Outils numériques pour la simulation Monte Carlo des produits dérivés complexes.

Pierre-Alain Patard*
Université de Lyon, Lyon, F-69007, France ;
université Lyon 1, Ecole ISFA, Lyon, F-69007, France.†‡

12 juillet 2007

Résumé

L'essor récent des marchés de produits dérivés, la mise en place de réformes comptables IAS/IFRS et l'entrée en vigueur des directives réglementaires Bâle II et Solvabilité II dans les banques et les sociétés d'assurance ont transformé les méthodes de Monte Carlo en un outil incontournable pour les spécialistes de la gestion du risque. L'objectif de ce travail est de montrer comment le praticien peut mettre en œuvre les techniques de simulation, présentées sous un angle théorique dans la littérature spécialisée, pour implémenter un outil d'évaluation des produits dérivés. En particulier, il s'attache à : (i) identifier un générateur pseudo-aléatoire uniforme rapide et robuste, (ii) proposer une technique d'échantillonnage de la loi normale adaptée aux contraintes calculatoires de la simulation numérique intensive, (iii) présenter les techniques de simulation Monte Carlo et deux solutions pour accélérer la convergence de l'estimateur et (iv) montrer comment ces méthodes peuvent être appliquées de manière systématique pour évaluer une option dont le prix dépend du chemin suivi par le sous-jacent.

Mots-Clefs : générateur pseudo-aléatoire, Mersenne Twister, loi normale, méthode de Box-Muller, inversion de la fonction de répartition, approximation de Acklam, effet Neave, Monte Carlo, efficacité d'un estimateur, réduction de variance, variables antithétiques, méthode de Monte Carlo adaptative, produits dérivés, Black et Scholes, mouvement Brownien.

*L'auteur remercie Frédéric Planchet (l'éditeur), Jean-Claude Augros, Jean-Paul Laurent, Alexis Bienvenue et le relecteur anonyme pour leurs remarques constructives sur le fond et la forme de ce document. Nous souhaitons aussi remercier Makoto Matsumoto pour les échanges que nous avons eus au sujet du Mersenne Twister et de ses évolutions.

†Ecole ISFA - Institut de Science Financière et d'Assurances - Université Claude Bernard Lyon 1, 50 Avenue Tony Garnier, 69366 Lyon Cedex 7.

‡Pierre-Alain Patard est actuaire de l'Institut de Science Financière et d'Assurances. Contact : pierrealain.patard@free.fr.

Abstract

The recent rise of the derivatives products markets, the installation of accounting reforms IAS/IFRS and of the lawful directives Basle II and Solvency II in the banks and the insurance companies transformed the Monte Carlo methods in an essential tool for the risk management. The objective of this work is to show how the practitioner can implement the simulations techniques, presented under a theoretical point of view in the specialized literature, to build a tool for evaluation of the derivatives products. This study is focused on the following aspects : (i) to identify a fast and robust uniform pseudo-random number generator, (ii) to propose a sampling technique of the normal distribution adapted to the computational constraints of intensive numerical simulations, (iii) to present the Monte Carlo principles and two solutions to improve the convergence rate of the estimator and (iv) to show how these methods can be applied in a systematic way to evaluate an option whose price depends on the path followed by the underlying.

Keywords : pseudo-random number generator, Mersenne Twister, Gaussian distribution, Box-Muller method, inverse cumulative normal distribution, Acklam approximation, Neave effect, Monte Carlo, estimator efficiency, variance reduction, antithetic variates, adaptative Monte Carlo method, derivatives products, Black and Scholes, Brownian motion.

Introduction

Depuis quelques années, on assiste à un développement sans précédent de l'industrie des produits dérivés qui se traduit par une augmentation considérable des volumes de transactions, une complexification notoire des stratégies (produits hybrides, options sur CPPI, dérivés climatiques) et une diversification des sous-jacents (actions, taux, inflation, matières premières, températures, chutes de pluies). Ce phénomène s'explique principalement par une concurrence accrue entre les banques d'investissement qui souhaitent préserver leurs marges et par les exigences croissantes de clients toujours plus avertis et informés et désireux de profiter de toutes les opportunités offertes par les marchés mondiaux. Pour analyser et mesurer les risques impliqués par cette intensification des échanges, les intervenants (salles de marché, sociétés de gestion, fonds d'investissement) doivent développer des modèles de marché probabilistes toujours plus réalistes. Dans ce contexte, les méthodes de simulation deviennent un outil incontournable pour manipuler les systèmes stochastiques sophistiqués servant à évaluer et couvrir les produits dérivés complexes (voir Jäckel [11] ou Glasserman [8]).

Par ailleurs, les banques et les sociétés d'assurance connaissent une profonde évolution de leur environnement réglementaire : d'abord avec la mise en oeuvre des normes comptables internationales IAS/IFRS (en 2005) qui insistent sur la nécessité de valoriser l'actif et le passif d'une société à leur valeur de marché (et non plus selon leur valeur historique amortie), ensuite avec la mise en place de

la directive Bâle II pour les banques[1] et de la directive Européenne Solvabilité II pour les sociétés d'assurance[2], qui ont pour objectif de promouvoir la stabilité et la sécurité du système financier en édictant des normes prudentielles axées sur une quantification probabiliste des risques. Ces directives doivent aboutir à la mise en place de méthodes internes d'évaluation des risques, qui permettront aux établissements de mieux adapter leurs fonds propres à la réalité des risques qu'ils encourent.

Notre travail s'inscrit dans un contexte où les méthodes de simulation numérique sont devenues un outil indispensable pour la modélisation et la quantification des risques dans les banques et les compagnies d'assurance. Il s'attache à montrer comment le praticien peut utiliser les résultats théoriques de la littérature spécialisée pour répondre au problème de l'évaluation des produits dérivés par la méthode de Monte Carlo. Les sections 1 et 2 sont consacrées au choix fondamental des outils de simulation, tandis que les sections 3 et 4 présentent la méthode de Monte Carlo, deux méthodes systématiques pour réduire la variance ainsi que leur implémentation pour évaluer un produit dérivé.

La première partie pose le problème de l'imitation du hasard sur un ordinateur, i.e. par un procédé déterministe (cf. L'Ecuyer [19]). A ce titre, nous présentons la technologie dite "Mersenne Twister" et nous montrons qu'elle conduit à des solutions rapides et robustes pour simuler la loi uniforme $\mathcal{U}(0,1)$. Dans la seconde partie, nous étudions deux techniques pour échantillonner la loi gaussienne à partir de la loi uniforme : la transformation de Box-Muller, souvent proposée dans la littérature et la méthode d'inversion de la fonction de répartition proposée par Acklam en 2004. Les tests pratiqués s'inspirent des travaux de Neave [27] et montrent que le premier algorithme induit des biais d'échantillonnage non négligeables, tandis que la seconde solution permet de supprimer ces biais. Dans la troisième partie, nous rappelons les principes et les propriétés de la méthode de Monte Carlo. Nous présentons ensuite deux techniques pour réduire la variance de l'estimateur (de manière systématique et indépendamment de la forme du problème) : la méthode "classique" des variables antithétiques et une méthode dite "adaptative", plus récente et plus flexible que la méthode antithétique (voir Arouna [2]). Dans la dernière partie, nous appliquons les méthodes étudiées précédemment pour évaluer des produits dérivés complexes. Nous prenons comme exemple le cas d'une option asiatique géométrique mono sous-jacent, pour laquelle le prix est connu sous une forme explicite[3], puis nous montrons comment les méthodes de réduction de variance permettent de contrôler l'incertitude sur le prix simulé.

[1] La réforme Bâle II est entrée en vigueur dans l'Union Européenne en 2007. Au sujet des normes Bâle II, le lecteur pourra consulter les ouvrages de Chorafas [5] ou de Ong [29].

[2] La version finalisée de la réforme Solvabilité II (ou Solvency II) est attendue courant 2007 et sa mise en application prévue en 2010. Pour une discussion approfondie sur les problématiques techniques soulevées par cette directive, le lecteur pourra se référer à l'ouvrage de Planchet et al. [33] et aux travaux de Therond : http ://therond.pierre.free.fr.

[3] Etant donné que le prix "réel" du produit est connu, nous serons en mesure d'apprécier la convergence de la méthode numérique vers son objectif théorique.

1 Générateurs pseudo-aléatoires

En raison de leur simplicité et parce qu'elles nécessitent des calculs intensifs et répétitifs, les méthodes de simulation se prêtent bien à une implémentation informatique, dans la mesure où l'on sait produire rapidement des nombres au hasard par un procédé déterministe.

1.1 Considérations générales sur les nombres aléatoires

1.1.1 Choix d'une source de hasard

Sources de hasard réel On connaît aujourd'hui une seule méthode pour obtenir des nombres véritablement aléatoires. Elle consiste à mesurer des phénomènes physiques intrinsèquement aléatoires, comme le bruit thermique dans les semi-conducteurs ou les émissions d'une source radioactive [16]. Cette approche semble particulièrement prometteuse dans le domaine de la cryptographie. Ainsi, on sait obtenir des clés de chiffrement uniques et imprédictibles en exploitant les propriétés quantiques de photons polarisés [14]. Cependant, elle nécessite des équipements spéciaux particulièrement coûteux, ce qui la rend impropre à la simulation numérique sur les systèmes courants.

Sources de hasard virtuel Les spécialistes préfèrent exploiter d'autres techniques, dont l'objectif est d'imiter le hasard plutôt que de le créer. Pour cela on utilise des algorithmes purement déterministes, appelés générateurs pseudo-aléatoires. Les séquences construites par un tel générateur sont sensées reproduire fidèlement les propriétés statistiques de suites de nombres véritablement aléatoires. On démontre qu'il ne suffit pas de juxtaposer "au hasard" des instructions machine pour obtenir un bon générateur. Cette démarche peut s'avérer désastreuse. En conséquence, l'élaboration d'un générateur doit reposer sur des fondements théoriques solides.

Architecture d'un générateur pseudo-aléatoire La plupart des générateurs pseudo-aléatoires fabriquent des nombres U_k apparemment i.i.d. de loi $\mathcal{U}(0,1)$ selon un schéma récurrent et déterministe de la forme suivante :

$$U_k = g(s_k), \text{ où } s_k = f(s_{k-1}) \text{ et } s_0 \in \mathcal{S}. \tag{1.1}$$

Les fonctions $f : \mathcal{S} \to \mathcal{S}$ (fonction de transfert) et $g : \mathcal{S} \to (0,1)$ (fonction de sortie) sont déterministes. L'espace des états $\mathcal{S}$ est un ensemble fini de symboles représentables en machine. Le symbole produit à la k-ième itération, s_k, est l'état interne du générateur. Cette présentation formelle des générateurs pseudo-aléatoires est due à L'Ecuyer [19].

Choix de l'état initial L'état initial s_0, qui permet d'amorcer la récurrence, est aussi appelé graine (seed en anglais) ou encore germe du générateur. Lorsqu'il est fixé une fois pour toutes, on obtient invariablement la même séquence.

Cela facilite le développement et la mise au point des modèles et permet de reproduire une expérience virtuelle avec les mêmes conditions initiales. En dehors de ces besoins particuliers, il est recommandé d'amorcer le générateur avec des graines uniformément i.i.d. dans l'espace des états, ce qui permet d'envisager, équitablement et sans biais, l'ensemble des évolutions possibles pour le modèle. Comme l'objectif visé est l'analyse d'un phénomène simulé et non pas la sécurité d'un système, on peut engendrer les graines successives avec un générateur pseudo-aléatoire auxiliaire (plus facile à exploiter qu'une source de hasard physique).

1.1.2 Propriétés indésirables et propriétés recherchées

Défauts structurels des algorithmes pseudo-aléatoires Comme l'espace des états est fini, l'algorithme ne peut renvoyer qu'un nombre fini de valeurs distinctes et, comme la dynamique (1.1) est déterministe, le générateur retrouve le même état interne au bout d'un certain nombre d'itérations. Ensuite, les mêmes séquences sont à nouveau générées. En d'autres termes, les générateurs pseudo-aléatoires sont périodiques. Ces propriétés des séquences simulées ne sont pas en accord avec le fait qu'une séquence véritablement aléatoire de loi $\mathcal{U}(0,1)$ est par nature non-périodique et qu'elle prend une infinité de valeurs.

En pratique, on exige que la période T du générateur (déterminée par f et $\mathrm{card}(\mathcal{S})$) soit largement supérieure à la longueur de toutes les séquences envisageables et que l'échantillonnage du segment unité (déterminé par $\mathcal{S}$ et g) soit le plus fin possible. Il est communément admis que pour un bon générateur on doit avoir

$$T \simeq \mathrm{card}(\mathcal{S}) \text{ et, si possible, } T \geq 2^{60} \simeq 1.15 \times 10^{18}. \tag{1.2}$$

Pour cela, on peut choisir f comme une permutation imprédictible des éléments de $\mathcal{S}$ et construire g de façon à transformer les états internes successifs en une suite de valeurs discrètes bien équidistribuées.

Propriétés statistiques recherchées Le critère (1.2) ne suffit pas à définir un bon générateur ([19], p. 4). Il faut aussi vérifier les propriétés statistiques des séquences générées (uniformité, équidistribution, indépendance, imprédictibilité) par des tests exigeants qui permettent d'identifier les algorithmes les plus efficaces. De tels tests sont présentés de manière approfondie dans Knuth [13], L'Ecuyer [17], Niederreiter [28]. Malgré tout, chaque générateur pseudo-aléatoire a des caractéristiques intrinsèques qui le rendent impropre à certains types d'applications. C'est pourquoi, il est recommandé d'utiliser exclusivement des générateurs dont les propriétés théoriques ont été établies par des spécialistes, puis validées par un jeu de tests connus comme `DIEHARD` (Marsaglia [22]) ou `TestU01` (L'Ecuyer et Simard [21]).

Propriétés non statistiques souhaitables Lorsque le générateur est utilisé pour la simulation numérique intensive, certaines propriétés, de nature non statistique, comme la rapidité des calculs, la reproductibilité des séquences (qui permet de recommencer une expérience virtuelle dans des conditions identiques) et la portabilité du code (pour la mise en oeuvre sur différentes machines) s'avèrent particulièrement intéressantes.

1.1.3 Evolution de la technologie

Les algorithmes pseudo-aléatoires les plus anciens (les plus simples aussi) sont les générateurs à congruences linéaires (L'Ecuyer[4], Knuth [13]). Bien qu'ils équipent la plupart des systèmes de calcul standards, leurs propriétés s'avèrent souvent décevantes (cf. L'Ecuyer [18], Klimasauskas [12]).
Les spécialistes ont su faire évoluer les techniques (voir Gentle [7]) parallèlement à l'évolution de la puissance de calcul des ordinateurs, d'abord en combinant des générateurs connus [19], puis en explorant des solutions nouvelles. Aussi, les générateurs récents sont-ils conçus autour de l'architecture binaire des ordinateurs (voir L'Ecuyer et Panneton [20] ou Panneton [30]).

1.2 Générateurs linéaires congruentiels

La méthode des congruences linéaires fut introduite par Lehmer en 1949. Elle est particulièrement bien présentée dans l'ouvrage de Knuth ([13] p. 10).

1.2.1 Approche théorique

Dynamique linéaire congruentielle La dynamique d'un générateur linéaire congruentiel (LCG) est donnée par :

$$X_k = (aX_{k-1} + c) \bmod m \text{ et } X_0 \in \mathbb{N}, \tag{1.3}$$

avec $m \in \mathbb{N}^*$ (le module), $a \in \mathbb{N}^*$ (le multiplicateur), $c \in \mathbb{N}$ (l'incrément).

La récurrence (1.3) est appelée suite de Lehmer et son comportement est entièrement déterminé par le triplet (m, a, c) et X_0 (cf. L'Ecuyer [19], p. 4). Par construction, l'espace des états et la période d'un générateur congruentiel vérifient

$$\mathcal{S} \subset \mathbb{N}_m \stackrel{\text{def}}{=} \{0, \ldots, m-1\} \text{ et } T \leq m.$$

En distinguant le cas $c > 0$ (générateurs purement affines) du cas $c = 0$ (générateurs congruentiels multiplicatifs) on sait trouver des jeux de paramètres qui permettent de maximiser la période.

[4] http ://www.iro.umontreal.ca/~lecuyer/

Obtenir des nombres uniformes dans $(0,1)$ Comme $0 \leq X_k \leq m-1$, il y a trois possibilités pour construire un nombre uniforme U_k entre 0 et 1 :

$$U_k = \frac{X_k}{m} \text{ ou } U_k = \frac{X_k}{m-1} \text{ ou } U_k = \frac{X_k + 0.5}{m} = \frac{X_k}{m} + \frac{1}{2m}.$$

La première (resp. la seconde) solution conduit à des nombres dans l'intervalle semi-ouvert $[0,1[$ (resp. l'intervalle fermé $[0,1]$), tandis que la troisième solution génère des nombres dans l'intervalle ouvert $]0,1[$. Nous recommandons cette dernière approche, car elle présente deux avantages : (i) il n'est pas possible d'obtenir 0 ou 1 (intéressant lorsqu'on applique l'inverse d'une fonction de répartition aux sorties du générateur), (ii) les valeurs possibles pour U_k sont dans l'ensemble $\{1/(2m), \ldots, 1-1/(2m)\}$, qui est symétrique autour de 1/2.

1.2.2 Générateurs congruentiels multiplicatifs

La plupart des logiciels de calcul ou de développement disposent d'un générateur de nombres aléatoires. Pour des raisons principalement historiques, celui-là est souvent de type linéaire congruentiel.

Définition et propriété Lorsque $c = 0$, on parle de générateur congruentiel multiplicatif (MLCG) et la récurrence (1.3) devient :

$$\forall k \in \mathbb{N}^*, \; X_k = (aX_{k-1}) \bmod m. \tag{1.4}$$

Dans ce cas, l'état 0 est absorbant : si $X_k = 0$, alors les termes suivants dans la suite seront tous nuls. Le générateur doit donc prendre ses valeurs dans $\mathbb{N}_m \setminus \{0\}$, de cardinal $m-1$. Knuth ([13] p. 20) démontre le théorème suivant.

Théorème 1.1 *Soit X un MLCG défini par (m,a) et X_0.*
- *Si m est premier, la période maximale vaut $m-1$. Elle est atteinte si et seulement si $X_0 \wedge m = 1$ et a est primitif* [5] *modulo m.*
- *Si $m = 2^n$ $(n \geq 4)$, la période maximale vaut $m/4$. Elle est atteinte si et seulement si $X_0 \bmod 8 = 1$ et $a \bmod 8 = \pm 3$.*

Une implémentation naïve de la dynamique (1.4) suppose que le produit ax soit représentable en machine, ce qui est rarement le cas. En effet, les générateurs acceptables ont un module m voisin du plus grand entier représentable et un multiplicateur a élevé. Lorsque $a^2 < m$, Schrage [35] propose une méthode efficace pour calculer $ax \bmod m$ sans dépassement de capacité.

Générateur `Ran0` (Park et Miller, 1988) Dans un article de 1988, Park et Miller [31] considèrent le générateur suivant

$$U_k = X_k / \left(2^{31} - 1\right), \text{ où } X_k = \left(16807 \; X_{k-1}\right) \bmod \left(2^{31} - 1\right).$$

[5] Un entier a est primitif modulo m si, et seulement si, $a^{m-1} \bmod m = 1$ et $a^{k-1} \bmod m \neq 1$ pour $k = 1, \ldots, m-1$.

Selon les auteurs, ce générateur constitue le standard minimal utilisable par les non-spécialistes. En effet, l'algorithme est convenablement testé, le code est portable sur tous les systèmes et la période ($T_{\texttt{Ran0}} = 2^{31} - 1 \simeq 2.15 \times 10^9$) est maximale au sens du théorème 1.1. Soulignons toutefois que la période de `Ran0` semble un peu courte pour des simulations intensives.

1.3 Générateurs Mersenne Twister

1.3.1 Fondements de l'approche Mersenne Twister (MT)

Les Mersenne Twister sont des générateurs récents, proposés pour la première fois par Matsumoto[6] et Kurita en 1998 [26]. L'idée originale des auteurs est de définir la récurrence du générateur, non pas à partir des opérations arithmétiques classiques sur les entiers (comme pour la plupart des générateurs courants), mais à partir des opérations d'arithmétique matricielle dans le corps fini $\mathbb{N}_2 = \{0, 1\}$.

Cette approche nouvelle présente quatre avantages majeurs : (i) on peut écrire l'algorithme avec les opérateurs de bits présentés dans le paragraphe suivant (cf. L'Ecuyer et Panneton [20]), de sorte que le générateur exploite pleinement l'architecture binaire de l'ordinateur (voir Panneton [30]), (ii) les temps de calcul sont considérablement réduits (les opérateurs de bits sont très rapides), (iii) on peut obtenir des générateurs de période arbitrairement longue (les bits alloués à la mémoire du générateur sont mélangés à chaque itération, ce qui accroît l'espace des états du générateur), et (iv) on améliore l'équidistribution multidimensionnelle du générateur en appliquant un ultime mélange des bits avant renvoi d'un nouveau nombre uniforme [24, 25].

1.3.2 Opérateurs de bits

Représentation des entiers en machine L'ensemble des entiers représentables en machine est de la forme $\mathbb{N}_{2^\omega}$, où ω désigne le nombre de bits de l'ordinateur[7]. Tout entier $X \in \mathbb{N}_{2^\omega}$, de décomposition binaire $\sum_{i=0}^{\omega-1} x_i 2^i$, est stocké sous la forme d'un "vecteur de bits" : $X \equiv (x_{\omega-1}, x_{\omega-2}, \ldots, x_0)$.

Décalage de bits Soit $0 \leq v \leq \omega$. On note "$\gg v$" le décalage de v bits vers la droite (lire v bits right shift) défini par

$$X \gg v \stackrel{\text{def}}{=} (0, \ldots, 0, x_{\omega-1}, \ldots, x_{v+1}) = \lfloor X/2^v \rfloor .$$

De manière symétrique, le décalage de v bits vers la gauche (v bits left shift) est noté "$\ll v$" et correspond à l'opération

$$X \ll v \stackrel{\text{def}}{=} (x_{\omega-v-1}, \ldots, x_0, 0, \ldots, 0) = (2^v X) \bmod 2^\omega.$$

[6] http ://www.math.sci.hiroshima-u.ac.jp/~m-mat/eindex.html

[7] $\omega = 32$ sur une machine 32 bits et $\omega = 64$ sur une machine 64 bits.

Les opérateurs de décalage de bits sont donc des raccourcis pour la division (resp. la multiplication) par une puissance de 2. Ils sont particulièrement rapides, car ils agissent directement sur les bits de l'entier X.

Arithmétique bit à bit Soit $Y = \sum_{i<\omega} y_i 2^i \in \mathbb{N}_{2^\omega}$, on définit les opérateurs "bit à bit" :

$$X \oplus Y \stackrel{\text{def}}{=} \sum_{i=0}^{\omega-1} (x_i \oplus y_i)\, 2^i \text{ et } X \otimes Y \stackrel{\text{def}}{=} \sum_{i=0}^{\omega-1} (x_i \otimes y_i)\, 2^i,$$

avec la convention $x_i \oplus y_i = (x_i + y_i) \bmod 2$ et $x_i \otimes y_i = (x_i \times y_i) \bmod 2$.

1.3.3 La dynamique Mersenne Twister

Paramètres du Mersenne Twister Soit $0 \leq r \leq \omega - 1$. On note $\underline{M}_r$ (resp. $\overline{M}_r$) l'entier dont les r bits d'ordre inférieur (resp. les $\omega - r$ bits d'ordre supérieur) sont égaux à 1, les autres bits étant nuls : $\underline{M}_r = 2^r - 1$ et $\overline{M}_r = 2^\omega - 2^r$. Ces entiers sont appelés masques de bits du générateur.

Par ailleurs, on définit une fonction sur l'ensemble des entiers machine par

$$A(x) = (x \gg 1) \oplus \begin{cases} 0 & si \quad x \bmod 2 = 0 \\ a & si \quad x \bmod 2 = 1 \end{cases},$$

où $a \in \mathbb{N}_{2^\omega}$ est une constante entière "bien choisie". Cette fonction, appelée perturbation du générateur, décale les bits de l'entier x de 1 rang vers la droite et, lorsque x est impair, mélange le résultat avec les bits de la constante a.

Récurrence du générateur La dynamique Mersenne Twister est basée sur un schéma récurrent d'ordre n dans l'ensemble des entiers machines. Pour $k \geq 0$, le terme X_{k+n} est construit à partir de X_k, X_{k+1} et X_{k+m} $(0 \leq m < n)$ de la manière suivante :

$$X_{k+n} = X_{k+m} \oplus A\left((X_{k+1} \otimes \underline{M}_r) \oplus (X_k \otimes \overline{M}_r)\right).$$

L'entier $(X_{k+1} \otimes \underline{M}_r) \oplus (X_k \otimes \overline{M}_r)$ est formé en concaténant les r bits d'ordre inférieur de X_{k+1} avec les $\omega - r$ bits d'ordre supérieur de X_{k+1}, puis il est mélangé par la fonction A. Le nouvel entier ainsi obtenu est additionné (bit à bit) avec X_{k+m}. Ces mélanges successifs augmentent l'imprédictibilité du générateur. Notons que la séquence est initialisée en choisissant n entiers machine $(X_0, \ldots, X_{n-1}) \in \mathbb{N}_{2^\omega}^n$.

Opération de tempering Afin d'améliorer l'équidistribution multidimensionnelle des sorties du générateur, les concepteurs proposent de mélanger les bits de X_{k+n} selon l'algorithme suivant :

Algorithme 1.1 Tempering de Matsumoto et Kurita (1998)

$Y \longleftarrow X_{k+n}$
$Y \longleftarrow Y \oplus (Y \gg u)$
$Y \longleftarrow Y \oplus ((Y \ll s) \otimes b)$
$Y \longleftarrow Y \oplus ((Y \ll t) \otimes c)$
$Y \longleftarrow Y \oplus (Y \gg l)$

Cette opération, appelée tempering, intervient avant de renvoyer un nouveau réel dans le segment unité. Différentes techniques de tempering sont discutées dans la thèse de Panneton ([30] pp. 34-37).

Sorties du générateur Le k-ième réel uniforme dans $]0,1[$ est donné par

$$U_k = \frac{Y + 0,5}{2^\omega} \in \left\{ \frac{1}{2^{\omega+1}}, \frac{3}{2^{\omega+1}}, \ldots, 1 - \frac{1}{2^{\omega+1}} \right\}$$

et la période maximale théorique vaut :

$$T_{\texttt{MT}} = 2^{\omega n - r} - 1.$$

Pour certains choix de ω, n et r, la période est un nombre de Mersenne (i.e. un nombre premier de la forme $2^i - 1$), ce qui justifie, à posteriori, le nom de cette famille de générateurs.

1.3.4 Générateur `MT19937` (Matsumoto et Kurita, 1998)

Choix des paramètres Les paramètres de récurrence du générateur `MT19937` sont les suivants :

$$\omega = 32, \quad n = 624, \quad r = 31, \quad m = 397, \quad a = 2567483615.$$

Ce choix permet de maximiser la période :

$$T_{\texttt{MT19937}} = 2^{\omega n - r} - 1 = 2^{19937} - 1 \simeq 4.32 \times 10^{6001}.$$

La période obtenue est un nombre premier de Mersenne comportant environ 6000 chiffres, ce qui est absolument colossal.

Les paramètres de tempering sont

$$u = 11, s = 7, t = 15, l = 18, b = 2636928640 \text{ et } c = 4022730752.$$

Ils assurent à `MT19937` une équidistribution optimale dans 623 dimensions (voir Niederreiter [28] pour une approche théorique et le paragraphe 1.4.1 pour une illustration).

Procédure d'initialisation Soulignons que le générateur `MT19937` est très sensible au choix de l'état initial. S'il contient trop de bits nuls, la suite générée conservera cette tendance sur plus de 10000 simulations [30]. Depuis 2002, les auteurs proposent de construire $(X_0, \ldots, X_{n-1})$ selon une récurrence qui assure une bonne diffusion des bits du registre :

$$X_i = 1812433253 \times ((X_{i-1} \oplus X_{i-1} \gg 30) + i), \quad i = 1, \ldots, n-1,$$

où $X_0 \in \mathbb{N}_{2^\omega}$ est fixé arbitrairement.

1.4 Choix d'un générateur

Le choix d'un générateur pseudo-aléatoire dépend de ses propriétés intrinsèques (longueur de la période et comportement statistique) ainsi que de la machine utilisée (vitesse d'exécution), l'objectif étant de trouver l'algorithme réalisant le meilleur compromis entre ces critères et les besoins réels de l'utilisateur.

1.4.1 Caractéristiques intrinsèques des générateurs présentés

Longueur de la période La période de `Ran0` représente une fraction infinitésimale de la période de `MT19937` :

$$\frac{T_{\texttt{Ran0}}}{T_{\texttt{MT19937}}} \simeq \frac{2.15 \times 10^9}{4.32 \times 10^{6001}} \simeq 4,97 \times 10^{-5993}.$$

Pour les applications pratiques, on considèrera que la période du Mersenne Twister est infinie.

Propriétés statistiques des séquences L'objectif de ce travail est d'offrir au lecteur une présentation synthétique de deux techniques permettant de produire des nombres au hasard et non pas de procéder à des tests statistiques exhaustifs pour comparer différents algorithmes. En effet, il existe pour cela des logiciels bien spécifiques (cf. paragraphe 1.1.2), développés et utilisés par les spécialistes qui ont, par ailleurs, déjà publié ce type de tests comparatifs [21].

Nous retiendrons simplement que le générateur à opérations binaires `MT19937` passe sans difficultés les tests statistiques les plus exigeants [21], tandis que `Ran0` échoue certains tests plus élémentaires (voir l'Ecuyer [18]). De ce point de vue, `MT19937` surclasse le générateur linéaire congruentiel (et plus généralement tous les générateurs de ce type).

Structure latticielle Pour simuler des vecteurs i.i.d. selon la loi uniforme sur l'hypercube unité $]0,1[^s$ à partir des sorties successives d'un générateur pseudo-aléatoire, il suffit de considérer la suite de terme général

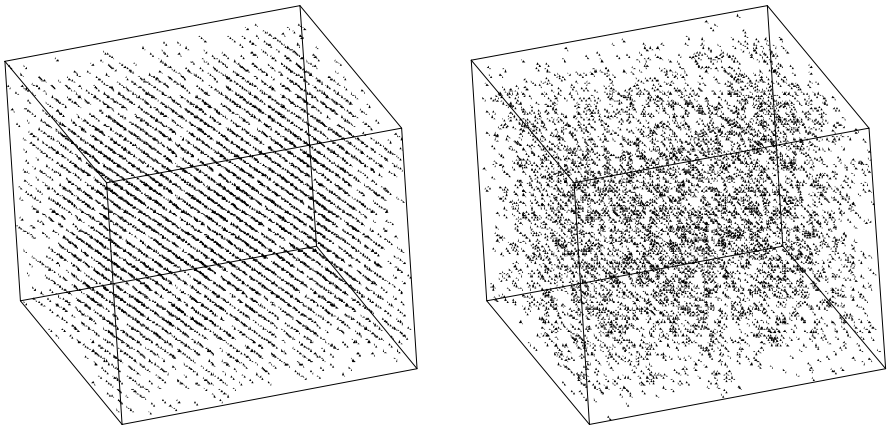
$$\mathrm{U}_k = \left(U_{(k-1)s+1}, U_{(k-1)s+2}, \ldots, U_{ks}\right), \quad k \geq 1. \tag{1.5}$$

D'un point de vue théorique, les points construits selon cette formule doivent occuper le cube "au hasard" et, par conséquent, leur disposition ne doit présenter aucune structure déterministe apparente.

Nous avons réalisé l'expérience suivante avec chacun des deux générateurs :

1. simulation de 625 millions de points "aléatoires" avec $s = 3$,
2. observation du comportement des points dans le "petit" cube $]0, 0.02[^3$.

La figure ci-dessous permet de comparer les résultats obtenus selon que le générateur utilisé est `Ran0` (cube de gauche) ou `MT19937` (cube de droite).



Les points construits avec `Ran0` se répartissent sur des plans parallèles, ce qui est inacceptable pour un générateur censé produire des nombres "au hasard". Cette configuration spatiale particulière est caractéristique des générateurs linéaires congruentiels [13] : elle est appelée structure latticielle. Notons que les points simulés avec le Mersenne Twister ne présentent pas de structure déterministe, ce qui est un atout incontestable pour la simulation numérique multidimensionnelle.

1.4.2 Considérations d'implémentation

Choix d'un langage de programmation La simulation numérique nécessite des calculs intensifs, répétitifs et précis. C'est pourquoi, les spécialistes recommandent de coder les générateurs pseudo-aléatoires dans un langage de programmation puissant, typiquement le C ou le C++. Ces langages sont dits "compilés", car le code d'un programme est traduit une fois pour toutes en instructions machine qui s'exécutent très rapidement. Toutefois, pour des tests ponctuels, il est possible d'utiliser un langage interprété, comme Visual Basic Application. Dans ce cas, les lignes de code sont contrôlées puis converties en instructions machine au fur et à mesure de leur appel, ce qui ralentit l'exécution mais n'affecte en rien la précision des calculs.

Vitesse de calcul Nous avons procédé à un test comparatif consistant à générer 1 milliard de nombres pseudo-aléatoires sur un PC équipé d'un processeur Intel Pentium IV cadencé à 3.20GhZ, de 1Go de RAM et de Microsoft Windows XP Professionnel. Le compilateur utilisé est Microsoft Visual C++ 6.0.

Le générateur minimal standard (`Ran0`) met 62.8 secondes pour achever la simulation : en d'autres termes, il produit environ 16 millions de nombres par seconde. Avec un temps d'exécution 30% plus court que celui du générateur minimal standard (48.2 secondes pour réaliser le test), le générateur `MT19937` se classe en première position du comparatif. Une explication est que l'algorithme Mersenne Twister profite pleinement de l'architecture binaire de la machine et s'exécute particulièrement rapidement : il produit environ 21 millions de nombres par seconde.

1.4.3 Conclusion

Le Mersenne Twister `MT19937` surpasse le générateur minimal standard dans tous les domaines : propriétés statistiques, équidistribution, période et vitesse de calcul. De plus, il est implanté, en versions 32 et 64 bits, dans la très sérieuse bibliothèque de calcul numérique IMSL[8] (International Mathematical and Statistical Library), ce qui est un gage de son efficacité. Ce générateur semble donc répondre parfaitement aux contraintes de la simulation numérique intensive.

2 Simulation de variables gaussiennes

La loi gaussienne est particulièrement utilisée dans les applications financières, notamment pour représenter l'aléa des variables de marché [15]. Nous allons envisager deux techniques pour simuler cette loi de probabilité : la transformation non linéaire d'un jeu de variables uniformes (Box-Muller) puis l'inversion de la fonction de répartition (méthode de Acklam).

2.1 Loi normale unidimensionnelle

Soit G une variable aléatoire définie sur un espace probabilisé $(\Omega, \mathcal{T}, P)$, à valeurs dans $(\mathbb{R}, \mathcal{B}_{\mathbb{R}})$, où $\mathcal{B}_{\mathbb{R}}$ désigne la tribu borélienne. On dit que G suit une loi normale ou gaussienne de paramètres $\mu \in \mathbb{R}$ et $\sigma \in \mathbb{R}_+^*$, si elle admet une densité par rapport à la mesure de Lebesgue de la forme suivante :

$$\varphi_{\mu,\sigma}(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right), \quad x \in \mathbb{R}.$$

On peut démontrer que $\mathrm{E}[G] = \mu$ et $\mathrm{Var}[G] = \sigma^2$: la loi normale est entièrement déterminée par son espérance et sa variance et l'on note indifféremment $G \sim \mathcal{N}(\mu, \sigma)$ ou $\mathcal{N}(\mu, \sigma^2)$.

[8] Cette bibliothèque est éditée par Visual Numerics : http ://www.visualnumerics.com/

La loi $\mathcal{N}(0,1)$ est appelée loi normale centrée réduite ou encore loi normale standard et sa densité sera notée φ. Le théorème suivant nous montre qu'il suffit de savoir générer des variables de loi normale standard pour ensuite simuler des variables de loi $\mathcal{N}(\mu,\sigma)$.

Théorème 2.1 *Si* $G \sim \mathcal{N}(0,1)$ *et* $X = \mu + \sigma G$, *alors* $X \sim \mathcal{N}(\mu,\sigma)$.

2.2 Simulation par transformation de variables uniformes

Une approche classique pour simuler une loi de probabilité donnée consiste à transformer judicieusement un jeu de variables uniformes indépendantes (cf. Niederreiter [28] pp. 164-165).

2.2.1 Méthode de Box-Muller

La méthode de Box-Muller repose sur la transformation :

$$(X,Y) = \Psi(U,V) = \left(\sqrt{-2\ln(U)}\cos(2\pi V), \sqrt{-2\ln(U)}\sin(2\pi V)\right), \quad (2.1)$$

où U et V sont indépendantes de loi $\mathcal{U}(0,1)$. On peut démontrer que X et Y ainsi construites sont indépendantes de loi $\mathcal{N}(0,1)$ (voir Knuth [13] pp. 122-123). Pour simuler une suite (X_k) de variables aléatoires i.i.d. de loi $\mathcal{N}(0,1)$, il suffit donc de poser :

$$\forall k \geq 1,\ (X_{2k-1}, X_{2k}) = \Psi(U_{2k-1}, U_{2k}), \quad (2.2)$$

où U_{2k-1}, U_{2k} sont deux sorties consécutives d'un générateur pseudo-aléatoire. Cette méthode est fréquemment présentée comme une solution simple et efficace pour générer des échantillons gaussiens. La figure 2.1 montre que la distribution simulée est indiscernable de la distribution gaussienne théorique.

2.2.2 Effet Neave

L'interaction entre des systèmes non-linéaires complexes, comme les générateurs pseudo-aléatoires [10] et la méthode de Box-Muller, produit parfois des effets de bord imprévisibles et indésirables [11]. En 1973, Neave [27] a mis en évidence une déformation systématique des queues de distribution de la loi normale quand l'algorithme de Box-Muller est associé avec un générateur linéaire congruentiel.

Distorsion de la densité de probabilité Les résultats présentés dans la suite ont été obtenus en appliquant la formule (2.2) aux sorties de `Ran0` jusqu'à épuisement de la période du générateur, soit $2^{31} - 1$ simulations.

Sur la figure 2.2, on observe un échantillonnage irrégulier (en "dents de scie") au niveau des queues de distribution. Cette déviation entre la distribution théorique et la distribution empirique ne peut être imputée ni au générateur, car il passe les tests standards, ni à la longueur de l'échantillon, car elle correspond

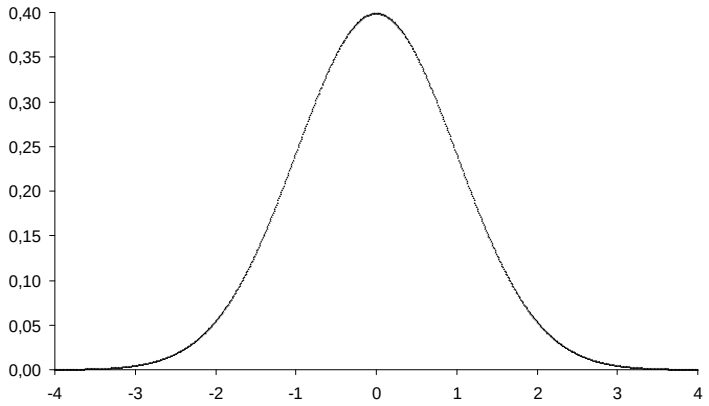


FIG. 2.1 – Distribution empirique des variables gaussiennes obtenues en appliquant la fomule (2.2) aux sorties de `Ran0` jusqu'à épuisement de la période du générateur ($2^{31}-1$ simulations).

à la période de `Ran0`. Ce phénomène est donc la conséquence d'une interaction non souhaitée entre l'algorithme de Box-Muller et le générateur. Afin de confirmer les observations précédentes, nous nous proposons d'analyser les fréquences de la distribution empirique de $X_1, \ldots, X_n$ sur des intervalles $[a, b[$ arbitrairement petits. On note $n_{[a,b[}$ le nombre de réalisations dans l'intervalle $[a, b[$. Sous l'hypothèse $\mathcal{H}_0$: "les X_k sont i.i.d. de loi $\mathcal{N}(0,1)$", on doit avoir

$$\mathrm{E}\left[n_{[a,b[}\right] = np_{_{[a,b[}} \text{ et } \mathrm{Var}\left[n_{[a,b[}\right] = np_{_{[a,b[}}\left(1 - p_{_{[a,b[}}\right),$$

où $p_{_{[a,b[}} = \Phi(b) - \Phi(a)$ et Φ est la fonction de répartition de la loi $\mathcal{N}(0,1)$. Lorsque n est suffisamment grand, on a l'approximation gaussienne pour le biais d'échantillonnage sur l'intervalle $[a, b[$:

$$d_{[a,b[} = \frac{n_{[a,b[} - \mathrm{E}\left[n_{[a,b[}\right]}{\sqrt{\mathrm{Var}\left[n_{[a,b[}\right]}} \underset{n\to\infty}{\sim} \mathcal{N}(0,1).$$

La p-value du test s'écrit :

$$\hat{\alpha}_{[a,b[} = 2\left(1 - \Phi\left(\left|d_{[a,b[}\right|\right)\right)$$

et on rejette l'hypothèse $\mathcal{H}_0$ pour tous les seuils α inférieurs à $\hat{\alpha}_{a,b}^{(n)}$. Le tableau ci-dessous donne les résultats obtenus pour différents intervalles lorsque $n = 2^{31}-1$.

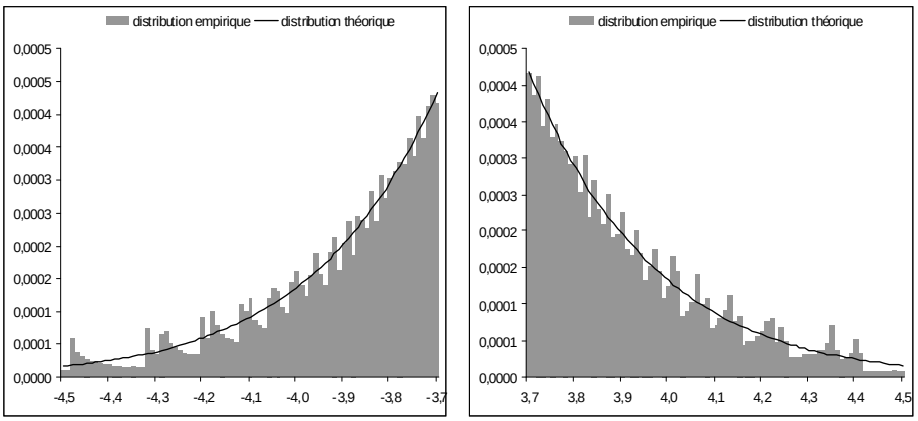


FIG. 2.2 – Queues de distribution (Box-Muller + `Ran0`).

intervalle $[a, b[$	observé $n_{[a,b[}$	attendu $np_{[a,b[}$	déviation $d_{[a,b[}$	p-value $\hat{\alpha}_{[a,b[}$
$-3.77 : -3.76$	6929	7157	-2.70	$6.93\text{E}-03$
$-3.76 : -3.75$	7794	7432	4.20	$2.67\text{E}-05$
$-3.75 : -3.74$	7200	7716	-5.87	$4.36\text{E}-09$
$-3.74 : -3.73$	8516	8010	5.65	$1.60\text{E}-08$
$-3.73 : -3.72$	7812	8314	-5.51	$3.59\text{E}-08$
$3.72 : 3.73$	8868	8314	6.08	$1.20\text{E}-09$
$3.73 : 3.74$	7357	8010	-7.30	$2.88\text{E}-13$
$3.74 : 3.75$	8156	7716	5.01	$5.44\text{E}-07$
$3.75 : 3.76$	7064	7432	-4.27	$1.95\text{E}-05$
$3.76 : 3.77$	7405	7157	2.93	$3.39\text{E}-03$

L'hypothèse d'adéquation à la loi gaussienne est rejetée sur tous les intervalles testés (les p-values sont toutes voisines de zéro) : on en déduit que l'effet Neave se traduit par une distorsion non négligeable de la densité de probabilité de la loi gaussienne.

Distorsion de la fonction de répartitition Nous allons à présent montrer que les déformations identifiées dans l'analyse précédente induisent un biais lors de l'estimation de probabilités cumulées (fonction de répartition et fonction de survie) au niveau des extrêmes de la distribution. Nous reprenons les raisonnements du paragraphe 2.2.2 en travaillant sur les fréquences (et non plus sur le nombre d'observations) et en remplaçant a par $-\infty$ (resp. b par $+\infty$) pour les intervalles situés dans la partie négative (resp. dans la partie positive) de la distribution. Les résultats obtenus sont présentés dans le tableau suivant.

probabilité à estimer	probabilité mesurée	probabilité attendue	déviation	p-value
$P\{X < -3.8\}$	0.00704%	0.00723%	-10.36	$3.79\text{E}-25$
$P\{X < -3.9\}$	0.00461%	0.00481%	-13.41	$5.40\text{E}-41$
$P\{X < -4.0\}$	0.00298%	0.00317%	-15.14	$8.27\text{E}-52$
$P\{X < -4.1\}$	0.00189%	0.00207%	-17.83	$4.39\text{E}-71$
$P\{X < -4.2\}$	0.00112%	0.00133%	-27.03	$6.73\text{E}-161$
$P\{X \geq 3.8\}$	99.99259%	99.99277%	-9.75	$1.76\text{E}-22$
$P\{X \geq 3.9\}$	99.99501%	99.99519%	-12.20	$2.98\text{E}-34$
$P\{X \geq 4.0\}$	99.99666%	99.99683%	-14.50	$1.28\text{E}-47$
$P\{X \geq 4.1\}$	99.99778%	99.99793%	-15.91	$5.66\text{E}-57$
$P\{X \geq 4.2\}$	99.99849%	99.99867%	-22.60	$4.79\text{E}-113$

Les déviations mesurées sont significativement plus élevées que celles observées dans le test de déformation de la densité. Cela suggère que les effets des distorsions locales observées sur des intervalles d'amplitude faible se cumulent lors de l'estimation du poids des queues de distribution dans les régions extrêmes. On ne peut donc pas retenir l'hypothèse d'adéquation gaussienne pour l'échantillon généré.

L'effet Neave avec le Mersenne Twister En 1991, Tezuka [36] a observé un phénomène semblable à l'effet Neave lorsque la méthode de Box-Muller est combinée avec un générateur de Tausworthe, un précurseur du Mersenne Twister. On peut donc se demander si l'effet Neave se produit aussi lorsque `MT19937` est associé à la méthode Box-Muller. Etant donné la longueur de la période de ce générateur (4.32×10^{6001}), on ne peut évidemment pas reproduire l'expérience réalisée avec le générateur `Ran0`. En effet, il n'est pas envisageable d'épuiser la période de `MT19937` dans un temps raisonnable. Ainsi, l'ordinateur décrit au paragraphe 1.4.2 produit 10^7 nombres par seconde. Lorsque le soleil disparaîtra dans 5 milliards d'années, la machine aura simulé environ 10^{24} nombres "gaussiens", ce qui représente une fraction de l'ordre de 10^{-5977} de la période du Mersenne Twister.

Cependant, nous avons pu constater la présence d'effet Neave (parfois atténué) sur différents jeux de $2^{31} - 1$ simulations gaussiennes (comme dans le cas de l'expérience faite avec `Ran0`) obtenus en choisissant différentes valeurs de la graine pour amorcer le générateur `MT19937`. Les statistiques déterminées sur les tests menés avec le Mersenne Twister étant très similaires à celles présentées lors de l'étude de l'effet Neave avec `Ran0`, nous avons choisi de ne pas les faire figurer.

Du rôle des queues de distribution Les queues de distribution jouent un rôle fondamental dans les applications financières, car elles représentent les scénarios les plus extrêmes, donc les plus risqués et les plus redoutés par les opérateurs (krachs boursiers). Il est donc fondamental de mettre en oeuvre une

méthode numérique qui ne conduise pas à sous-estimer la probabilité d'un scénario catastrophe. Les déviations calculées dans l'étude du paragraphe 2.2.2 sont toutes négatives, ce qui prouve que les probabilités empiriques déterminées à partir de l'échantillon sous-estiment systématiquement les poids des queues de distribution calculés en évaluant la fonction de répartition (ou la fonction de survie) de la loi normale. En conséquence, l'algorithme de Box-Muller ne constitue pas la meilleure alternative pour simuler l'aléa de la dynamique des cours de bourse. Dans le paragraphe suivant, nous étudions une méthode plus performante pour échantillonner la loi gaussienne à partir d'un générateur aléatoire.

2.3 Simulation par inversion de la fonction de répartition

2.3.1 Fonction inverse gaussienne

Soit Φ la fonction de répartition de la loi normale standard. Elle est définie par

$$\Phi(x) = P\{G \leq x\} = \int_{-\infty}^{x} \varphi(z)\, dz, \quad x \in \mathbb{R}.$$

L'application Φ est clairement bijective (continue et strictement croissante) de $\mathbb{R}$ vers $]0,1[$. La fonction inverse gaussienne Φ^{-1} étant définie sur $]0,1[$, on peut construire une variable aléatoire réelle G à valeurs dans $\mathbb{R}$ en posant

$$G = \Phi^{-1}(U) \text{ avec } U \sim \mathcal{U}(0,1).$$

On vérifie que G suit une loi normale standard, i.e. G admet Φ pour fonction de répartition :

$$\forall x \in \mathbb{R},\ P\{G \leq x\} = P\left\{\Phi^{-1}(U) \leq x\right\} = P\{U \leq \Phi(x)\} = \Phi(x).$$

On en déduit que, pour simuler des réalisations i.i.d. de la loi $\mathcal{N}(0,1)$, il suffit de poser

$$G_k = \Phi^{-1}(U_k), \text{ avec } U_k \text{ i.i.d. } \mathcal{U}(0,1). \tag{2.3}$$

Cette technique est appelée méthode de simulation par inversion de la fonction de répartition[9]. Sa mise en oeuvre suppose que l'on soit capable d'approcher numériquement Φ^{-1}, car cette fonction n'admet pas d'expression analytique.

2.3.2 Méthode d'inversion de Acklam (2000)

Description de l'algorithme Acklam [1] propose d'approcher Φ^{-1} sur l'intervalle $]0, 0.5]$ puis de considérer la symétrie de la loi normale standard, soit

$$\Phi^{-1}(u) = -\Phi^{-1}(1-u), \text{ pour } u \in]0,1[, \tag{2.4}$$

pour étendre l'approximation à l'intervalle complémentaire en conservant la même précision.

[9] Notons que la méthode d'inversion de la fonction de répartition peut être utilisée pour simuler n'importe quelle loi de probabilité.

Pour $0.02425 < u \leq 0.5$ l'auteur modélise la fonction inverse gaussienne par une fonction rationnelle de $q = u - 1/2$ et $r = q^2$:

$$\Phi^{-1}(u) \simeq q \times \frac{a_1 r^5 + a_2 r^4 + a_3 r^3 + a_4 r^2 + a_5 r + a_6}{b_1 r^5 + b_2 r^4 + b_3 r^3 + b_4 r^2 + b_5 r + 1}.$$

Pour $0 < u < 0.02425$ (queue de distribution gauche), l'approximation est basée sur une fonction rationnelle de $q = (-2 \ln u)^{1/2}$:

$$\Phi^{-1}(u) \simeq \frac{c_1 q^5 + c_2 q^4 + c_3 q^3 + c_4 q^2 + c_5 q + c_6}{d_1 q^4 + d_2 q^3 + d_3 q^2 + d_4 q + 1}.$$

Les jeux de coefficients a_k, b_k, c_k et d_k sont donnés ci-dessous :

$$\begin{array}{rr@{\qquad}rr}
a_1 = & -3.969683028665376\text{E}+01 & c_1 = & -7.784894002430293\text{E}-03 \\
a_2 = & 2.209460984245205\text{E}+02 & c_2 = & -3.223964580411365\text{E}-01 \\
a_3 = & -2.759285104469687\text{E}+02 & c_3 = & -2.400758277161838\text{E}+00 \\
a_4 = & 1.383577518672690\text{E}+02 & c_4 = & -2.549732539343734\text{E}+00 \\
a_5 = & -3.066479806614716\text{E}+01 & c_5 = & 4.374664141464968\text{E}+00 \\
a_6 = & 2.506628277459239\text{E}+00 & c_6 = & 2.938163982698783\text{E}+00
\end{array}$$

$$\begin{array}{rr@{\qquad}rr}
b_1 = & -5.447609879822406\text{E}+01 & d_1 = & 7.784695709041462\text{E}-03 \\
b_2 = & 1.615858368580409\text{E}+02 & d_2 = & 3.224671290700398\text{E}-01 \\
b_3 = & -1.556989798598866\text{E}+02 & d_3 = & 2.445134137142996\text{E}+00 \\
b_4 = & 6.680131188771972\text{E}+01 & d_4 = & 3.754408661907416\text{E}+00 \\
b_5 = & -1.328068155288572\text{E}+01 &
\end{array}$$

Précision et rapidité de la méthode Nous avons évalué[10] la fonction de Acklam sur un ensemble de 10^9 points, équidistribués dans le segment unité et définis par :

$$u_0 = 0 \text{ et } \forall k \geq 1,\ u_k = u_{k-1} + 10^{-9}.$$

L'algorithme de Acklam a effectué l'ensemble des calculs en 375 secondes. Il produit donc des nombres au rythme de 2.67 millions par seconde.

Supposons que Φ^{-1} constitue une très bonne approximation de la fonction inverse gaussienne, alors $\Phi \circ \Phi^{-1}$ est "pratiquement" la fonction identité et l'on doit avoir

$$e_\infty \stackrel{\text{def}}{=} \max_{k=1,\ldots,10^9} \left\{ \frac{\left| \Phi \circ \Phi^{-1}(u_k) - u_k \right|}{u_k} \right\} \simeq 0.$$

Comme la fonction de répartition gaussienne n'admet pas d'expression analytique, on remplace Φ par l'excellente approximation de West [37] et l'on obtient :

$$e_\infty = 2.48 \times 10^{-15} \simeq 0.$$

La méthode est donc extrêmement précise, ce qui confirme les conclusions de l'auteur [1] et de Jäckel ([11] p. 11).

[10] La machine utilisée est décrite au paragraphe 1.4.2.

2.3.3 Atténuation de l'effet Neave

La simulation par inversion de la fonction de répartition est une méthode plus naturelle que la méthode de Box-Muller. Une conséquence de cela est l'atténuation considérable des phénomènes pathologiques identifiés au paragraphe précédent (effet Neave). Nous avons reproduit l'expérience du paragraphe 2.2.2 en combinant l'inversion de Acklam et le générateur `Ran0`. Sur la figure 2.3, on

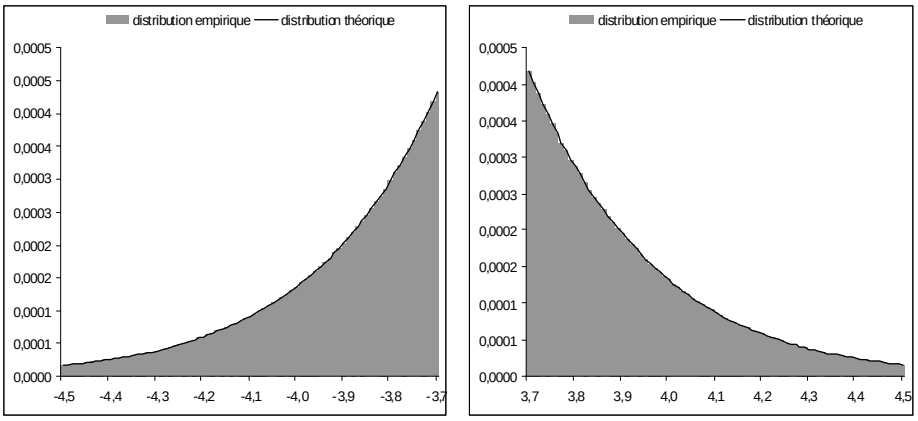


FIG. 2.3 – Queues de distribution (Acklam + `Ran0`).

remarque que l'histogramme s'ajuste parfaitement avec la distribution théorique de la loi normale standard au niveau des queues de distribution.

Disparition du biais d'échantillonage de la densité Afin de confirmer cette observation graphique, nous donnons ci-dessous un tableau d'analyse des fréquences de l'échantillon.

intervalle $[a, b[$	observé $n_{[a,b[}$	attendu $np_{[a,b[}$	déviation $d_{[a,b[}$	p-value $\hat{\alpha}_{[a,b[}$
$-3.77 : -3.76$	7157	7157	0.00	100.00%
$-3.76 : -3.75$	7432	7432	0.00	100.00%
$-3.75 : -3.74$	7715	7716	-0.01	99.20%
$-3.74 : -3.73$	8010	8010	0.00	100.00%
$-3.73 : -3.72$	8314	8314	0.00	100.00%
$3.72 : 3.73$	8314	8314	0.00	100.00%
$3.73 : 3.74$	8010	8010	0.00	100.00%
$3.74 : 3.75$	7715	7716	-0.01	99.20%
$3.75 : 3.76$	7432	7432	0.00	100.00%
$3.76 : 3.77$	7157	7157	0.00	100.00%

L'hypothèse d'adéquation à la loi gaussienne est largement retenue pour tous les intervalles testés (les p-values sont toutes voisines de 100%), ce qui prouve que le biais d'échantillonnage est à présent négligeable.

Disparition du biais d'échantillonage des probabilités cumulées Nous complétons notre étude en mesurant les biais d'échantillonage au niveau des probabilités cumulées dans les extrêmes des queues de distribution. Les résultats obtenus sont présentés dans la tableau ci-après.

probabilité à estimer	probabilité mesurée	probabilité attendue	déviation	p-value
$P\{X < -3.8\}$	0.00723475%	0.00723480%	$-3.15\text{E}-03$	99.75%
$P\{X < -3.9\}$	0.00480958%	0.00480963%	$-3.46\text{E}-03$	99.72%
$P\{X < -4.0\}$	0.00316706%	0.00316712%	$-5.65\text{E}-03$	99.55%
$P\{X < -4.1\}$	0.00206572%	0.00206575%	$-3.13\text{E}-03$	99.75%
$P\{X < -4.2\}$	0.00133454%	0.00133457%	$-4.59\text{E}-03$	99.63%
$P\{X \geq 3.8\}$	99.99276521%	99.99276520%	$6.12\text{E}-04$	99.95%
$P\{X \geq 3.9\}$	99.99519037%	99.99519037%	$3.49\text{E}-04$	99.97%
$P\{X \geq 4.0\}$	99.99683290%	99.99683288%	$1.82\text{E}-03$	99.86%
$P\{X \geq 4.1\}$	99.99793423%	99.99793425%	$-1.62\text{E}-03$	99.87%
$P\{X \geq 4.2\}$	99.99866541%	99.99866543%	$-1.31\text{E}-03$	99.90%

Les p-values calculées pour chaque intervalle testé sont toutes supérieures à 99.5% (alors qu'elles sont toutes nulles dans le cas de la méthode de Box-Muller), ce qui prouve que l'algorithme de Acklam permet de simuler les queues de distribution avec une grande précision.

Choix d'une méthode de simulation Nous avons établi que la méthode d'inversion de la fonction de répartition permet de réduire considérablement les biais d'échantillonnage induits par l'algorithme de Box-Muller, ce qui est particulièrement intéressant pour les applications financières. C'est pourquoi nous proposons d'utiliser systématiquement l'approximation de Acklam pour simuler des échantillons gaussiens sans perturber les queues de distribution.

3 Méthode de Monte Carlo

Soit $(\Omega, \mathcal{T}, P)$ un espace probabilisé, X une variable aléatoire à valeurs dans $\mathbb{R}^s$ muni de la tribu borélienne $\mathcal{B}_{\mathbb{R}^s}$ et $h : \mathbb{R}^s \to \mathbb{R}$ une application mesurable. Le problème est d'évaluer numériquement l'intégrale

$$I \overset{\text{def}}{=} \mathrm{E}\left[h\left(\mathrm{X}\right)\right] = \int_{\Omega} h\left(\mathrm{X}\right) dP, \tag{3.1}$$

lorsque $h\left(\mathrm{X}\right)$ est P-intégrable, i.e. $h\left(\mathrm{X}\right) \in \boldsymbol{L}^1\left(\Omega, \mathcal{T}, P\right)$.

3.1 Calcul d'espérance par simulation

La méthode de Monte Carlo consiste à générer N copies indépendantes de X (notées $\mathrm{X}_1, \ldots, \mathrm{X}_N$) puis à former l'approximation

$$I \underset{N\to\infty}{\simeq} \hat{I}_N \text{ où } \hat{I}_N \overset{\text{def}}{=} \frac{1}{N}\sum_{n=1}^{N} h(\mathrm{X}_n). \tag{3.2}$$

Un simple calcul d'espérance montre que $\hat{I}_N$ est un estimateur sans biais de I et la loi forte des grands nombres garantit que cet estimateur est fortement consistant (i.e. $\hat{I}_N$ converge presque sûrement vers I lorsque $N \to \infty$).

3.1.1 Construction d'un intervalle de confiance pour le résultat

Majoration de l'erreur probabiliste Dès que $h(\mathrm{X}) \in \boldsymbol{L}^2(\Omega, \mathcal{T}, P)$, le théorème de la limite centrale s'applique :

$$\sqrt{N}\left(\hat{I}_N - I\right) \underset{N\to\infty}{\overset{\text{loi}}{\to}} \mathcal{N}\left(0, \sigma^2\right) \text{ où } \sigma^2 \overset{\text{def}}{=} \mathrm{Var}\left[h(\mathrm{X})\right].$$

On peut alors construire un intervalle de confiance au niveau $1-\alpha$:

$$P\left\{\left|\hat{I}_N - I\right| < q_{1-\alpha/2}\,\frac{\sigma}{\sqrt{N}}\right\} \underset{N\to\infty}{=} 1-\alpha, \tag{3.3}$$

où $q_{1-\alpha/2}$ désigne le quantile d'ordre $1-\alpha/2$ de la loi normale standard. L'erreur d'intégration est donc majorée par la quantité

$$\varepsilon_N = q_{1-\alpha/2}\,\frac{\sigma}{\sqrt{N}} \tag{3.4}$$

avec une probabilité de $1-\alpha$ et il subsiste un risque α que l'erreur soit supérieure à ce seuil.

Estimation de l'erreur commise Dans la plupart des cas pratiques, σ^2 est inconnue. On calcule alors la variance empirique modifiée de l'échantillon :

$$\hat{\sigma}_N^2 = \frac{1}{N-1}\sum_{n=1}^{N}\left(h(\mathrm{X}_n) - \hat{I}_N\right)^2. \tag{3.5}$$

Cette quantité est un estimateur fortement consistant de la variance de $h(\mathrm{X})$ et, dès que N est suffisamment grand, on estime l'erreur d'intégration (3.4) en remplaçant σ par $\hat{\sigma}_N$:

$$\hat{\varepsilon}_N = q_{1-\alpha/2}\,\frac{\hat{\sigma}_N}{\sqrt{N}}. \tag{3.6}$$

Cette quantité permet de mesurer la qualité de l'approximation de I.

3.1.2 Avantages et inconvénients de l'approche Monte Carlo

La méthode de Monte Carlo possède trois avantages : (i) elle est très facile à implémenter pour peu que l'on sache simuler la loi de X, (ii) elle permet d'intégrer des fonctions irrégulières et de construire une estimation réaliste de l'erreur commise, (iii) elle converge en $\mathcal{O}\left(N^{-1/2}\right)$ indépendamment de la dimension s du problème, ce qui constitue un atout incontestable par rapport aux quadratures classiques qui convergent comme $\mathcal{O}\left(N^{-2/s}\right)$ et deviennent impraticables dès que $s \geq 5$.

Toutefois, cette approche présente deux défauts majeurs : (i) la majoration de l'erreur est probabiliste, de sorte qu'il subsiste toujours une incertitude (faible) quant à la vraie valeur de l'intégrale et (ii) la vitesse de convergence en $N^{-1/2}$ s'avère "relativement lente". En effet, si l'on souhaite une précision de l'ordre de 10^{-2}, on doit choisir $N \simeq 10^4$ et pour espérer réduire l'erreur d'un facteur 10, il faut multiplier la taille de l'échantillon par 100, ce qui induit un accroissement significatif (et inacceptable) des temps de calcul.

3.2 Techniques de réduction de variance

3.2.1 Efficacité d'un estimateur

Supposons que l'on dispose de deux fonctions h et g telles que $\mathrm{E}\left[h\left(\mathrm{X}\right)\right] = \mathrm{E}\left[g\left(\mathrm{X}\right)\right] = I$. Nous allons répondre à la question suivante : est-il préférable d'utiliser des copies de $h\left(\mathrm{X}\right)$ (méthode $\mathcal{H}$) ou des copies de $g\left(\mathrm{X}\right)$ (méthode $\mathcal{G}$) pour construire l'estimateur Monte Carlo de I ? Intuitivement, la méthode $\mathcal{G}$ sera préférée à la méthode $\mathcal{H}$ si $\mathcal{G}$ conduit systématiquement à une erreur plus faible que $\mathcal{H}$ dans le temps T (fixé arbitrairement).

Soit c_h le temps de calcul pour générer une seule copie de $h\left(\mathrm{X}\right)$, alors dans l'intervalle de temps T, on peut simuler $N_h\left(T\right) = T/c_h$ exemplaires de $h\left(\mathrm{X}\right)$. De la même manière, $N_g\left(T\right) = T/c_g$. On dit que $\mathcal{G}$ est plus efficace que $\mathcal{H}$ si

$$\forall T > 0, \quad \varepsilon_{N_g(T)} < \varepsilon_{N_h(T)}, \tag{3.7}$$

où $\varepsilon_{N_g(T)}$ et $\varepsilon_{N_h(T)}$ sont obtenus en remplaçant N par $N_h\left(T\right)$ ou $N_g\left(T\right)$ dans la formule (3.4) :

$$\varepsilon_{N_h(T)} = \frac{q_{1-\alpha/2}}{\sqrt{T}}\ \sigma_h\sqrt{c_h}, \quad \varepsilon_{N_g(T)} = \frac{q_{1-\alpha/2}}{\sqrt{T}}\ \sigma_g\sqrt{c_g}.$$

Alors, le critère d'efficacité (3.7) est équivalent à

$$c_g\sigma_g^2 < c_h\sigma_h^2. \tag{3.8}$$

La quantité $c_h\sigma_h^2$ est donc une mesure de la qualité de l'estimateur $\mathcal{H}$. La relation (3.8) montre que l'on ne peut pas conclure que la méthode $\mathcal{H}$ est meilleure que la méthode $\mathcal{G}$ sur le simple critère $\sigma_g^2 < \sigma_h^2$, car on doit aussi tenir compte du temps

de calcul nécessaire pour simuler une copie de chaque variable. Cependant, dans la majorité des cas où deux méthodes d'estimation $\mathcal{H}$ et $\mathcal{G}$ seront envisagées, on aura $c_g \simeq c_h$ et $\sigma_g^2 << \sigma_h^2$ ou bien $\sigma_g^2 >> \sigma_h^2$. Il est clair que dans cette situation, il faudra mettre en oeuvre l'estimateur de faible variance pour améliorer la convergence de l'algorithme.

L'objectif des méthodes dites de réduction de variance [8] est précisément de déterminer un autre estimateur de I, plus efficace que l'estimateur naturel au sens du critère (3.8). Si l'on considère que l'accroissement du temps de calcul induit par le choix d'un autre estimateur est marginal, le problème revient à déterminer une fonction g telle que

$$\mathrm{E}\left[g\left(\mathrm{X}\right)\right] = I \text{ et } \mathrm{Var}\left[g\left(\mathrm{X}\right)\right] < \mathrm{Var}\left[h\left(\mathrm{X}\right)\right]. \tag{3.9}$$

La plupart des techniques de réduction de variance sont étroitement liées à l'expression analytique de h et à la loi de X, de sorte que (i) chaque intégrale doit être traitée comme un cas particulier et (ii) il n'est pas possible d'envisager une approche universelle pour réduire la variance de façon systématique.

Nous présentons la méthode des variables antithétiques et la méthode adaptative, car elles reposent sur des hypothèses suffisamment générales pour être mises en oeuvre de manière quasi-systématique, en particulier lorsque le vecteur X est à composantes gaussiennes.

3.2.2 Méthode des variables antithétiques

Description de la méthode Cette technique, due à Hammersley et Morton [9], consiste à exploiter les symétries de la loi de X pour réduire la variance. L'idée est de trouver une transformation $T : \mathbb{R}^s \to \mathbb{R}^s$ telle que : (i) $T\left(\mathrm{X}\right)$ ait même loi que X et (ii) $h\left(\mathrm{X}\right)$ et $h\left(T\left(\mathrm{X}\right)\right)$ soient négativement corrélées, i.e. $\mathrm{Cor}\left[h\left(\mathrm{X}\right), h\left(T\left(\mathrm{X}\right)\right)\right] < 0$ (on parle de variables antithétiques).

Dans ces conditions, l'estimateur Monte Carlo basé sur l'application $g : \mathbb{R}^s \to \mathbb{R}$ définie par

$$g\left(\mathrm{x}\right) = \frac{h\left(\mathrm{x}\right) + h\left(T\left(\mathrm{x}\right)\right)}{2}, \quad \mathrm{x} \in \mathbb{R}^s,$$

est un estimateur sans biais et fortement consistant de l'intégrale (3.1) et sa variance est plus faible que celle de l'estimateur naturel (3.2). En effet, comme $\mathrm{X} \sim T\left(\mathrm{X}\right)$ on a

$$\mathrm{E}\left[g\left(\mathrm{X}\right)\right] = \frac{1}{2}\mathrm{E}\left[h\left(\mathrm{X}\right) + h\left(T\left(\mathrm{X}\right)\right)\right] = I$$

et

$$\sigma_g^2 = \frac{1 + \mathrm{Cor}\left[h\left(\mathrm{X}\right), h\left(T\left(\mathrm{X}\right)\right)\right]}{2}\ \sigma_h^2. \tag{3.10}$$

Etant donné que la corrélation est inférieure à 1, on a $\sigma_g^2 \leq \sigma_h^2$, de sorte que la variance de l'estimateur antithétique sera toujours inférieure à la variance de l'estimateur naturel. Si de plus, $\mathrm{Cor}\left[h\left(\mathrm{X}\right), h\left(T\left(\mathrm{X}\right)\right)\right] < 0$, il vient $\sigma_g^2 < \sigma_h^2/2$. Dans ce cas, le gain sur la variance est au moins égal à 50%.

Variables antithétiques dans le cas gaussien Si les composantes de X sont des variables gaussiennes centrées, alors $-\mathrm{X} \sim \mathrm{X}$ et l'on peut poser $T(\mathrm{X}) = -\mathrm{X}$. Si de plus h est une fonction monotone en chacun de ses arguments, alors $h(-\mathrm{X})$ est décroissante lorsque $h(\mathrm{X})$ est croissante et inversement. Les deux variables ont donc tendance à varier dans des sens opposés et elles forment une paire antithétique (voir Glasserman [8] p.207). Pour une généralisation de la méthode des variables antithétiques à des lois non gaussiennes, le lecteur pourra consulter Fishman et Huang [6].

3.2.3 Méthode de Monte Carlo adaptative

On suppose que X est un vecteur centré. Sous cette hypothèse, les variables aléatoires de la forme $g(\omega, \mathrm{X}) = h(\mathrm{X}) - \langle \omega, \mathrm{X} \rangle$ ($\omega \in \mathbb{R}^s$ et $\langle ., . \rangle$ est le produit scalaire euclidien sur $\mathbb{R}^s$) sont des estimateurs sans biais de I.

Un problème d'optimisation Parmi les candidats de la forme précédente, l'idée est de retenir celui qui minimise la variance de $g(\omega, \mathrm{X})$. On doit donc déterminer

$$\omega^* = \operatorname{argmin} \{\operatorname{Var}[g(\omega, \mathrm{X})] : \omega \in \mathbb{R}^s\} \tag{3.11}$$

en espérant que $\operatorname{Var}[g(\omega^*, \mathrm{X})] < \sigma_h^2$. Si tel est le cas, il devient intéressant d'intégrer $g(\omega^*, \mathrm{X})$ plutôt que $h(\mathrm{X})$.

On note $X_1, \ldots, X_s$ (resp. $\omega_1, \ldots, \omega_s$) les composantes de X (resp. de ω) et l'on pose $H(\omega) = \operatorname{Var}[g(\omega, \mathrm{X})]$. La condition du premier ordre du problème de minimisation (3.11) s'écrit

$$\frac{\partial H}{\partial \omega_i} = 2\left(\omega_i - \mathrm{E}\left[X_i h(\mathrm{X})\right]\right) = 0, \quad i = 1, \ldots, s.$$

Alors, les coefficients du vecteur ω^* sont donnés par

$$\omega_i^* = \mathrm{E}\left[X_i h(\mathrm{X})\right], \quad i = 1, \ldots, s.$$

Les ω_i^* s'exprimant sous la forme d'une espérance, ils peuvent être approchés par la méthode de Monte Carlo classique : si $\mathrm{X}_n = (X_{1,n}, \ldots, X_{s,n})'$ est une suite de variables i.i.d. selon la loi de X, alors

$$\hat{\omega}_{i,N}^* = \frac{1}{N} \sum_{n=1}^{N} X_{i,n} h(\mathrm{X}_n) \tag{3.12}$$

est un estimateur sans biais et fortement consistant de ω_i^*.

Construction de l'estimateur adaptatif La mise en oeuvre naturelle de cette approche consiste à lancer deux simulations Monte Carlo : (i) la première simulation permet d'approcher ω^* par $\hat{\omega}_N^* = \left(\hat{\omega}_{1,N}^*, \ldots, \hat{\omega}_{s,N}^*\right)'$, (ii) la seconde simulation permet d'estimer l'intégrale par $\frac{1}{N} \sum_{n=1}^{N} g(\hat{\omega}_N^*, \mathrm{X}_n)$. Cela n'est pas

envisageable dans la mesure où l'on souhaite conserver des temps de calcul raisonnables.

Bouchard [3] propose de contourner ce problème, en estimant simultanément l'intégrale et le vecteur optimal α^*. Pour cela, il considère l'estimateur adaptatif défini par :

$$\hat{I}_N^{\mathrm{AD}} = \frac{1}{N} \sum_{n=1}^{N} g\left(\hat{\omega}_{n-1}^*, \mathrm{X}_n\right), \tag{3.13}$$

où l'on a pris la convention $\omega_0^* = 0$. L'idée est d'utiliser le vecteur optimal de l'étape $n-1$ (ω_{n-1}^*) pour simuler le terme $\left\langle \hat{\omega}_{n-1}^*, \mathrm{X}_n \right\rangle$ à l'étape n puis d'estimer une nouvelle approximation $\hat{\omega}_n^*$ qui servira aux calculs de l'étape $n+1$, etc. Cette approche est adaptative, dans le sens où l'estimateur se modifie d'une étape à l'autre. Notons que la suite de terme général $g\left(\hat{\omega}_{n-1}^*, \mathrm{X}_n\right)$ n'est plus i.i.d. car $\hat{\omega}_{n-1}^*$ dépend de toutes les simulations précédentes. On ne peut donc plus appliquer directement la loi forte des grands nombres et le théorème de la limite centrale.

Lorsque h est au plus à croissance exponentielle, Arouna [2] démontre le résultat suivant.

Proposition 3.1 *La suite de terme général $\hat{I}_N^{\mathrm{AD}}$ est un estimateur sans biais et fortement consistant de I. De plus, on a un résultat équivalent au théorème de la limite centrale :*

$$\sqrt{N}\left(\hat{I}_N^{\mathrm{AD}} - I\right) \underset{N\to\infty}{\overset{\text{loi}}{\to}} \mathcal{N}\left(0, \mathrm{Var}\left[g\left(\omega^*, \mathrm{X}\right)\right]\right).$$

Enfin, la "variance empirique" définie par

$$\hat{\sigma}_N^2 = \frac{1}{N-1} \sum_{n=1}^{N} \left(g\left(\hat{\omega}_{n-1}^*, \mathrm{X}_n\right) - \hat{I}_N^{\mathrm{AD}}\right)^2 \tag{3.14}$$

est un estimateur fortement consistant de $\mathrm{Var}\left[g\left(\omega^*, \mathrm{X}\right)\right]$.

Cette proposition montre que l'on peut construire un intervalle de confiance pour I comme dans la méthode de Monte Carlo classique et procéder ainsi à une analyse pertinente de l'erreur d'estimation.

4 Evaluation d'options par simulation

Dans cette dernière partie nous appliquons les outils de simulation présentés dans les sections précédentes pour évaluer une option exotique portant sur la trajectoire d'un sous-jacent de type action ou indice.

4.1 Présentation du problème

4.1.1 L'approche risque-neutre

On se place dans le cadre d'analyse de Black et Scholes [15] : le sous-jacent ne détache pas de dividende et sa dynamique dans l'univers risque-neutre est donnée par

$$S_t = s_0 \exp\left(\left(r - \frac{\sigma^2}{2}\right) t + \sigma B_t\right), \quad t \geq 0, \tag{4.1}$$

où s_0 (le cours de l'actif observé à l'instant 0), r (le taux sans risque de l'économie) et σ (la volatilité de l'actif) sont des constantes positives. $\{B_t : t \geq 0\}$ est un mouvement Brownien standard sous la mesure risque-neutre.

On considère le problème de l'évaluation d'une option asiatique géométrique :

$$\Pi_0 = \mathrm{E}\left[\mathrm{e}^{-rT}\ \Pi_T\right] \text{ où } \Pi_T = \max\left(\bar{S}_T - K, 0\right). \tag{4.2}$$

T et K désignent respectivement la date d'expiration et le prix d'exercice de l'option, $\bar{S}_T = \left(\prod_{k=1}^{m} S_{t_k}\right)^{1/m}$ est la moyenne géométrique des cours du sous-jacent aux dates d'observation $0 \leq t_1 < \cdots < t_m \leq T$. Notre objectif est d'estimer Π_0 en appliquant les méthodes de simulation présentées dans ce travail.

4.1.2 Formule analytique pour l'option asiatique géométrique

Lorsque le sous-jacent suit une loi log-normale (ce qui est le cas dans le modèle retenu), le prix d'une option asiatique géométrique peut être déterminé de manière analytique (cf. Bruno [4]) :

$$\Pi_0 = \mathrm{e}^{-rT}\left(s_0 \mathrm{e}^{a+b^2/2}\Phi\left(\frac{a - \ln K}{b} + b\right) - K\Phi\left(\frac{a - \ln K}{b}\right)\right), \tag{4.3}$$

où

$$a = \left(r - \frac{\sigma^2}{2}\right)\sum_{k=1}^{m}\left(1 - \frac{k-1}{m}\right)h_k \text{ et } b = \sigma\sqrt{\sum_{i=1}^{m}\left(1 - \frac{k-1}{m}\right)^2 h_k}.$$

Etant donné que nous connaissons le prix de l'option sous une forme explicite, nous pouvons quantifier l'erreur d'estimation et ainsi comparer les performances des méthodes proposées.

4.2 Simulation du sous-jacent

4.2.1 Simulation incrémentale

A partir de la formule (4.1) on obtient très facilement la relation suivante :

$$S_{t_k} = S_{t_{k-1}} \exp\left(\left(r - \frac{\sigma^2}{2}\right)(t_k - t_{k-1}) + \sigma\left(B_{t_k} - B_{t_{k-1}}\right)\right), \quad k = 1, \ldots, m. \tag{4.4}$$

On rappelle que les incréments du mouvement Brownien sont mutuellement indépendants et que $B_{t_k} - B_{t_{k-1}} \sim \mathcal{N}(0, \sqrt{t_k - t_{k-1}})$ [15]. On peut donc les simuler en posant

$$B_{t_k} - B_{t_{k-1}} = \sqrt{t_k - t_{k-1}} G_k, \quad k = 1, \ldots, m, \tag{4.5}$$

où $G_1, \ldots, G_m$ sont i.i.d. de loi $\mathcal{N}(0, 1)$.

En injectant la formule (4.5) dans la relation (4.4), on obtient une procédure récursive pour simuler le cours du sous-jacent aux dates $t_1, \ldots, t_m$:

$$\tilde{S}_0 = s_0, \quad \tilde{S}_k = \tilde{S}_{k-1} \exp\left(\left(r - \frac{\sigma^2}{2}\right) h_k + \sigma\sqrt{h_k} G_k\right), \tag{4.6}$$

où l'on a posé $h_k = t_k - t_{k-1}$. Ce schéma de simulation est exact dans la mesure où le m-uplet $\left(\tilde{S}_1, \ldots, \tilde{S}_m\right)$ suit la même loi de probabilité que $(S_{t_1}, \ldots, S_{t_m})$. L'estimateur Monte Carlo naturel est simplement défini comme le payoff actualisé :

$$X_{\mathrm{MC}} = \mathrm{e}^{-rT}\ \Pi_T(\tilde{S}_1, \ldots, \tilde{S}_m).$$

4.2.2 Simulation antithétique

Pour obtenir une trajectoire antithétique, on remplace G_k par $-G_k$ dans la formule (4.6) :

$$\tilde{S}_0^- = s_0, \quad \tilde{S}_k^- = \tilde{S}_{k-1}^- \exp\left(\left(r - \frac{\sigma^2}{2}\right) h_k - \sigma\sqrt{h_k} G_k\right). \tag{4.7}$$

Il est donc facile d'obtenir une trajectoire antithétique du sous-jacent en réutilisant les points déjà simulés. L'estimateur Monte Carlo antithétique s'écrit

$$X_{\mathrm{MC-AV}} = \mathrm{e}^{-rT} \frac{\Pi_T(\tilde{S}_1, \ldots, \tilde{S}_m) + \Pi_T(\tilde{S}_1^-, \ldots, \tilde{S}_m^-)}{2} = \frac{X_{\mathrm{MC}} + X_{\mathrm{AV}}}{2},$$

où $X_{\mathrm{AV}} = \mathrm{e}^{-rT}\ \Pi_T(\tilde{S}_1^-, \ldots, \tilde{S}_m^-)$ désigne le payoff actualisé obtenu à partir de la trajectoire antithétique.

4.2.3 Simulation adaptative

L'estimateur Monte Carlo adaptatif est de la forme suivante :

$$X_{\mathrm{MC-AD}} = \mathrm{e}^{-rT}\ \Pi_T(\tilde{S}_1, \ldots, \tilde{S}_m) - \sum_{k=1}^{m} \omega_k^* G_k = X_{\mathrm{MC}} - X_{\mathrm{AD}},$$

où $X_{\mathrm{AD}} = \sum_{k=1}^{m} \omega_k^* G_k$ représente le terme de correction adaptative. Les poids $\{\omega_k^*\}$ seront ré-estimés à chaque itération selon le processus décrit au paragraphe 3.2.3.

4.3 Tests comparatifs

4.3.1 Mise en oeuvre des tests

Les paramètres retenus pour les applications numériques sont les suivants : $r = 4\%$, $\sigma = 20\%$, $T = 10$ et $s_0 = K = 100$. Les dates $t_1, \ldots, t_m$ sont définies par $t_k = kT/m$ avec $m = 120$, ce qui correspond à une fréquence d'observation mensuelle. La formule analytique (4.3) nous donne $\Pi_0 \simeq 17.8958$.

Nous avons mis en oeuvre la méthode de Monte Carlo classique (MC), la méthode antithétique (MC-AV) et la méthode adaptative (MC-AD) pour les valeurs de N suivantes : $5000n$, $n = 1, \ldots, 50$. Les variables gaussiennes ont été simulées en appliquant la méthode d'inversion de Acklam aux sorties du générateur Mersenne Twister. La figure 4.1 présente deux graphes de convergence des estimateurs obtenus pour deux choix différents de l'état initial. Une simple com-

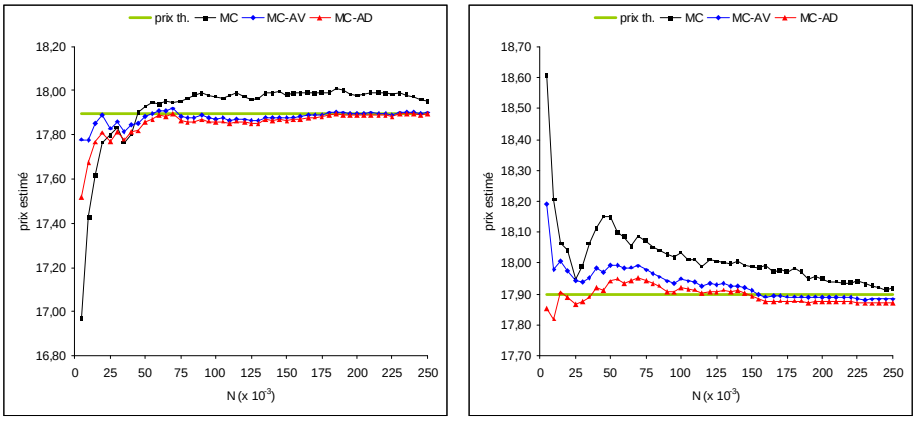


FIG. 4.1 – Diagrammes de convergence des estimateurs vers le prix théorique $\Pi_0 \simeq 17.8958$. Les deux figures ont été construites à partir de deux graines différentes.

paraison de la figure de gauche et de la figure de droite montre que deux graines distinctes peuvent conduire à des courbes d'approximation très différentes (les courbes de gauche présentent une tendance croissante, tandis que les courbes de droite ont une tendance globalement décroissante). Par ailleurs, sur la figure de gauche, les estimateurs MC-AV et MC-AD convergent vers le prix cherché au bout de 70000 itérations environ, tandis que la convergence de ces mêmes estimateurs a lieu au bout de 160000 simulations sur la figure de droite. Cela montre que la vitesse de convergence observée dépend fortement de l'amorce du générateur. En pratique, il existe deux solutions équivalentes pour s'affranchir du "risque d'amorce" : (i) choisir une seule graine mais augmenter le nombre de d'itérations ou (ii) lancer différents jeux de simulations à partir de graines

distinctes[11] et prendre la moyenne des résultats obtenus comme approximation de la valeur cherchée.

Remarque 4.1 Dans les deux exemples présentés, l'estimateur naturel (MC) converge plus lentement vers la solution du problème que les deux autres estimateurs. Soulignons également que les courbes de convergence des estimateurs MC-AV et MC-AD ont la même forme quel que soit l'état initial choisi. Ce phénomène s'explique par le fait que ces méthodes d'estimation reposent toutes deux sur l'utilisation d'une variable auxiliaire anticorrélée avec l'estimateur naturel (voir paragraphe suivant).

4.3.2 Explication de la réduction de variance par la corrélation entre les estimateurs

En utilisant les différentes réalisations des variables X_{MC}, X_{AV} et X_{AD} obtenues lors des simulations qui ont permis de générer le graphique de droite, nous avons estimé les coefficients de corrélation empiriques suivants :

$$\begin{aligned} \rho_{\mathrm{MC-AV}} &= \mathrm{Cor}\left[X_{\mathrm{MC}}, X_{\mathrm{AV}}\right] = -0.5026, \\ \rho_{\mathrm{MC-AD}} &= \mathrm{Cor}\left[X_{\mathrm{MC}}, X_{\mathrm{AD}}\right] = 0.8634. \end{aligned}$$

En appliquant la formule (3.10) on obtient

$$\sigma^2_{\mathrm{MC-AV}} = \frac{1+\rho_{\mathrm{MC-AV}}}{2}\sigma^2_{\mathrm{MC}} \simeq 0,249\ \sigma^2_{\mathrm{MC}} \simeq \sigma^2_{\mathrm{MC}}/4,$$

où σ^2_{MC} désigne la variance de l'estimateur naturel X_{MC}. La méthode antithétique doit donc diviser la variance par 4 environ.

De même en considérant que les estimateurs X_{MC} et X_{AD} ont des variances comparables (i.e. $\sigma^2_{\mathrm{MC}} \simeq \sigma^2_{\mathrm{AD}}$), on peut écrire :

$$\begin{aligned} \sigma^2_{\mathrm{MC-AD}} &= \sigma^2_{\mathrm{MC}} + \sigma^2_{\mathrm{AD}} - 2\ \sigma_{\mathrm{MC}}\ \sigma_{\mathrm{AD}}\ \rho_{\mathrm{MC-AD}} \\ &\simeq 2\ \sigma^2_{\mathrm{MC}}\left(1-\rho_{\mathrm{MC-AD}}\right) \simeq 0,273\ \sigma^2_{\mathrm{MC}} \simeq \sigma^2_{\mathrm{MC}}/4. \end{aligned}$$

Ce second résultat montre que l'estimateur adaptatif doit réduire la variance d'un facteur voisin de 4 (sur l'exemple choisi). Nous allons à présent confirmer cette analyse préliminaire en analysant le comportement des intervalles de confiance.

4.3.3 Effet de la réduction de variance sur la précision asymptotique

On désigne par $\hat{\Pi}_0^N$ et $\hat{\sigma}_N^2$ le prix estimé et la variance empirique de l'estimateur au bout de N itérations. Alors, l'intervalle de confiance pour le prix au seuil de 95% est défini par

$$\hat{I}_N = \left[\hat{\Pi}_0^N - 1.96 \times \frac{\hat{\sigma}_N}{\sqrt{N}}, \hat{\Pi}_0^N + 1.96 \times \frac{\hat{\sigma}_N}{\sqrt{N}}\right].$$

[11] Pour une discussion sur le choix de la graine dans les applications scientifiques, le lecteur pourra consulter l'article de Marsaglia [23].

Comme nous l'avons exposé au paragraphe 3.1.1, le prix théorique appartient à cet intervalle avec une probabilité de 95% et l'amplitude de l'intervalle de confiance, i.e. l'incertitude autour du prix cherché, diminue proportionnellement à $1/\sqrt{N}$. La figure 4.2 illustre ce phénomène pour les trois estimateurs[12].

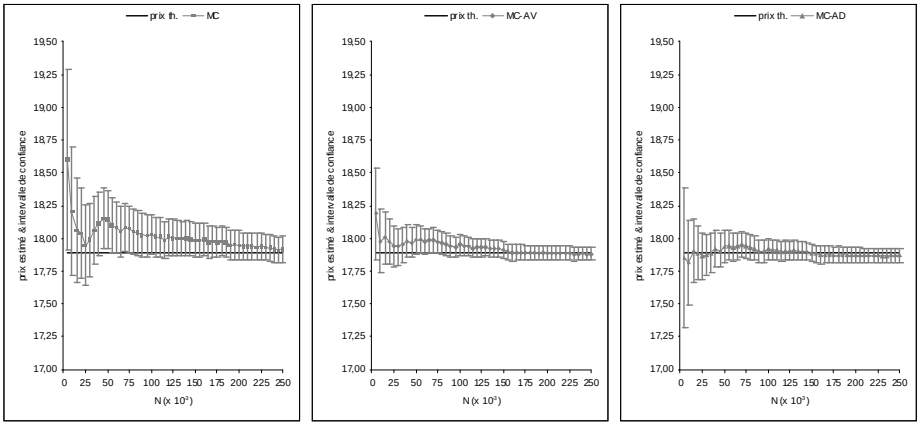


FIG. 4.2 – Réduction de l'amplitude de l'intervalle de confiance (donc de l'incertitude) sur le prix cherché pour l'estimateur naturel (à gauche), l'estimateur antithétique (au milieu) et l'estimateur adaptatif (à droite).

Comme attendu, lorsque N augmente, les bornes de l'intervalle de confiance de chacun des estimateurs se resserrent. Les effets de la réduction de variance sont particulièrement nets sur les deux derniers graphiques où les intervalles de confiance sont environ moitié moins larges que sur le premier graphique qui représente l'estimateur naturel MC. Afin d'appréhender l'impact des méthodes de réduction de variance proposées, nous donnons ci-dessous les prix estimés, l'intervalle de confiance associé et la largeur de l'intervalle de confiance au bout de $N = 250000$ simulations.

estimateur	prix estimé	intervalle de confiance	largeur
MC	17.9176	$[17.8198, 18.0155]$	0.1957
MC-AV	17.8823	$[17.8336, 17.9311]$	0.0975
MC-AD	17.8709	$[17.8206, 17.9212]$	0.1006

La largeur de l'intervalle de confiance avec la méthode MC est 0.1957, tandis qu'elle est environ moitié moindre avec les méthodes MC-AV et MC-AD. Cela confirme que les techniques de réduction de variance mises en oeuvre ont bien atteint leur objectif en réduisant la variabilité de l'estimateur (i.e. l'incertitude sur le prix cherché) d'un facteur 2, soit la variance par 4.

[12] Les courbes de convergence utilisées sont celles du graphique de droite dans la figure 4.1.

Remarque 4.2 En pratique, on met en oeuvre une méthode de simulation numérique quand on ne sait pas déterminer le prix sous une forme analytique. Nous avons choisi un exemple où la solution du problème est connue ($\Pi_0 \simeq 17.8958$), ce qui permet de déterminer l'erreur d'approximation ($\hat{\varepsilon} = |\hat{\Pi}_0^N/\Pi_0 - 1|$) pour chaque estimateur testé. En utilisant les résultats du tableau précédent, on trouve $\hat{\varepsilon}_{\mathrm{MC}} \simeq 0.12\%$ pour l'estimateur MC, $\hat{\varepsilon}_{\mathrm{MC-AV}} \simeq 0.08\%$ pour l'estimateur MC-AV et $\hat{\varepsilon}_{\mathrm{MC-AD}} \simeq 0.14\%$ pour l'estimateur MC-AD. Ces valeurs montrent que les méthodes de réduction de variance ne donnent pas obligatoirement une approximation plus précise que l'estimateur naturel. Par contre, elles permettent de diminuer l'incertitude autour de la valeur estimée, ce qui accroît la probabilité de trouver un résultat plus proche de la solution du problème.

4.3.4 Efficacité des méthodes de réduction de variance

Pour chaque estimateur, nous avons déterminé l'effort calculatoire $c = T_N/N$ (où T_N est le temps nécessaire pour construire l'échantillon de taille N) et l'indice d'efficacité $c\hat{\sigma}_N^2$ tel que défini au paragraphe 3.2.1. Les résultats obtenus pour $N = 250000$ sont indiqués dans le tableau ci-dessous.

estimateur	variance ($\hat{\sigma}_N^2$)	effort calculatoire (c)	efficacité ($c\hat{\sigma}_N^2$)
MC	622.90	$1.4060\mathrm{E}-04$	$8.76\mathrm{E}-02$
MC-AV	154.60	$1.7441\mathrm{E}-04$	$2.70\mathrm{E}-02$
MC-AD	158.56	$1.9372\mathrm{E}-04$	$3.07\mathrm{E}-02$

L'estimateur MC-AV présente un indice d'efficacité légèrement meilleur que celui de l'estimateur MC-AD ($2.70\mathrm{E}-02$ pour MC-AV contre $3.07\mathrm{E}-02$ pour MC-AD), ce qui prouve que la méthode des variables antithétiques est préférable à la méthode adaptative sur l'exemple choisi. Dans tous les cas, les méthodes de réduction de variance sont plus efficaces que la méthode naturelle, au sens du critère (3.8). Il est donc particulièrement intéressant de les mettre en oeuvre.

Conclusion

Les profondes modifications de l'environnement financier au cours des dernières années (explosion des marchés de produits dérivés de toutes natures, entrée en vigueur de nouvelles normes comptables et réglementaires) ont fait des méthodes de simulation numérique un outil incontournable pour la gestion des risques. Dans ce contexte, nous nous sommes attachés à montrer comment élaborer une solution complète pour évaluer des produits dérivés par la méthode de Monte Carlo à partir des solutions théoriques proposées dans la littérature.

Dans un premier temps, nous avons envisagé le problème de la simulation du hasard par des moyens déterministes en comparant deux familles de générateurs pseudo-aléatoires uniformes, les générateurs linéaires congruentiels et les générateurs Mersenne Twister, plus récents et conçus pour exploiter l'architecture

binaire des ordinateurs. Notre étude a confirmé que cette nouvelle famille de générateurs présentait de solides atouts pour la simulation numérique intensive : le Mersenne Twister `MT19937` possède une période infinie (à l'échelle informatique), il produit des séquences bien équidistribuées et il s'exécute rapidement sur les machines standards.

Dans la seconde partie, nous avons étudié les méthodes de simulation de la loi gaussienne. Nous avons montré comment les transformations non-linéaires d'un jeu de variables uniformes (Box-Muller) pouvaient conduire à des effets de bord indésirables (effet Neave). C'est pourquoi nous avons choisi de simuler la loi normale standard en inversant la fonction de répartition. Cela nécessite d'approcher la fonction inverse gaussienne par un algorithme robuste, comme la méthode proposée par Acklam : cette approche est particulièrement rapide, très précise et elle atténue considérablement les artefacts de simulation.

Dans la troisième partie, nous avons présenté la méthode de Monte Carlo qui, en raison de sa simplicité et parce qu'elle nécessite des calculs intensifs et répétitifs, se prête particulièrement bien à une implémentation informatique. Nous avons par ailleurs étudié la technique des variables antithétiques et la technique adaptative. Ces deux approches pour réduire la variance reposent sur des hypothèses très générales et peuvent être mises en oeuvre de manière quasi-systématique. Cela est un atout considérable lorsqu'on envisage d'évaluer des produits dérivés aux payoffs très différents sans avoir à modifier l'algorithme de simulation pour l'adapter aux caractéristiques de chaque produit.

L'application financière de la dernière partie a permis de mettre en oeuvre les différents algorithmes étudiés dans ce travail pour évaluer une option exotique en simulant l'évolution des cours boursiers dans le cadre du modèle de Black et Scholes. Les tests réalisés ont montré comment les méthodes de réduction de variance permettent de contrôler l'incertitude sur l'erreur commise et d'accélérer la convergence de l'algorithme vers la valeur cherchée. Soulignons enfin que la méthode adaptative, qui permet d'ajuster les caractéristiques de l'estimateur en fonction des simulations réalisées, est une technique prometteuse. En effet, elle repose sur des hypothèses moins contraignantes que la technique antithétique et elle a donné, sur l'exemple traité, des résultats très satisfaisants.

Les recherches dans le domaine de la simulation numérique sur ordinateur sont loin d'être achevées, car l'accroissement de la puissance de calcul permet d'envisager des solutions toujours plus performantes, comme la simulation en parallèle. Une solution consiste à générer simultanément plusieurs trajectoires du sous-jacent sur différents processeurs (voir Pauletto [32]), ce qui permet de multiplier le nombre de réplications du payoff par le nombre de processeurs sur la grille de calcul. Une autre solution est de générer plusieurs nombres aléatoires simultanément sur un seul processeur "multi-coeurs" avec la technologie SIMD (Single Instruction Multiple Data). En utilisant cette technologie, Saito [34] a pu développer une version optimisée du Mersenne Twister, deux fois plus rapide que l'algorithme original, `MT19937`.

Références

[1] P.J. Acklam (2000). *An algorithm for computing the inverse normal cumulative distribution function*, Technical Paper, http :// home.online.no/ ~pjacklam/notes/invnorm/.

[2] B. Arouna (2004). *Adaptative Monte Carlo Method, A Variance Reduction Technique*, Monte Carlo Methods and Applications, Vol. 10, No. 1, pp. 1-24.

[3] B. Bouchard-Denize (2006). *Méthodes de Monte Carlo en Finance*, Notes de cours, Université de Paris VI.

[4] M.G. Bruno (1991). *Calculation methods for evaluating asian options*, Working Paper.

[5] D.N. Chorafas (2004). *Economic Capital Allocation with Basel II : Cost, Benefit and Implementation Procedures*, Elesevier Finance.

[6] G.S. Fishman, B.D. Huang (1983). *Antithetic Variates Revisited*, Communications of the ACM, 26, November, pp. 964-971.

[7] J.E. Gentle (2003). *Random Number Generation and Monte Carlo Methods (2nd ed.)*, Springer-Verlag.

[8] P. Glasserman (2004). *Monte Carlo methods in financial engineering*, Springer.

[9] J.M. Hammersley and K.W. Morton (1956). *A New Monte Carlo Technique : Antithetic Variates*, Proceedings of the Cambridge Philosophical Society, 52, pp. 449-475.

[10] C. Herring, J.I. Palmore (1989). *Random Number Generators Are Chaotic*, Communications of the ACM, 38 :1, pp. 121-127.

[11] P. Jäckel (2002). *Monte Carlo methods in finance*, John Wiley & Sons Ltd.

[12] C. Klimasauskas (2003). *Not Knowing Your Random Number Generator Could Be Costly : Random Generators - Why Are They Important*, Information Article, Advanced Technology For Developers Group, http :// www.klimasauskas.com/ pub_rng.php.

[13] D.E. Knuth (1998). *The Art of Computer Programming, Volume 2 : Seminumerical Algorithms (third edition)*, Addison-Wesley.

[14] M. Langlois (1999). *Cryptographie quantique - solution au problème de distribution de clefs secrètes*, Working Paper, Université d'Ottawa.

[15] D. Lamberton, B. Lapeyre (1997). *Introduction au calcul stochastique appliqué à la finance*, Ellipse.

[16] A. Lachaud, G. Leclanche (2003). *Génération de nombres aléatoires par numérisation d'impulsions radiatives*, Rapport de fin d'études, Maîtrise d'Electronique, Université de Limoges.

[17] P. L'Ecuyer (1998). *Random Number Generators and Empirical Tests*, Lecture Notes in Statistics 127, Springer-Verlag, pp. 124-138.

[18] P. L'Ecuyer (2001). *Software for uniform random number generation : distinguishing the good and the bad*, Proceedings of the 2001 Winter Simulation Conference, IEEE Press, December, pp. 95-105.

[19] P. L'Ecuyer (2004). *Random Number Generation*, chapter 2 of the Handbook of Computational Statistics, J. E. Gentle, W. Haerdle, and Y. Mori, eds., Springer-Verlag, pp. 35-70.

[20] P. L'Ecuyer, F. Panneton (2000). *A New Class of Linear Feedback Shift Register Generators*, Proceedings of the 2000 Winter Simulation Conference, December, pp. 690-696.

[21] P. L'Ecuyer, R. Simard (2005). *TestU01 - A software Library in ANSI C for Empirical Testing of Random Number Generators*, User's guide (compact version), Université de Montréal - DIRO, http :// www.iro.umontreal.ca/ ~simardr/.

[22] G. Marsaglia (1996). *DIEHARD, a battery of tests of randomness*, voir http ://www.stat.fsu.edu/pub/DIEHARD/.

[23] G. Marsaglia (2003). *Seeds for random number generators*, Communications of the ACM, 46 :5, pp. 90-93.

[24] M. Matsumoto, Y. Kurita (1992). *Twisted GFSR generators*, ACM Trans. on Modeling and Computer Simulation, 2, pp. 179-194.

[25] M. Matsumoto, Y. Kurita (1994). *Twisted GFSR generators II*, ACM Trans. on Modeling and Computer Simulation, 4, pp. 254-266.

[26] M. Matsumoto, T. Nishimura (1998). *Mersenne Twister : A 623-dimensionally equidistributed uniform pseudorandom number generator*, ACM Trans. on Modeling and Computer Simulation, Vol. 8, No. 1, January, pp. 3-30

[27] H.R. Neave (1973). *On using the Box-Muller tranformation with multiplicative congruential pseudo-random number generators*, Applied Statistics, 22, pp. 92-97.

[28] H. Niederrreiter (1992). *Random Number Generation and Quasi-Monte Carlo Methods*, SIAM-CBMS Lecture Notes 63.

[29] M.K. Ong (2007). *The Basel Handbook Second Edition, A Guide for Financial Practinioners*, RiskBooks & KPMG.

[30] F. Panneton (2004). *Construction d'ensembles de points basée sur des récurrences linéaires dans un corps fini de caractéristique 2 pour la simulation Monte Carlo et l'intégration quasi-Monte Carlo*, Thèse de Doctorat, Université de Montréal.

[31] S. K. Park, K.W. Miller (1988). *Random Number Generators : Good Ones Are Hard To Find*, Communications of the ACM, Vol 31, 10, pp. 1192-1201.

[32] P. Pauletto (2001). *Parallel Monte Carlo Methods for Derivative Security Pricing*, in Numerical Analysis and Its Applications, L. Vulkov, J. Wasnievski and P. Yalamov, eds., Lecture Notes in Computer Science, Vol. 1988, Springer-Verlag, pp. 650-657.

[33] F. Planchet, P. Therond, J. Jacquemin (2005). *Modèles financiers en assurance. Analyse de risque dynamiques*, Economica.

[34] M. Saito (2007). *An Application of Finite Field : Design and Implementation of 128-bit Instruction-Based Fast Pseudorandom Number Generator*, Research Paper, Dept. of Math., Graduate School of Science, Hiroshima University, http ://www.math.sci.hiroshima-u.ac.jp/~m-mat/JSPS-CoreToCore/index.html.

[35] L. Schrage (1979). *A More Portable Fortran Random Number Generator*, ACM Transactions on Mathematical Software, Vol 5, No 2, pp. 132-138.

[36] S. Tezuka (1991). *Neave Effect Also Occurs With Tausworthe Sequences*, Proceedings of the 1991 Winter Simulation Conference, pp. 1030-1034.

[37] G. West (2005). *Better approximations to cumulative normal functions*, Wilmott Magazine, pp.70-76.