$X = F^{-1}(U)$$

admet F comme fonction de répartition.

- ▷ La fonction de densité de la **loi de Cauchy** étant définie par

$$f_X(x) = \frac{1}{\pi(x^2 + 1)}$$

- ▷ On en déduit sa fonction de répartition $F_X(\cdot)$ donnée par

$$F_X(x) = \frac{1}{\pi} \left(\arctan(x) - \frac{\pi}{2} \right)$$

- ▷ Pour générer ainsi la variable X , on utilise la relation suivante

$$X = F_X^{-1}(U) = \tan \left(\pi \left(U + \frac{1}{2} \right) \right)$$

- ▷ La fonction tangente étant périodique de période π , on peut l'écrire

$$X = \tan \left(\pi \left(U - \frac{1}{2} \right) \right)$$



- ▷ Une variable aléatoire X est dite exponentielle si sa densité de probabilité est donnée par

$$f_X(x) = \alpha e^{-\alpha x} \mathbf{1}_{\{x \geq 0\}}, \quad \alpha > 0$$

- ▷ La fonction de répartition de la **loi exponentielle** est définie par

$$F_X(x) = (1 - \alpha e^{-\alpha x}) \mathbf{1}_{\{x \geq 0\}}$$

- ▷ Pour générer X , on utilise la relation suivante

$$X = -\frac{1}{\alpha} \log(1 - U) + \frac{1}{\alpha} \log(\alpha)$$

- ▷ Puisque U est uniformément repartie sur $[0, 1]$ donc $(1 - U)$ l'est également. On peut également générer la variable X comme suit

$$X = -\frac{1}{\alpha} \log(U) + \frac{1}{\alpha} \log(\alpha)$$



- ▷ X est une variable aléatoire de Rayleigh si sa densité de probabilité est

$$f_X(x) = xe^{-\frac{x^2}{2}} \mathbf{1}_{\{x \geq 0\}}$$

- ▷ La fonction de répartition de la **variable aléatoire de Rayleigh** est alors

$$F_X(x) = 1 - e^{-\frac{x^2}{2}}$$

- ▷ Pour générer X , on utilise la relation suivante

$$X = \sqrt{-2 \log(1 - U)}$$

- ▷ Puisque U est uniformément repartie sur $[0, 1]$ donc $(1 - U)$ l'est également. On peut également générer la variable X comme suit

$$X = \sqrt{-2 \log(U)}$$



- ▷ Soit X une variable aléatoire gaussienne de moyenne nulle et de variance unitaire. X peut être générée de trois façons différentes.
- ▷ **1^e façon:** On pose $X = U_1 + U_2 + \dots + U_{12} - 6$ où $U_i, i = 1, \dots, 12$, sont uniformément réparties sur $[0, 1]$. On a donc $E[X] = 0$ et $\text{Var}[X] = 1$. Comme X est la somme de 12 variables aléatoires i.i.d, sous la loi du théorème de la limite centrale, on peut considérer X comme gaussienne.
- ▷ **2^e façon:** On peut générer deux variables aléatoires gaussiennes X et Y à partir de 2 variables uniformes U_1 et U_2 sur $[0, 1]$. On a:

$$X = \sqrt{-2 \log(U_1)} \cos(2\pi U_2)$$

$$Y = \sqrt{-2 \log(U_1)} \sin(2\pi U_2)$$

- ▷ **3^e façon:** On utilise la commande de MATLAB `randn` pour générer une variable $N(0, 1)$. La commande `randn(m, n)` génère une matrice $m \times n$ dont les éléments sont gaussiens de moyenne nulle et de variance 1.

Note: Si la variable aléatoire X suit une loi normale $N(\mu, \sigma^2)$, alors on peut l'écrire $X = \mu + \sigma Z$ où $Z \sim N(0, 1)$, afin de simuler Z .

Simulation de Vecteurs Aléatoires

- ▷ On souhaite générer un vecteur aléatoire gaussien \underline{X} ayant une matrice de variance-covariance connue

$$\Lambda_{\underline{X}} = \text{Cov}(\underline{X}, \underline{X}^T)$$

- ▷ Soit $\underline{Z} = (Z_1, \dots, Z_n)^T$ un vecteur gaussien indépendant de moyenne nulle et de variance unitaire.

$$E[\underline{Z}] = 0 \text{ et } [\underline{Z}] = I$$

- ▷ I est la matrice identité de dimension n , c'est à dire une matrice contenant des 1 sur sa diagonale et 0 ailleurs.
- ▷ Le vecteur \underline{Z} sera utilisé pour exprimer le vecteur aléatoire \underline{X} .

- Soit le vecteur X ayant les propriétés suivantes

$$\underline{X} = \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} \sim N\left(\underline{0}, \begin{pmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix}\right) \quad (5)$$

où $E[X_i] = 0$, $E[X_i^2] = \sigma_i^2$ et $E[X_1X_2] = \rho\sigma_1\sigma_2$.

- Soit le vecteur gaussien $\underline{Z} = (Z_1, Z_2)^T \sim N(\underline{0}, I)$.
- Comment choisir α et β tels que la variable \underline{X} s'écrit

$$\begin{cases} X_1 = \sigma_1 Z_1 \\ X_2 = \alpha Z_1 + \beta Z_2 \end{cases} \iff \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \begin{pmatrix} \sigma_1 & 0 \\ \alpha & \beta \end{pmatrix} \begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix}$$

Il faut prendre alors $\alpha = \rho\sigma_2$ et $\beta = \sigma_2\sqrt{1 - \rho^2}$.

- On générera ainsi le vecteur X à l'aide du vecteur Z à travers la relation

$$\begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \begin{pmatrix} \sigma_1 & 0 \\ \rho\sigma_2 & \sigma_2\sqrt{1 - \rho^2} \end{pmatrix} \begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix} \quad (6)$$



- ▷ Le cas général d'un vecteur \underline{X} de dimension n quelconque, défini par

$$\underline{X} = L\underline{Z} \implies \text{Cov}(\underline{X}, \underline{X}^T) = E[\underline{L}\underline{Z}\underline{Z}^T\underline{L}^T] \quad (7)$$

- ▷ De plus, sachant que $E[\underline{Z}] = 0$ et $\text{Cov}(\underline{Z}, \underline{Z}^T) = I$, alors on obtient

$$E[\underline{X}] = E[L\underline{Z}] = 0$$

$$\text{Cov}(\underline{X}, \underline{X}^T) = LE[\underline{Z}\underline{Z}^T]L^T = LIL^T = LL^T$$

- ▷ Sachant que $LL^T = \text{Cov}(\underline{X}, \underline{X}^T) = \Lambda_{\underline{X}}$, on obtient donc L par la factorisation LL^T (**factorisation de Cholesky**) de la matrice de variance-covariance $\Lambda_{\underline{X}}$.

Note: Connaissant la décomposition LL^T de la matrice $\Lambda_{\underline{X}}$, pour générer un vecteur aléatoire gaussien \underline{X} de matrice d'auto-covariance $\Lambda_{\underline{X}}$, il suffit de générer un vecteur aléatoire gaussien \underline{Z} de moyenne nulle et de variance la matrice unité, $N(0, I)$, et de prendre $\underline{X} = L\underline{Z}$.

- ▷ Si $\Lambda_{\underline{X}}$ est définie positive alors la matrice L existe et est unique. La commande `chol($\Lambda_{\underline{X}}$)` de MATLAB nous permet d'avoir la matrice L .



- ▷ Elle est aussi connue sous le nom d'**analyse en composantes principales**.
- ▷ L'avantage de cette méthode est de
 - ★ Marcher même avec des matrices qui ne sont pas définies positives.
 - ★ Donner des intuitions sur la structure aléatoire des risques.
- ▷ La Procédure s'effectue comme suit:
 - ★ Trouver deux matrices D et P telle que $\text{Var}(\underline{X}) = \Lambda_{\underline{X}} = PDP^T$ où $\Lambda_{\underline{X}}$ est la matrice variance-covariance, D une matrice diagonale et P une matrice orthogonale.
 - ★ On pose $\Lambda_{\underline{X}} = LL^T$ avec $L = P\sqrt{D}$.
 - ★ Le vecteur \underline{X} est obtenu exactement comme dans le cas de la factorisation de Cholesky en posant $\underline{X} = LZ$ avec \underline{Z} composé de n variables indépendantes et de variance unitaire.

Méthode de Génération par Rejet



- ▷ On veut simuler un vecteur aléatoire \underline{X} de densité de probabilité $f(x)$.
- ▷ On suppose qu'il existe déjà sur l'ordinateur un programme pour simuler le vecteur aléatoire Y de même dimension que X , ayant la densité de probabilité $g(y)$ avec $f(x) \leq kg(x)$, où k est une constante positive donnée.
- ▷ On pose $\alpha(x) = \frac{f(x)}{kg(x)}$
- ▷ Algorithme
 - ★ On génère le vecteur Y_1 de densité de probabilité $g(y)$ et une variable aléatoire U_1 indépendante de Y_1 , uniformément répartie entre 0 et 1.
 - ★ Si $U_1 \leq \alpha(Y_1)$, on choisit $X = Y_1$, sinon rejettions Y_1 et reprenons la génération de Y et U jusqu'à ce que $U_m \leq \alpha(Y_m)$, on choisira $X = Y_m$.

Ainsi généré, le vecteur aléatoire X obéit à la loi $f(x)$.

Méthode Monte Carlo et Chaînes de Markov (MCMC)



- Soit un processus aléatoire $\{X_n, n = 0, 1, 2, \dots\}$. Le processus $\{X_n, n = 0, 1, 2, \dots\}$ est une chaîne de Markov si

$$\begin{aligned} P(X_{n+1} = x_{n+1} | X_n = x_n, X_{n-1} = x_{n-1}, \dots, X_0 = x_0) \\ = P(X_{n+1} = x_{n+1} | X_n = x_n) \end{aligned} \quad (8)$$

Note: La probabilité que le processus soit à l'état x_{n+1} à $n+1$ sachant qu'il a été à l'état x_n à n est indépendant de tout ce qui s'est passé avant n .

- Une chaîne de Markov est dite homogène si

$$P(X_{n+1} = y | X_n = x) = p_{xy} \quad (9)$$

Note: La probabilité d'être à l'état y sachant que le processus a été à l'état x précédemment est totalement indépendant de n .

Note: La méthode Monte Carlo sert à estimer l'espérance mathématique d'une quantité aléatoire $q(X)$, où X est une variable aléatoire:

$$E[q(X)] \approx \frac{1}{N} \sum_{i=1}^N q(X_i) \quad (10)$$

- ▷ Supposons qu'on ait une chaîne de Markov $\{X_n, n = 0, 1, 2, \dots\}$.
- ▷ Notons $\phi_X(x)$ la fonction densité de probabilité de X de l'état de la chaîne en régime permanent si l'on arrive à construire la chaîne pour que $\phi_X(x)$ soit identique à $f_X(X)$ désirée.
- ▷ On laisse évoluer la chaîne jusqu'au régime stationnaire (M grand) et on prend ensuite les états suivants comme les échantillons générés pour estimer $E[q(X)]$:

$$E[q(X)] \approx \frac{1}{N - M} \sum_{i=M+1}^N q(X_i) \quad (\text{Moyenne ergodique}) \quad (11)$$

- ▷ Soit $g(x|y)$ une fonction de densité de probabilité conditionnelle quelconque.
- ▷ Posons $\alpha(x, y) = \min\left(1, \frac{f_X(y)g(x|y)}{f_X(x)g(y|x)}\right)$
- ▷ Pour construire la chaîne de Markov désirée, on suppose qu'à l'instant n la chaîne se trouve à l'état $X_n = x_n$.
- ▷ On génère deux variables aléatoires, la première Y se caractérise par la densité de probabilité $g(y|x_n)$ et la deuxième U_n est indépendante de la première et est uniformément repartie entre 0 et 1. X_{n+1} est alors définie

$$X_{n+1} = \begin{cases} Y, & \text{si } U \leq \alpha(x_n, y) \\ X_n, & \text{si } U > \alpha(x_n, y) \end{cases} \quad (12)$$

$g(x|y)$ n'a pas besoin de prendre une forme particulière, on la choisira de façon à convenir au problème à traiter. Pour lectures complémentaires, voir [1, 2, 3, 4, 5, 7, 6, 8].

- [1] D. Etter and D. Kuncicky.
Introduction to Matlab 6.
Prentice-Hall, 2002.
- [2] E. P. Kloeden, P. E. and H. Schurz.
Numerical solution of SDE through computer experiments.
Springer, 2nd edition, 1997.
- [3] P. E. Kloeden and E. Platen.
Numerical solution of stochastic differential equations.
Springer, 1992.
- [4] W. L. Martinez and A. R. Martinez.
Computational statistics handbook with MATLAB.
Chapman Hall/CRC, 2002.
- [5] S. T. W. V. Press, W. and B. Flannery.
Numerical Recipes in C: The Art of Scientific Computing.
Cambridge University Press, 1992.

[6] S. M. Ross.

A first course in probability.

Prentice Hall, 6th edition, 2002.

[7] R. Y. Rubinstein.

Simulation and the Monte Carlo Method.

Wiley, 1981.

[8] R. Seydel.

Tools for Computational Finance.

Springer, 2002.