

MÉTHODES DE MONTE-CARLO  
*Master 1 PSA: 2019–2020*

Y. Boumzaid

U de Béjaïa



# Table des matières

<b>1</b>	<b>Simulation</b>	<b>5</b>
1.1	Nombres aléatoires et pseudo-aléatoires . . . . .	5
1.2	Générateur de nombres uniformes . . . . .	6
1.2.1	Méthode des congruances . . . . .	6
1.2.2	Quelques générateurs informatiques . . . . .	7
1.3	Générateur de nombres non uniformes . . . . .	7
1.3.1	Méthodes d'inversion . . . . .	8
1.3.2	Méthode d'acceptation-rejet . . . . .	11
1.3.3	Méthode de Box-Müller . . . . .	15
<b>2</b>	<b>Intégration Monte-Carlo</b>	<b>19</b>
2.1	Intégration Monte-Carlo . . . . .	19
2.1.1	Loi faible des grands nombres . . . . .	19
2.1.2	Loi forte des grands nombres . . . . .	20
2.1.3	Exemples d'application . . . . .	20
2.2	Erreur de l'estimation M.C . . . . .	23
2.3	Réduction de variance . . . . .	27
2.3.1	Méthode d'échantillonnage préférentiel . . . . .	27
2.3.2	Méthode de Variable de Contrôle . . . . .	30
2.3.3	Méthode variable antithétique . . . . .	33
<b>3</b>	<b>Méthode de Monte-Carlo par chaîne de Markov</b>	<b>37</b>
3.1	Généralités sur les chaînes de Markov . . . . .	37
3.1.1	Chaînes de Markov . . . . .	37
3.1.2	Probabilité de transition . . . . .	37
3.1.3	Matrice de transition (noyau, opérateur) . . . . .	38
3.1.4	Propriétés fondamentales . . . . .	38
3.1.5	Classification des états d'une chaîne de Markov . . . . .	38
3.1.6	Distribution stationnaire . . . . .	39
3.1.7	Réversibilité(Symétrie) . . . . .	39
3.1.8	Lois des grands nombres (convergence et vitesse de convergence) . . . . .	40
3.2	Méthode de Monte-Carlo par chaîne de Markov . . . . .	40
3.2.1	Algorithme de Metropolis-Hastings . . . . .	41
3.2.2	Typologies des algorithmes de Hastings-Metropolis . . . . .	44
3.2.3	Echantillonneur de Gibbs . . . . .	47
3.3	La qualité de la simulation . . . . .	48
3.3.1	Choix de la valeur de l'état initial . . . . .	49
3.3.2	Choix de la loi de proposition . . . . .	49



# Introduction

## Avant propos

Simuler un système avec des paramètres ou des conditions initiales probabilistes demande la capacité de générer des nombres selon une distribution de probabilité. Les techniques de simulation Monte Carlo sont utilisées pour simuler des systèmes déterministes avec des paramètres ou des entrées stochastiques. Le nom de ces derniers a été proposé par les scientifiques du projet Manhattan lors de la deuxième guerre mondiale et fait allusion aux jeux de hasard pratiqués aux casino de la principauté de Monaco. Les méthodes de Monte Carlo notées "MC" sont aujourd'hui utilisées pour simuler des phénomènes physiques complexes dans plusieurs domaines scientifiques et appliqués : radioactivité, physique des hautes énergies, réseaux, économétrie, logistique. La technique de simulation Monte Carlo s'appuie sur l'échantillonnage des distributions des quantités incertaines.

## L'objectif de l'enseignement de cette matière

L'objectif de l'enseignement de cette matière est d'initier les étudiants à la simulation et particulièrement à la simulation de Monte Carlo en utilisant différentes méthodes d'échantillonnages et de maîtriser les approximations par simulation.

## Le contenu de la matière

Le contenu de cette matière s'étend sur 3 chapitres. Dans le premier chapitre, nous donnons une brève introduction à la simulation ainsi qu'à la génération des suites de nombres aléatoires suivant différentes lois de probabilités. Dans le deuxième chapitre, nous abordons les méthodes de Monte Carlo : Méthodes classiques de simulation de Monte Carlo, principe de la méthode, méthodes de réduction de variance, Autres méthodes d'échantillonnage utilisées en simulation de Monte Carlo, application aux calculs d'intégral. Dans le troisième chapitre, nous abordons les méthodes de simulation de Monte Carlo par chaîne de Markov, le principe de la méthode, algorithme de Gibbs, algorithme de Hastings-Metropolis.



# Chapitre 1

## Simulation

Trouver une solution analytique d'un modèle probabiliste est souvent impossible. Dans ce cas, la seule manière d'étudier le système est donc la simulation. Simuler un système avec des paramètres ou des conditions initiales probabilistes demande la capacité de générer des nombres selon une distribution de probabilité. Dans ce chapitre, nous donnons quelques techniques de simulation permettant de générer un échantillon de variables aléatoires distribuées selon une densité donnée.

### 1.1 Nombres aléatoires et pseudo-aléatoires

Pour la simulation d'un modèle non-déterministe, il est nécessaire de disposer d'une source de nombres susceptibles de jouer le rôle des variables aléatoires qui interviennent dans la définition du modèle.

**Définition 1.** *Un générateur de nombres aléatoires (G.N.A) est un mécanisme capable de produire une séquence de nombres qui ont l'air d'être choisis complètement au hasard.*

**Exemple 1.** — Suite d'entiers de 1 et 100 : 31 45 02 72.....  
— Suite de nombres réels entre 0 et 1.

De vrais nombres aléatoires peuvent être produits par n'importe quel processus naturel considéré comme aléatoire. Une méthode pour générer des nombres aléatoires consiste à tirer d'une urne des billets numérotés de 0 à 9, un billet à la fois, en remettant dans l'urne chaque billet tiré. Le développement de la technologie informatique a toutefois supplanté ce type de générateurs au profit des générateurs de nombres pseudo-aléatoires.

**Définition 2.** *Un générateur algorithmiques ou de nombres pseudo-aléatoires (GNPA) est un algorithme qui génère une séquence de nombres présentant certaines propriétés du hasard. De sorte que, les nombres générés par cet algorithme sont supposés être suffisamment indépendants les uns des autres, et qu'il est potentiellement difficile de repérer des groupes de nombres qui suivent une certaine règle (comportements de groupe).*

Cependant, les sorties d'un tel générateur ne sont pas entièrement aléatoires ; elles s'approchent seulement des propriétés idéales des sources complètement aléatoires, comme le faisait remarquer John von Neumann : « **Quiconque considère des méthodes arithmétiques pour produire des nombres aléatoires est, bien sûr, en train de commettre un péché** ».

La raison pour laquelle on se contente de nombres pseudo-aléatoires est que :

- Il est difficile d'obtenir un grand échantillon de « vrais » nombres aléatoires.
- Les algorithmes générateurs sont particulièrement adaptés à une implémentation informatique, donc plus facilement et plus efficacement utilisables.

Dans la suite de ce document, un générateur de nombres uniformes sera un algorithme qui permet de produire une suite de nombres aléatoires issu d'une population distribuée selon la loi  $\mathcal{U}([0, 1])$  dont les éléments sont indépendants les uns des autres. Autrement dit, un nombre aléatoire généré par cet algorithme peut-être vu comme la réalisation d'une variable aléatoire distribuée uniformément dans l'intervalle  $[0, 1]$ .

## 1.2 Générateur de nombres uniformes

Pour aborder la simulation de la meilleur façon possible, on commence tout d'abord par introduire les générateurs de nombres uniformes (simulation de la loi uniforme sur l'intervalle  $[0, 1]$ , sur laquelle la simulation de toutes les autres lois est basée.

La plupart des algorithmes générateurs de nombres pseudo-aléatoires ont pour but de produire des suites uniformément distribuées. Une classe très répandue de générateurs est basée sur la méthode des congruences.

### 1.2.1 Méthode des congruances

La méthode des congruences repose sur un algorithme simple pour la génération de nombres pseudo-aléatoires. En effet, cette dernière est basée sur la relation de récurrence suivante :

$$X_i = (aX_{i-1} + b) \bmod m \quad i = 1, 2, \dots, n-1, \quad (1.1)$$

où  $a$ ,  $b$ ,  $m$  et  $x_0$  sont des entiers positifs donnés. On dit que  $a$  est le multiplicateur,  $b$  est l'incrément,  $m$  le modulus,  $n$  le nombre des échantillons à générer et  $x_0$  la valeur initiale appelée aussi graine.

Le nombre pseudo-aléatoire,  $U_i$ ,  $i = 0..n-1$ , compris entre 0 et 1, est obtenu par la relation :

$$U_i = X_i / m.$$

La longueur d'une suite quelconque générée par une relation récursive du type  $(aX_{i-1} + b) \bmod m$  ne peut pas dépasser  $m$  puisqu'il y a au plus  $m$  nombres différents modulo  $m$ . De manière générale, il faut choisir une grande valeur de  $m$ . Pour obtenir une suite de longueur maximale, c'est-à-dire de période  $m$ . Pour cela, il faut que les conditions du théorème suivant soient vérifiées :

**Théorème 1.** (Hull et Dobell (1962))

Soit  $k \in \{0, 1, \dots, m-1\}$ . Soient  $a$ ,  $b$ ,  $m$ , tels que :

1.  $b$  et  $m$  sont premiers entre eux ;
2.  $(a-1)$  est un multiple de chaque nombre premier qui divise  $m$  ;
3. si  $m$  est un multiple de 4 alors  $(a-1)$  l'est aussi.

Alors la suite définie par :

$$\begin{cases} X_0 = k \\ X_i = (aX_{i-1} + b) \bmod m \end{cases}$$

a un cycle de longueur  $m$

**Remarque 1.** Le nombre pseudo-aléatoire compris entre 0 et 1 est obtenu par la relation

$$U_i = \frac{X_i}{m} \quad (1.2)$$



### 1.2.2 Quelques générateurs informatiques

La plupart des générateurs de nombres aléatoires utilisés par les logiciels et les langages de programmation informatique sont construits à l'aide de la méthode de congruence présentée dans la section précédente. Dans cette sous-section, nous donnons les paramètres de quelques générateurs utilisés en informatique (voir tableau 1.1).

Langage/Logiciel	Générateur	$a$	$b$	$M$
IBM	RANDU	$2^{16} + 3$	0	$2^{31}$
Scilab	RAND	843314861	453816693	$2^{31}$
Turbo Pascal	RANDOM	129	907633385	$2^{32}$
Langage c	RAND	103515245	12345	$2^{31}$
Maple	RAND	427419669081	0	$10^{12} - 11$

TABLE 1.1 – Générateurs informatique

**Exemple 2.** L'algorithme ci-dessus, génère une suite de nombres aléatoires avec le générateur de Scilab.

---

#### Algorithme

---

##### Debut

$lire(n)$ ;

$a = 843314861$ ;

$b = 453816693$ ;

$m = 2^{31}$ ;

$X(1) = 0$ ;    %la graine

**pour**  $i = 1 : n$  **faire**

$X(i + 1) = (aX(i) + b) \bmod m$ ;

$U(i) = \frac{X(i + 1)}{m}$ ;

**fin pour**;

$afficher(U(i))$ ;

**Fin.**

---

La figure 1.1 montre la simulation de 1000 tirages de variable aléatoire de loi  $\mathcal{U}([0, 1])$  obtenues par le générateur Scilab.

Rappelons, qu'il existe beaucoup de tests permettant de s'assurer de la qualité des nombres aléatoires produits par un générateur algorithmique. Ces derniers permettent de vérifier la stochasticité et l'uniformité des séquences  $(u_1, \dots, u_n)$  générées. Parmi ces tests, on peut citer, le test de khi-deux, le test de Kolmogorov-Smirnov ou encore des tests basés sur l'étude de corrélation entre les termes des séries temporelles  $u_i$  et  $(u_{i-1}, \dots, u_{i-n})$ .

## 1.3 Générateur de nombres non uniformes

Dans la section précédente nous avons vu comment simuler ou générer des suites de nombres aléatoires distribués de façon uniforme sur l'intervalle  $[0, 1]$ . Ceux-ci sont à la base de toute simulation. Dans cette section, nous présentons quelques méthodes de génération de nombres non uniformes. Dans ce cas on parle de la simulation d'une variable aléatoire qui suit une loi de probabilité quelconque. Rappelons à juste titre, qu'il existe plusieurs méthodes pour la simulation ou la génération de nombres aléatoires non-uniforme. Dans cette section, nous nous contentons par la présentation des trois méthodes principales à savoir :

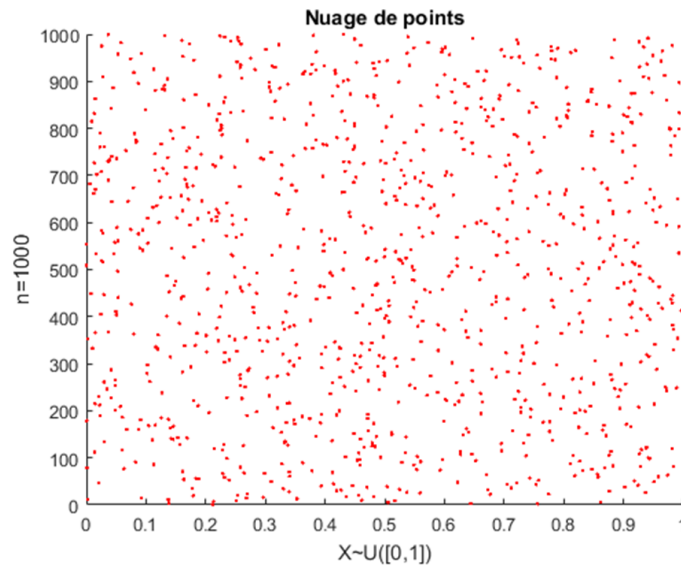


FIGURE 1.1 – Génération des nombres aléatoires

- La méthode d'inversion .
- La méthode d'acceptation rejet.
- La méthode de Box-Müller .

### 1.3.1 Méthodes d'inversion

Une des méthodes de simulation des variables aléatoires est la méthode d'inversion. Comme son nom l'indique cette méthode est fondée sur l'inversion de la fonction de répartition. Cette méthode est basée sur les deux théorème suivants :

**Théorème 2.** Soit  $X$  une v.a de fonction de répartition  $F$ , continue et strictement croissante, on a :

Si  $U \sim \mathcal{U}([0, 1])$  Alors  $F^{-1}(U)$  à même loi que  $X$ .

Autrement dit, il suffit de simuler  $U$  suivant  $\mathcal{U}([0, 1])$  puis appliquer la transformation  $X = F^{-1}(U)$  pour simuler suivant la loi de  $X$ .

**Preuve 1.** Supposons que  $X$  continue,  $F$  croissante et  $F^{-1}$  existe, on pose :  $X = F^{-1}(U)$

$$P(X \leq x) = P(F^{-1}(U) \leq x) = P(U \leq F(x)) = F(x).$$

L'algorithme de cette méthode est donné comme suit :

---

#### Algorithme de simulation par inversion

---

**Debut**

lire( $n$ ) ; %nombre d'échantillon

$i = 1$  ;

**pour**  $i = 1 : n$  **faire**

    générer  $U(i) \sim \mathcal{U}([0, 1])$  ;

$X(i) = F_X^{-1}(U(i))$  ;

**fin pour** ;

afficher ( $X(i)$ ) ;

**Fin.**

---

Si  $F$  n'est pas continue (continue par partie ou  $X$  suit une loi discrète sur un ensemble fini). La fonction  $F^{-1}$  dans ce cas est appelée fonction "pseudo-inverse". Dans ces deux cas, la méthode d'inversion peut être généralisée à des fonctions de répartition non bijective grâce au théorème suivant :

**Théorème 3. (Inverse généralisée)**

Soit  $X$  une v.a de fonction de répartition  $F$ . posons pour  $0 \leq u \leq 1$

$$F^{-1} = \inf\{x, F(x) \geq u\} \quad (1.3)$$

Alors, si  $U \sim \mathcal{U}([0, 1])$  Alors  $F^{-1}$  à même loi que  $X$ .

Autrement dit, considérons une variable aléatoire  $X$  discrète à valeurs dans l'ensemble fini  $\{x_1, x_2, \dots, x_n\}$  avec probabilités  $(p_1, \dots, p_n)$ . Il est facile de vérifier que pour tout  $u \in ]0, 1[$ ,

$$F^{-1}(u) = \begin{cases} x_1 & \text{si } 0 \leq u < p_1 \\ x_2 & \text{si } p_1 \leq u < p_1 + p_2 \\ \vdots & \vdots \\ x_n & \text{si } \sum_{k=1}^{n-1} p_k \leq u < 1 \end{cases}$$

c'est-à-dire

$$X = \sum_{k=1}^n k \cdot \delta_{p_{k-1} \leq u < p_k},$$

avec

$$\begin{cases} P_0 = 0 \\ P_k = \sum_{i \leq k} p_i \end{cases}$$

**Exemple 3. (Simulation d'une loi discrète à support fini)**

Soit la loi de probabilité suivante :

$k$	1	2	3	4	5	total
$n_k$	10	20	30	30	10	100
$p_k$	0.1	0.2	0.3	0.3	0.1	1

La fonction de répartition de cette loi est donnée dans le tableau suivant :

$x$	$x < 1$	$1 \leq x < 2$	$2 \leq x < 3$	$3 \leq x < 4$	$4 \leq x < 5$	$x \geq 5$
$F(x)$	0	0.1	0.3	0.6	0.9	1

Pour simuler cette loi on tire une suite de nombres aléatoires  $u \sim \mathcal{U}([0, 1])$ . D'après le théorème 2, la fonction pseudo-inverse est donnée par :

$$F^{-1}(u) = \begin{cases} 1 & \text{si } 0 \leq u < 0.1 \\ 2 & \text{si } 0.1 \leq u < 0.3 \\ 3 & \text{si } 0.3 \leq u < 0.6 \\ 4 & \text{si } 0.6 \leq u < 0.9 \\ 5 & \text{si } 0.9 \leq u < 1 \end{cases} \quad (1.4)$$

Soit

$$U = (0.9, 0.2, 0.33, 0.45, 0.1)$$

Donc l'échantillon de  $X$  a généré est donné à partir de l'équation (1.4) par :

$$X = (5, 2, 3, 3, 2)$$

**Exemple 4.** Soit  $X \sim \varepsilon(\lambda)$  une variable aléatoire de loi exponentielle de paramètre  $\lambda > 0$ . La densité de la loi exponentielle s'écrit

$$f(x) = \lambda e^{-\lambda x}, \quad x > 0.$$

Et sa fonction de répartition est donnée par :

$$F(x) = 1 - e^{-\lambda x}, \quad x > 0.$$

Posons pour tout  $0 \leq u \leq 1$ ,  $u = F(x)$  alors

$$u = 1 - e^{-\lambda x} \Leftrightarrow x = -\frac{\ln(1-u)}{\lambda} \quad (1.5)$$

Si  $U \sim \mathcal{U}([0, 1])$ , nous avons, d'après le résultat ci-dessus :

$$-\frac{\ln(1-U)}{\lambda} \sim \varepsilon(\lambda) \quad (1.6)$$

Remarquons que  $(1-U)$  a même loi que  $U$  et donc

$$-\frac{\ln(U)}{\lambda} \sim \varepsilon(\lambda) \quad (1.7)$$

L'algorithme suivant permet de simuler un échantillon de variable aléatoire qui suit une loi exponentielle :

---

**Algorithme simulation de la loi  $\varepsilon(\lambda)$**

---

**Debut**

lire( $n$ ) ; %nombre d'échantillon

lire( $\lambda$ ) ;

$i = 1$  ;

**pour**  $i = 1 : n$  **faire**

    générer  $U(i) \sim \mathcal{U}([0, 1])$  ;

$X(i) = -\frac{1}{\lambda} \log(U(i))$  ;

**fin pour** ;

afficher ( $X(i)$ ) ;

**Fin.**

---

La figure 1.2, représente l'histogramme et la distribution d'un échantillon de taille  $n = 100$  généré à partir de la loi exponentielle de paramètre  $\lambda = \frac{1}{2}$ .

**Exemple 5.** Simulation de la loi Laplace de paramètre  $(0, \frac{1}{\lambda})$ ,  $\lambda > 0$ .

La densité de la loi Laplace de paramètre  $(0, \frac{1}{\lambda})$  est donnée par :

$$f(x) = \frac{\lambda}{2} e^{-\lambda|x|}, \quad x \in \mathbb{R} \quad (1.8)$$

Et sa fonction de répartition est donnée par :

$$F(x) = \begin{cases} \frac{1}{2} e^{\lambda x}, & \text{si } x < 0; \\ \frac{1}{2} (1 - e^{-\lambda x}), & \text{si } x \geq 0. \end{cases} \quad (1.9)$$

Posons pour tout  $0 \leq u \leq 1$ ,  $u = F(x)$ ,

on trouve :

$$F^{-1}(u) = \begin{cases} \frac{1}{\lambda} \ln(2u), & \text{si } 0 \leq u < \frac{1}{2}; \\ -\frac{1}{\lambda} \ln(2(1-u)), & \text{si } \frac{1}{2} \leq u \leq 1. \end{cases} \quad (1.10)$$

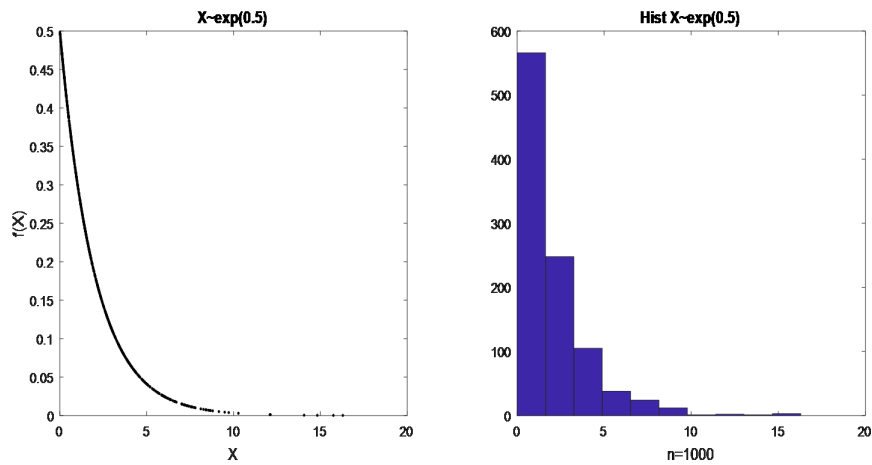


FIGURE 1.2 – Simulation de la loi exponentielle

On peut mettre en place l'algorithme suivant :

---

**Algorithme simulation de la loi Laplace**

---

**Debut**

lire( $n$ ) ; %nombre d'échantillon

lire( $\lambda$ ) ;

$i = 1$  ;

**pour**  $i = 1 : n$  **faire**

    générer  $U(i) \sim \mathcal{U}([0, 1])$  ;

**si**  $u \leq \frac{1}{2}$  ;

$X(i) = \frac{1}{\lambda} \log(2U(i))$  ;

**sinon**

$X(i) = -\frac{1}{\lambda} \log(2(1 - U(i)))$  ;

**fin si** ;

**fin pour** ;

afficher ( $X$ ) ;

**Fin.**

---

La figure 1.3, représente l'histogramme et la distribution d'un échantillon de taille  $n = 100$  génère à partir de la loi de Laplace de paramètre  $(0, \lambda = 0.5)$  :

**Limite de la méthode d'inversion**

Nous avons vu précédemment que la méthode d'inversion requiert la connaissance et le calcul rapide de la fonction inverse de la fonction de répartition. Ceci n'est pas toujours envisageable. En effet, il suffit de penser à la loi normale centrée réduite de densité,

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}, \quad x \in \mathbb{R},$$

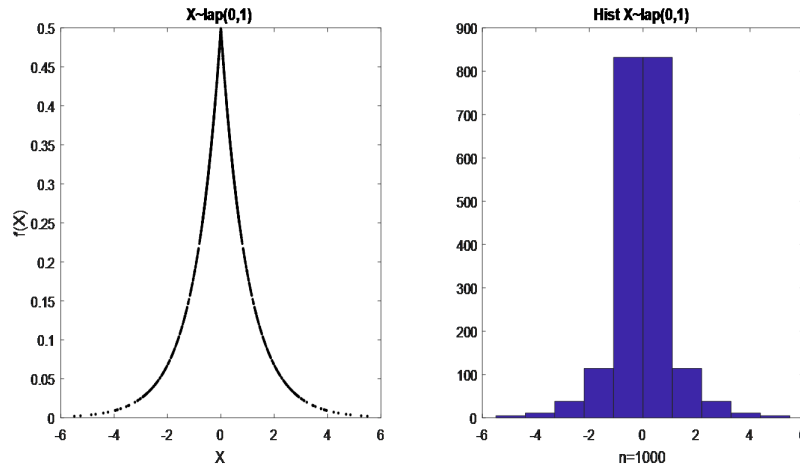


FIGURE 1.3 – simulation de la loi de Laplace

dont la fonction de répartition n'admet pas d'expression analytique simple et encore moins pour sa fonction inverse.

Dans ce cas la méthode de rejet détaillée dans la sous-section suivante peut représenter une alternative judicieuse.

### 1.3.2 Méthode d'acceptation-rejet

Il y en a plusieurs versions de la méthode de rejet. Nous nous contentons par présenter la méthode la plus générale. Supposons que l'on veuille simuler une variable aléatoire  $X$  de densité  $f$  (dans  $\mathbb{R}^d$ ) et supposons aussi qu'il existe une loi de densité  $g$  facile à simuler telle que :

1. On sait simuler  $Y$  de densité  $g$ .
2. Il existe une constante  $m$  telle que, pour presque tout  $x$ ,  $f(x) \leq mg(x)$ .

Considérons alors deux suites indépendantes de variables aléatoires :

- (a)  $(Y_n)_{n \geq 1}$  iid de densité  $g$ .
- (b)  $(U_n)_{n \geq 1}$  iid de loi uniforme.

Autrement dit,  $Y$  correspond à une proposition et  $U$  à un tirage pile ou face pour décider si on accepte ou non cette proposition. Nous noterons par  $r$  la fonction rapport d'acceptation-rejet pour la pile ou face, à savoir pour tout  $y \in \mathbb{R}^d$  :

$$r(y) = \begin{cases} \frac{f(y)}{mg(y)}, & \text{si } g(y) > 0; \\ 0, & \text{sinon.} \end{cases} \quad (1.11)$$

Par ailleurs,  $f$  et  $g$  étant des densités, la constante  $m$  intervenant dans la majoration est supérieure à 1.

La proposition suivante montre comment simuler suivant la densité  $f$  voulue.

**Proposition 1.** Soit  $N = \inf\{n \geq 1, U_n \leq r(Y_n)\}$  le premier instant où le tirage est accepté et  $X = Y_n$  la valeur de la proposition à cet instant. Alors  $X$  a pour densité  $f$ . Par ailleurs,  $N$  suit une loi géométrique de paramètre  $\frac{1}{m}$ .

**Remarque 2.** La variable  $N$  étant géométrique de paramètre  $\frac{1}{m}$ , elle a pour moyenne  $m$ . Concrètement, il faut donc faire en moyenne  $m$  essais pour obtenir une seule simulation selon la loi cible  $f$ . Dès lors, il s'agira de choisir le couple  $(g, m)$  de sorte que  $m$  soit aussi proche de 1 que possible. En d'autres termes. On a tout intérêt à choisir une densité  $g$  qui ressemble le plus possible à  $f$ .

L'algorithme de la méthode de rejet est donné comme suit :

---

**Algorithme d'acceptation rejet**

---

**Debut**

lire( $n$ );

$i = 1$ ;

$j = 1$ ;

**Tantque**( $i \leq n$ )**faire**

    générer  $U(i) \sim \mathcal{U}([0, 1])$ ;

    générer  $Y(i)$  selon  $g$ ;

$r(i) = \frac{f(Y(i))}{mg(Y(i))}$ ;

**si**  $U(i) \leq r(i)$

$X(j) = Y(i)$ ;

$j = j + 1$ ;

**fin si**;

$i = i + 1$ ;

**fin Tantque**;

afficher ( $X$ );

**Fin.**

---

Afin de mieux comprendre la méthode d'acceptation-rejet on donne un exemple d'application complet.

Dans cet exemple, nous montrons comment déterminer la constante  $m$  et comment minimiser la probabilité de rejet (le nombre d'itération).

**Exemple 6.** Simulation d'une loi normale centrée réduite à partir de la densité de Laplace vue dans l'exemple 5.

Soit  $f$  la densité de la loi normale  $\mathcal{N}(0, 1)$ ,

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}.$$

Soit  $g$  la densité de la loi de Laplace,

$$g(x) = \frac{\lambda}{2} e^{-\lambda|x|}.$$

Calculons le rapport  $f(x)/g(x)$ ;

nous avons :

$$\forall x \in \mathbb{R}, \frac{f(x)}{g(x)} = \frac{2}{\sqrt{2\pi}} \frac{1}{\lambda} e^{-\frac{x^2}{2} + \lambda|x|}$$

Par symétrie, nous pouvons écrire :

$$\forall x \geq 0, \frac{f(x)}{g(x)} = \frac{2}{\sqrt{2\pi}} \frac{1}{\lambda} e^{-\frac{x^2}{2} + \lambda x}.$$

Posons  $h(x) = e^{-\frac{x^2}{2} + \lambda x}$  et étudions cette fonction. Pour tout  $x \geq 0$ , nous avons

$$h'(x) = (-x + \lambda)h(x)$$

d'où le tableau de variation de la table (1.2).

$x$	0	$\lambda$	$+\infty$
$h'(x)$	+	0	-
$h(x)$	$\nearrow$	$h(\lambda)$	$\searrow$

TABLE 1.2 – Tableau de variation de  $h$ .

Donc, pour tout  $x \in \mathbb{R}^+$ ,

$$f(x)/g(x) \leq \frac{2}{\sqrt{2\pi}} \frac{1}{\lambda} h(\lambda).$$

Posons :

$$m(\lambda) = \frac{2}{\sqrt{2\pi}} \frac{1}{\lambda} e^{\frac{\lambda^2}{2}}.$$

Il reste à minimiser la fonction  $m(\lambda)$ .

Nous avons  $\forall \lambda \geq 0$ ,

$$m'(\lambda) = \frac{2}{\sqrt{2\pi}} \left(1 - \frac{1}{\lambda^2}\right) e^{\frac{\lambda^2}{2}}.$$

D'où le tableau de variation de  $m(\lambda)$  donné dans la table (1.3). D'après ce tableau, la valeur de  $\lambda$  qui permet de minimiser la probabilité de rejet est :  $\lambda = 1$ . Donc pour  $\lambda = 1$ , on trouve  $m = \frac{2}{\sqrt{2\pi}}$

$\lambda$	0	1	$+\infty$
$m'(\lambda)$	-	0	+
$m(\lambda)$	$\searrow$	$m(1)$	$\nearrow$

TABLE 1.3 – Tableau de variation de  $m$ .

et  $g(x) = e^{-|x|}$ .

D'après les résultats précédentes, on peut donc simuler  $f$  avec un algorithme de rejet (puisque'il est facile de simuler suivant la loi de densité  $g$ ).

#### 1. Simulation de la loi de densité $g$ :

Par l'application de la méthode d'inversion sur la fonction de répartition de  $g$ , on trouve :

$$\forall u \in [0, 1], |y| = -\log(u)$$

#### 2. Le rapport d'acceptation-rejet $r(y)$

Le rapport d'acceptation-rejet est donné par :

$$r(y) = e^{-\frac{(|y|-1)^2}{2}}.$$

La simulation d'une v.a qui suit la loi normale par la méthode de rejet est donnée par l'algorithme suivant :



---

**Algorithme**

---

**Debut** $\text{lire}(n);$  $i = 1;$  $j = 1;$ **Tantque**( $i \leq n$ )**faire**    générer  $U(i) \sim \mathcal{U}([0, 1]);$      $Y(i) = \log(2U(i));$      $r(i) = e^{-\frac{1}{2}(Y(i)-1)^2};$     **si**( $U(i) \leq r(i)$ )         $X(j) = Y(i);$          $j = j + 1;$     **fin si**;     $i = i + 1;$ **fin Tantque**;afficher ( $[X, -X]$ );**Fin.**

---

La figure 1.4 représente la simulation de la loi normale à partir de la loi de Laplace.

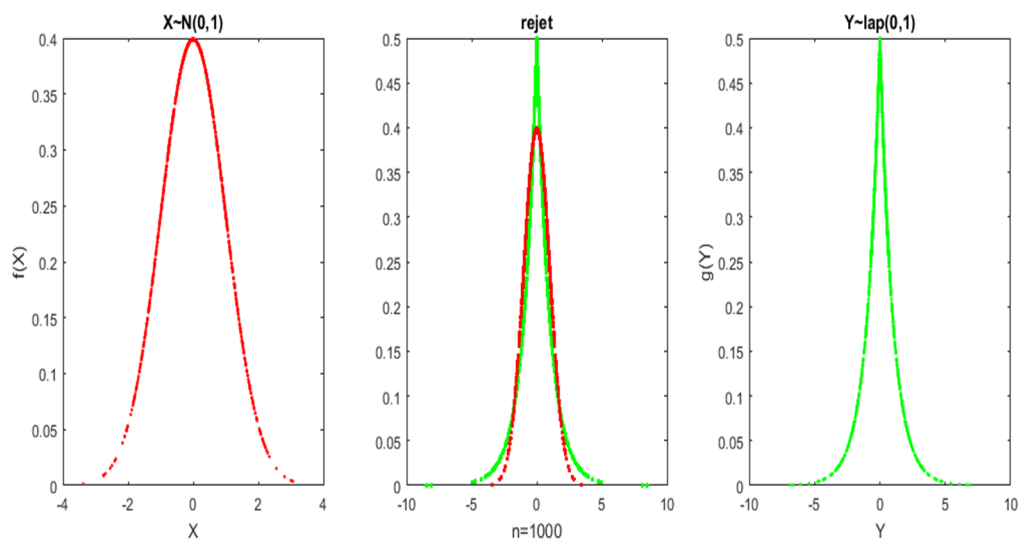


FIGURE 1.4 – Simulation d’une loi normale à partir de la loi de Laplace. A gauche, la simulation de la loi normale, au milieu la simulation de la loi normale par la méthode de rejet et à droite la simulation de la loi de Laplace

Après avoir présenté dans les deux sous-sections précédentes la méthode d’inversion et la méthode de rejet, nous présentons dans ce qui suit la méthode de Box-Müller.

### 1.3.3 Méthode de Box-Müller

Pour la simulation d'une variable aléatoire qui suit une loi normale, on peut employer plusieurs méthodes. Ces méthodes sont basées sur :

1. L'application du Théorème Central Limite.
2. L'algorithme du rejet aux densités à support non compact (exemple précédent).
3. La méthode de Box-Müller.

Dans cette section, nous nous contentons par la présentation de la méthode de Box-Müller. Cette dernière permet de simuler un couple des variables aléatoires normales, centrées, réduites et indépendantes.

Supposons que l'on veuille simuler  $X \sim \mathcal{N}(0, 1)$  et  $Y \sim \mathcal{N}(0, 1)$  indépendantes. On connaît la densité jointe de  $X$  et  $Y$  :

$$f(x, y) = \frac{1}{2\pi} e^{-\frac{x^2+y^2}{2}}$$

**Proposition 2.** Si  $U, V$  sont de loi  $\mathcal{U}([0, 1])$  et indépendantes. On pose :

$$\begin{cases} X = \sqrt{-2 \log(U)} \cos(2\pi V) \\ Y = \sqrt{-2 \log(U)} \sin(2\pi V) \end{cases}$$

Alors  $X$  et  $Y$  sont indépendantes et de même loi que  $\mathcal{N}(0, 1)$

**Preuve 2.** Soient :

$$X \sim \mathcal{N}(0, 1) \Rightarrow f(x)dx = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx$$

$$Y \sim \mathcal{N}(0, 1) \Rightarrow f(y)dy = \frac{1}{\sqrt{2\pi}} e^{-\frac{y^2}{2}} dy$$

Comme  $X$  et  $Y$  sont deux v.a indépendantes alors :

$$f(x, y)dxdy = \frac{1}{2\pi} e^{-\frac{x^2+y^2}{2}} dxdy$$

On considère le changement de variables en coordonnée polaire :

$$\begin{cases} X = r \sin(\theta) \\ Y = r \cos(\theta) \end{cases}$$

on a donc :

$$f(r, \theta)drd\theta = \frac{1}{2\pi} e^{-\frac{r^2}{2}} rdrd\theta$$

la fonction de répartition :

$$f(r, \theta)drd\theta = \frac{1}{2\pi} e^{-\frac{r^2}{2}} rdrd\theta$$

A partir de la dernière équation nous avons  $\frac{1}{2\pi}$  représente la densité de la loi  $\Theta \sim \mathcal{U}([0, 2\pi])$ .

$re^{-\frac{r^2}{2}}$  représente la densité de la loi  $R$ . Pour déterminer la loi que suit  $R$ , nous allons étudier sa fonction de répartition  $F(r)$ .

$$F(r) = P(R \leq r) = \int_{-\infty}^r te^{-\frac{t^2}{2}} dt = [-e^{-\frac{t^2}{2}}] = 1 - e^{-\frac{r^2}{2}}.$$

Si on pose  $R' = R^2$  alors on reconnaît une loi exponentielle de paramètre  $\frac{1}{2}$

### Simulation de la loi exponentielle

Pour la simulation de la loi exponentielle de paramètre  $\frac{1}{2}$ , on utilise la méthode d'inversion (voir exemple 3 pour  $\lambda = \frac{1}{2}$ ), pour  $\lambda = \frac{1}{2}$ ,  $R' = -2 \ln(U)$  avec  $U \sim \mathcal{U}([0, 1])$ . Donc  $R = \sqrt{-2 \ln(U)}$ . D'où la relation suivante :

$$\begin{cases} X = R \sin(\Theta) \\ Y = R \cos(\Theta) \end{cases} = \begin{cases} X = \sqrt{-2 \ln(U)} \sin(2\pi V) \\ Y = \sqrt{-2 \ln(U)} \cos(2\pi V) \end{cases}$$

L'algorithme de cette méthode est donné de la façon suivante :

---

#### Algorithme de box-Müller

---

**Debut**

lire( $n$ );

**pour**  $i = 1 : n$  **faire**

    générer  $U \sim \mathcal{U}([0, 1])$ ;

    générer  $V \sim \mathcal{U}([0, 1])$ ;

$X(i) = \sqrt{-2 \log(U)} \cos(2\pi V)$ ;

$Y(i) = \sqrt{-2 \log(U)} \sin(2\pi V)$ ;

**fin pour**;

**Fin.**

---

**Remarque 3.** 1. Pour simuler une v.a  $X \sim \mathcal{N}(0, 1)$ , il suffit de

a) Tirer deux suites iid  $U, V \sim \mathcal{U}([0, 1])$ .

b)  $X = \sqrt{-2 \log(U)} \cos(2\pi V)$ .

2. Pour simuler une v.a  $Y \sim \mathcal{N}(\mu, \sigma^2)$ , il suffit de poser  $Y = \mu + \sigma X$  avec  $X \sim \mathcal{N}(0, 1)$

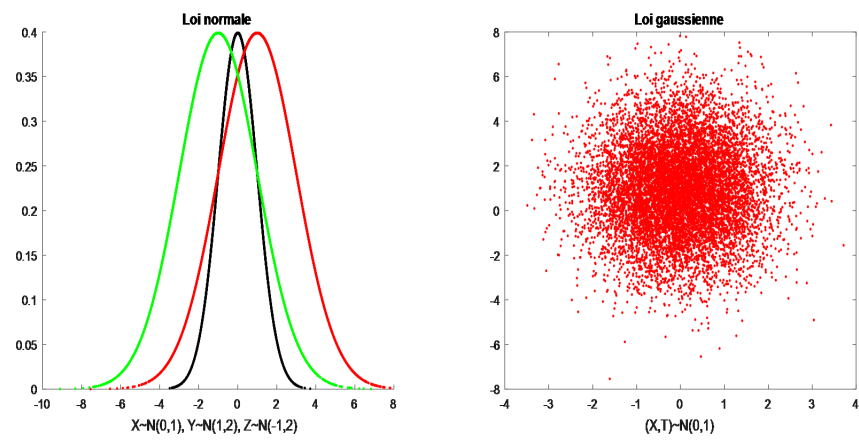


FIGURE 1.5 – A gauche la simulation de loi normale avec différentes paramètres et à droite la simulation d'un couple de v.a gaussien

## Chapitre 2

# Intégration Monte-Carlo

Après avoir présenté les différentes techniques de simulation dans le chapitre précédent, nous nous intéressons dans celui-ci à une application classique des méthodes de Monte-Carlo à savoir l'intégration Monte-Carlo. Dans un premier temps, nous donnons les techniques classiques utilisées pour l'estimation d'une quantité donnée sous forme d'une intégration Monte-Carlo, ainsi que les méthodes dites classiques. Par la suite, nous abordons la notion de l'erreur de l'estimation. Pour finir, nous donnons quelques méthodes de réduction de variance permettant de minimiser l'erreur d'estimation, le nombre d'itération et améliorer la précision. Rappelons ici que les méthodes présentées dans ce chapitre sont inutilisables lorsque la suite de variables aléatoires de la distribution dont nous voulons déterminer la valeur des moments n'est pas indépendante identiquement distribuée.

### 2.1 Intégration Monte-Carlo

Une application classique des méthodes Monte-Carlo est l'utilisation des méthodes de simulation pour estimer d'une manière approchée les quantités du type :

$$I = E[\varphi(X)] = \int_{\mathbb{R}^d} \varphi(x)f(x)dx \quad d \in \mathbb{N}^*. \quad (2.1)$$

Où  $\varphi : \mathbb{R}^d \longrightarrow \mathbb{R}$  est une fonction donnée et  $X$  un vecteur aléatoire de densité  $f$  suivant laquelle on sait simuler.

Dans ce contexte, l'estimateur Monte-Carlo de base est défini par :

$$\hat{I}_n = \frac{1}{n} \sum_{i=1}^n \varphi(X_i). \quad (2.2)$$

Où les  $X_i$  sont générées de façon i.i.d. selon  $f$ .

L'intégration Monte-Carlo est basée sur les lois des grands nombres à savoir la loi faible et la loi forte des grands nombres.

#### 2.1.1 Loi faible des grands nombres

La convergence en loi d'une suite de v.a  $X_n \xrightarrow[n \rightarrow +\infty]{loi} X$  ( $X$  une v.a de densité  $f$ ). Signifie que toute fonction  $\varphi : \mathbb{R}^d \longrightarrow \mathbb{R}$ ,  $d \in \mathbb{N}^*$  continue et bornée (mesurable  $E[\varphi(X)] < +\infty$ ).

$$E[\varphi(X_n)] \xrightarrow[n \rightarrow +\infty]{loi} E[\varphi(X)] = \int_{\mathbb{R}^d} \varphi(x)f(x)dx.$$

Autrement dit, Si nous cherchons à évaluer une intégrale de la forme,

$$I = \int_{\mathbb{R}^d} g(x)f(x)dx,$$

avec  $f$  une densité de probabilité (c'est à dire  $f$  positive et  $\int_{\mathbb{R}^d} f(x)dx = 1$ ), alors nous pouvons écrire  $I = E[\varphi(X)]$  avec  $X$  de loi de densité  $f$ . Nous sommes encore dans la situation où nous voulons calculer l'espérance d'une variable aléatoire (si  $X$  est une variable aléatoire, alors  $\varphi(X)$  est une variable aléatoire, à condition que  $\varphi$  soit mesurable).

### 2.1.2 Loi forte des grands nombres

#### **Théorème 4.** (Convergence)

Soit  $(X_n)_{n \geq 1}$  une suite de v.a.i.i.d à valeur dans  $\mathbb{R}^d$ ,  $d \in \mathbb{N}^*$ ,  $X_i \sim \mathcal{L}(f)$ . On suppose que  $\varphi(x)$  est une fonction mesurable,  $(E[\varphi(X)] < +\infty)$ , alors :

$$\frac{1}{n} \sum_{i=1}^n \varphi(X_i) \xrightarrow[n \rightarrow +\infty]{p.s.} E[\varphi(X)].$$

Ce théorème nous dit pourquoi l'approximation de Monte-Carlo (2.2) est valide (et sous quelle hypothèse).

Autrement dit, afin d'estimer une quantité de type  $I = \int_{\mathbb{R}^d} \varphi(x)f(x)dx$  avec une méthode de Monte-Carlo on doit suivre les étapes suivantes :

**Etape1 :** Si  $\varphi$  est une fonction mesurable ( $E[\varphi(X)] < +\infty$ ), alors par l'application de la loi faible des grands nombres, on peut mettre la quantité  $I$ , sous forme d'une espérance, i.e.

$$I = \int_{\mathbb{R}^d} \varphi(x)f(x)dx = E[\varphi(X)].$$

**Etape2 :** Si on sait simuler  $\varphi(X_1), \dots, \varphi(X_n)$ , alors par l'application de la loi forte des grands nombres nous pouvons approcher la quantité  $I$  par un estimateur Monte-Carlo, i.e.

$$I = E[\varphi(X)] \approx \frac{1}{n} \sum_{i=1}^n \varphi(X_i).$$

Sous les hypothèses données précédemment et les lois des grands nombres, l'algorithme qui permet d'estimer une quantité de type  $I = \int_{\mathbb{R}^d} \varphi(x)f(x)dx$  est donné comme suit :

---

#### **Algorithme**

---

##### **Debut**

lire( $n$ ) ;

$s = 0$  ;

**pour**  $i = 1 : n$  **faire**

générer  $X(i) \sim \mathcal{L}(f)$  ;

$s = s + \varphi(X(i))$  ;

**fin pour** ;

$\hat{I}_n \approx \frac{s}{n}$  ;

afficher ( $\hat{I}_n$ ) ;

**Fin.**

---

### 2.1.3 Exemples d'application

#### **Exemple 7. Cas Classique**

Estimation de la quantité  $I = \int_a^b \varphi(x)dx$ .

Soit  $X_1, \dots, X_n$  une suite de v.a.i.i.d,  $X_1 \sim \mathcal{U}([a, b])$ . La quantité  $I$  peut-être donnée sous la forme suivante :

$$\begin{aligned} I &= (b-a) \int_a^b \varphi(x) \frac{1}{b-a} dx \\ &= (b-a) \int_a^b \varphi(x) f(x) dx. \end{aligned}$$

avec  $f(x) = \frac{1}{b-a} \mathbb{1}_{a \leq x \leq b}$ .

On suppose que  $E[\varphi(X)] < +\infty$ .

Alors d'après la loi faible des grands nombres nous avons

$$\int_a^b \varphi(x) f(x) dx = E[\varphi(X)].$$

Supposons que  $\varphi(X_1), \dots, \varphi(X_n)$  Sont i.i.d et d'après la loi forte des grands nombres

$$\frac{1}{n} \sum_{i=1}^n \varphi(X_i) \xrightarrow[n \rightarrow +\infty]{p.s.} E[\varphi(X)].$$

Posons  $\hat{I}_n = \frac{1}{n} \sum_{i=1}^n \varphi(X_i)$ , alors

$$I = (b-a) \int_a^b \varphi(x) f(x) dx \approx (b-a) \hat{I}_n.$$

Donc l'estimation de la quantité  $I$  se fait grâce à l'algorithme suivant :

---

#### Algorithme

---

**Debut**

lire( $n$ );

lire( $a$ );

lire( $b$ );

$s = 0$ ;

**pour**  $i = 1 : n$  **faire**

générer  $X(i) \sim \mathcal{U}([a, b])$ ;

$s = s + \varphi(X(i))$ ;

**fin pour**;

$\hat{I}_n \approx \frac{s}{n} (b-a)$ ;

afficher ( $\hat{I}_n$ );

**Fin.**

---

#### Exemple 8. Estimation de la valeur de $\pi$

Soit une suite de couple  $(X_n, Y_n)_{n>0}$  de v.a.i.i.d  $(X_1, Y_1) \sim \mathcal{U}([-1, 1])$ .

Posons  $\varphi(x, y) = \mathbb{1}_{x^2+y^2 \leq 1}$ .

L'aire du cercle d'unité est donnée par :

$$\begin{aligned} I &= \int_{-1}^1 \int_{-1}^1 \mathbb{1}_{x^2+y^2 \leq 1} dx dy \\ &= 4 \int_{-1}^1 \int_{-1}^1 \mathbb{1}_{x^2+y^2 \leq 1} dx dy \times \frac{1}{2} \mathbb{1}_{-1 \leq x \leq 1} \times \frac{1}{2} \mathbb{1}_{-1 \leq y \leq 1} dx dy \\ &= 4 \int_{-1}^1 \int_{-1}^1 \varphi(x, y) f(x, y) dx dy \end{aligned}$$

D'après la loi des grands nombres on aura,

$$\frac{1}{n} \sum_{i=1}^n \varphi(X_i, Y_i) \xrightarrow[n \rightarrow +\infty]{p.s.} E[\varphi(X, Y)].$$

D'où  $I \approx 4 \times \frac{1}{n} \sum_{i=1}^n \varphi(X_i, Y_i)$ . Mais, nous savons que l'air du cercle d'unité est  $\pi$ , d'où

$$\pi \approx 4 \times \frac{1}{n} \sum_{i=1}^n \varphi(X_i, Y_i).$$

Un des algorithmes d'estimation de la valeur de  $\pi$  est donné comme suit :

---

**Algorithme**

---

**Debut**

lire ( $n$ );

$s = 0$ ;

**pour**  $i = 1 : n$  **faire**

    générer  $U \sim \mathcal{U}([0, 1])$ ;

    générer  $V \sim \mathcal{U}([0, 1])$ ;

$X(i) = 2 * U - 1$ ;

$Y(i) = 2 * V - 1$ ;

**si**  $(X(i)^2 + Y(i)^2 \leq 1)$ ;

$s = s + 1$ ;

**fin si**;

**fin pour**;

$\pi = 4 * \frac{s}{n}$ ;

**Fin.**

---

Autrement dit, afin d'estimer la valeur de  $\pi$ , on lance une pièce de monnaie à l'intérieur du carré  $[-1, 1]^2$  et on cherche à déterminer la probabilité  $P$  que la pièce soit à l'intérieur du cercle d'unité. La probabilité que la pièce tombe à l'intérieur du cercle est donnée par :

$$\begin{aligned} P &= \frac{\text{nbre de points dans le cercle}}{\text{nbre de points total}} \\ &= \frac{\text{Aire du cercle}}{\text{Aire du carré}} \\ &= \frac{\pi R^2}{4R^2} = \frac{\pi}{4} \quad (R = 1). \end{aligned}$$

D'où

$$P = \frac{\pi}{4} \implies \pi = 4 * P$$

Pour estimer la valeur de  $P$  on lance  $n$  pièce à l'intérieur du carré, on accepte les pièces qui tombent à l'intérieur du cercle et on rejette les pièces qui tombent à l'extérieur. La figure (2.1) donne une valeur estimée de  $\pi$  et le nombre des points qui tombent à l'intérieur du cercle d'unité.

On dispose donc d'un estimateur Monte-Carlo pour estimer des quantités de type  $I = \int_{\mathbb{R}^d} \varphi(x) f(x) dx$ .

Encore faut-il connaître la précision de ces estimateurs. C'est tout l'intérêt de la section suivante.



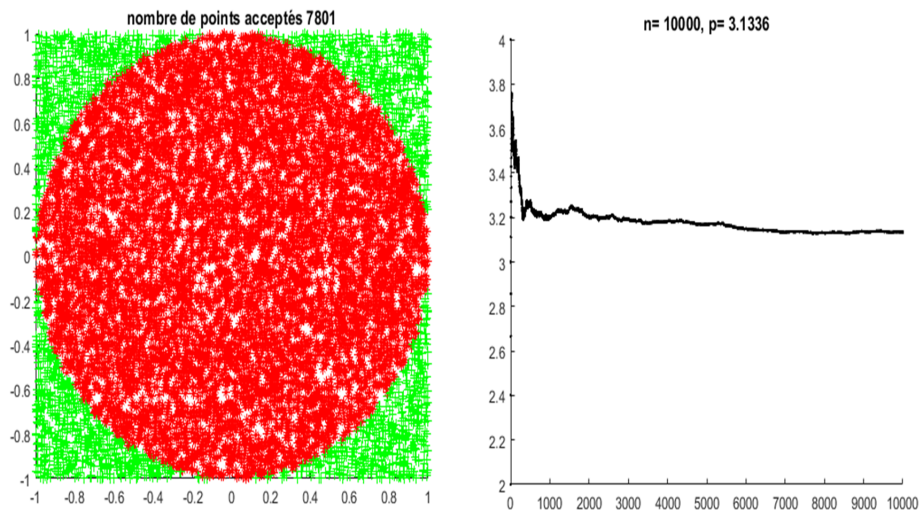


FIGURE 2.1 – Estimation de la valeur de  $\pi$ .

## 2.2 Erreur de l'estimation M.C

L'estimation d'une quantité  $E[\varphi(X)]$  par une méthode de Monte-Carlo génère une erreur aléatoire dont on ne peut pas la borner. En revanche, on peut donner un intervalle de confiance au résultat grâce au théorème centrale limite (TCL).

### Théorème 5. TCL

Soit  $\varphi(X_i)$ ,  $i = 1 \dots n$ , une suite de v.a.i.i.d à valeur dans  $\mathbb{R}^d$ ,  $d \in \mathbb{N}^*$ .

On suppose que  $E[\varphi(X)^2] \leq +\infty$ .

Soit  $\text{var}[\varphi(X)] = \sigma^2$  (Variance de  $\varphi(X)$ ), alors

$$\frac{\sqrt{n}}{\sigma} \left[ \frac{1}{n} \sum_{i=1}^n \varphi(X_i) - E[\varphi(X)] \right] \xrightarrow[n \rightarrow +\infty]{loi} \mathcal{N}(0, 1).$$

Notons l'erreur d'estimation  $\varepsilon_n = |\hat{I}_n - I| = \left| \frac{1}{n} \sum_{i=1}^n \varphi(X_i) - E[\varphi(X)] \right|$ .

Sous les hypothèses du théorème limite centrale, on cherche à approcher  $E[\varphi(X)]$  à  $\beta$  près avec une confiance de  $(1 - \alpha)\%$  (On cherche à construire un intervalle de confiance). C'est-à-dire que l'on veut calculer

$$P(|\varepsilon_n| \geq \beta) \leq \alpha,$$

ce qui est équivalent à

$$P(|\varepsilon_n| \leq \beta) \geq (1 - \alpha). \quad (2.3)$$

Nous avons

$$\begin{aligned} P(|\varepsilon_n| \leq \beta) &= P(-\beta \leq \varepsilon_n \leq \beta) \\ &= P\left(-\beta \frac{\sqrt{n}}{\sigma} \leq \frac{\sqrt{n}}{\sigma} \varepsilon_n \leq \frac{\sqrt{n}}{\sigma} \beta\right) \\ &= P\left(-\beta \frac{\sqrt{n}}{\sigma} \leq \frac{\sqrt{n}}{\sigma} \left( \frac{1}{n} \sum_{i=1}^n \varphi(X_i) - E[\varphi(X)] \right) \leq \frac{\sqrt{n}}{\sigma} \beta\right). \end{aligned}$$

D'après TCL :  $\frac{\sqrt{n}}{\sigma} \left( \frac{1}{n} \sum_{i=1}^n \varphi(X_i) - E[\varphi(X)] \right) \xrightarrow[n \rightarrow +\infty]{loi} \mathcal{N}(0, 1).$

Pour  $n$  assez grand

$$P(|\varepsilon_n| \leq \beta) \approx \int_{-\beta \frac{\sqrt{n}}{\sigma}}^{\beta \frac{\sqrt{n}}{\sigma}} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx.$$

Par symétrie de  $\mathcal{N}(0, 1)$ , on trouve,

$$P(|\varepsilon_n| \leq \beta) \approx 2 \int_{-\infty}^{\beta \frac{\sqrt{n}}{\sigma}} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx - 1.$$

Par substitution dans l'équation (2.3), on trouve,

$$\int_{-\infty}^{\beta \frac{\sqrt{n}}{\sigma}} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx \geq 1 - \alpha/2.$$

Par l'application de la fonction inverse  $\phi^{-1}$  de la loi normale  $\mathcal{N}(0, 1)$ , on trouve que

$$\begin{aligned} \frac{\sqrt{n}}{\sigma} \beta \geq \phi^{-1}(1 - \alpha/2) &\Rightarrow \sqrt{n} \geq \left( \frac{\phi^{-1}(1 - \alpha/2)}{\beta} \right) \sigma. \\ &\Rightarrow n \geq \left( \frac{\phi^{-1}(1 - \alpha/2)}{\beta} \right)^2 \sigma^2. \end{aligned}$$

On remarque que le nombre de tirage nécessaire  $n$  pour atteindre un certain niveau d'erreur avec une certaine confiance est une fonction linéaire en fonction  $\sigma^2$ .

**Proposition 3.** Soit  $\alpha \in [0, 1]$  fixé. Un intervalle de confiance de niveau asymptotique  $1 - \alpha$  pour  $I$  est,

$$\left[ \hat{I}_n - \phi^{-1}(1 - \alpha/2) \sqrt{\frac{\sigma^2}{n}}, \hat{I}_n + \phi^{-1}(1 - \alpha/2) \sqrt{\frac{\sigma^2}{n}} \right].$$

Où  $\phi^{-1}(1 - \alpha/2)$  désigne le quantile d'ordre  $1 - \alpha/2$  de la loi normale centrée réduite.

Dans la pratique la valeur de  $\sigma^2$  pourrait ne pas être connue mais on peut l'estimer par une méthode de Monte-Carlo grâce au théorème suivant.

**Théorème 6. Estimation de la variance**

Si  $\varphi(X_1) \dots \varphi(X_n)$  Sont i.i.d avec  $E[\varphi(X)^2] \leq +\infty$  et  $\text{var}[\varphi(X_1)] = \sigma^2$  alors,

$$\frac{1}{2n} \left[ (\varphi(X_1) - \varphi(X_2))^2 + \dots + (\varphi(X_{2n-1}) - \varphi(X_{2n}))^2 \right] \xrightarrow[n \rightarrow +\infty]{P.S} \sigma^2.$$

Dans ce cas l'intervalle de confiance au niveau  $1 - \alpha$  est donné par :

$$\left[ \hat{I}_n - \phi^{-1}(1 - \alpha/2) \sqrt{\frac{\hat{\sigma}_n^2}{n}}, \hat{I}_n + \phi^{-1}(1 - \alpha/2) \sqrt{\frac{\hat{\sigma}_n^2}{n}} \right].$$

Où  $\hat{\sigma}_n^2$  est un estimateur Monte-Carlo de la variance.

**Exemple 9.** Estimation de la variance de  $\pi$  ainsi que la construction de l'intervalle de confiance à  $(1 - \alpha) = 95\%$ .

Dans l'exemple de l'estimation de la valeur de  $\pi$ , nous avons posé

$\varphi(x, y) = \mathbb{1}_{x^2+y^2 \leq 1}(x, y)$  avec  $(X, Y) \sim \mathcal{U}([-1, 1])$ .

L'estimation de la variance et de l'intervalle de confiance de cette méthode sont donnés par l'algorithme suivant et la figure (2.2).

---

**Algorithme**

---

**Debut**

lire( $n$ );

var = 0;

**pour**  $i = 1 : n$  **faire**

générer  $U \sim \mathcal{U}[0, 1]$ ;

générer  $V \sim \mathcal{U}[0, 1]$ ;

générer  $T \sim \mathcal{U}[0, 1]$ ;

générer  $W \sim \mathcal{U}[0, 1]$ ;

$X(i) = 2 * U - 1$ ;

$Y(i) = 2 * V - 1$ ;

$Z(i) = 2 * T - 1$ ;

$R(i) = 2 * W - 1$ ;

**si**  $(X(i)^2 + Y(i)^2 \leq 1)$  **et**  $(Z(i)^2 + R(i)^2 \leq 1)$  **alors**

var = var +  $[(X(i)^2 + Y(i)^2) - (Z(i)^2 + R(i)^2)]^2$ ;

**fin si**;

**fin pour**;

$\hat{\sigma}_n^2 = \text{var} / (2 * n)$ ;

afficher( $\hat{\sigma}_n^2$ );

**Fin.**

---

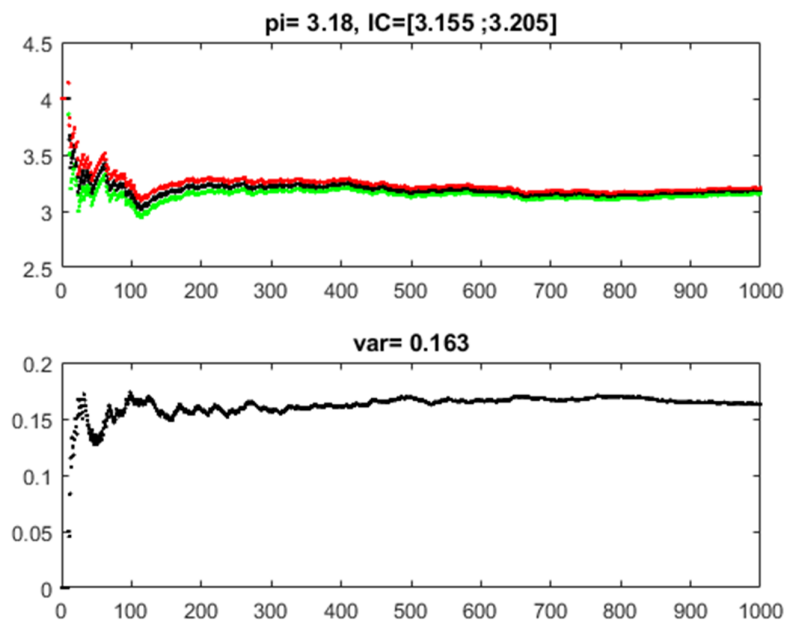


FIGURE 2.2 – Estimation de la valeur  $\pi$ , La variance de cette méthode et l'intervalle de confiance.

Avec 1000 simulations on obtient un estimateur qui vaut 3.1800 et une variance de 0.1630 pour un intervalle de confiance [3.1550, 3.2050].

Dans cette section nous avons vu que l'estimation d'une quantité de type  $I = E[\varphi(X)] = \int_{\mathbb{R}^d} \varphi(x)f(x)dx$  génère une erreur  $\varepsilon_n$  d'ordre  $\frac{\sigma^2}{n} = \text{var}[\varphi(X)]/n$ . Donc il est de notre intérêt de réduire la variance afin de minimiser l'erreur de l'estimation d'une part et minimiser le nombre d'itérations nécessaires à l'estimation de la variance d'autre part. En effet, supposons que l'on veuille estimer la quantité suivante :

$$I = E[\varphi(X)] = \int_{\mathbb{R}} \varphi(x)f(x)dx.$$

avec :

$$\begin{cases} \varphi(x) = e^{\gamma x} & \gamma \in \mathbb{R}^+ \\ f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} & x \in \mathbb{R} \end{cases}$$

Pour l'estimation de cette quantité on doit suivre les étapes suivantes.

1. On tire une suite de v.a.i.i.d  $X_1 \dots X_n$ ,  $X_1 \sim \mathcal{N}(0, 1)$ .
2. On montre que  $E[\varphi(X)] < +\infty$ .

Nous avons

$$\begin{aligned} E[\varphi(X)] &= \int_{\mathbb{R}} \varphi(x)f(x)dx \\ &= \int_{-\infty}^{+\infty} e^{\gamma x} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx \\ &= \int_{-\infty}^{+\infty} \frac{1}{\sqrt{2\pi}} e^{\left(-\frac{1}{2}(x-\gamma)^2 + \frac{\gamma^2}{2}\right)} dx \\ &= e^{\frac{\gamma^2}{2}} \left( \int_{-\infty}^{+\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(x-\gamma)^2} dx \right) \\ &= e^{\frac{\gamma^2}{2}}. \end{aligned}$$

D'après les points (1) et (2) et les lois des grands nombres, la quantité  $I$  peut-être approchée par un estimateur Monte-Carlo de la façon suivante :

$$I = E[\varphi(X)] \approx \frac{1}{n} \sum_{i=1}^n \varphi(X_i).$$

Nous avons vu précédemment qu'afin d'approcher la quantité  $E[\varphi(X)]$  à  $\beta$  près avec une confiance de  $(1 - \alpha)$ , nous aurons besoin de générer un nombre des échantillons qui dépend de la variance, i.e.

$$n \simeq \left( \frac{\phi^{-1}(1 - \alpha/2)}{\beta} \right)^2 \sigma^2. \quad (2.4)$$

**Calculons la variance de cette méthode.**

$$\begin{aligned} \sigma^2 &= \text{var}[\varphi(X)] \\ &= E[\varphi(X)^2] - E[\varphi(X)]^2. \end{aligned}$$

avec  $E[\varphi(X)]^2 = e^{\gamma^2}$ .

De la même méthode, on trouve  $E[(\varphi(X))^2] = e^{2\gamma^2}$ .

Donc

$$\begin{aligned} \sigma^2 &= e^{2\gamma^2} - e^{\gamma^2} \\ &= e^{\gamma^2}(e^{\gamma^2} - 1). \end{aligned}$$

Pour  $\beta = 0.01$ ,  $\alpha = 5\%$  et par substitution dans l'équation (2.4) on trouve

$$n \simeq \left( \frac{1,96}{0,01} \right)^2 e^{\gamma^2} (e^{\gamma^2} - 1)$$

Le tableau (2.1) donne quelques valeurs approximatives de  $n$  en fonction de la valeur de  $\gamma$ .

$\gamma$	3	4	5
$n \approx$	$3.10^{12}$	$3.10^{18}$	$2.10^{26}$

TABLE 2.1 – valeur de  $n$  en fonction de  $\gamma$

A partir du tableau (2.1), nous pouvons voir que la valeur minimale de  $n$  est pour  $\gamma = 3$ . En effet, pour  $\gamma = 3$ ,  $n \simeq 3.10^{12}$ . Autrement dit,  $n$  est d'ordre 12 ce qui n'est pas réalisable dans la pratique. C'est pourquoi il est important de réduire la variance.

## 2.3 Réduction de variance

Dans cette section, nous présentons des méthodes qui permettent d'améliorer la précision grâce à des techniques de réduction de variance. Rappelons ici qu'il existe plusieurs méthodes qui permettent de réduire la variance et donc d'améliorer l'erreur de précision. Mais nous nous contentons par présenter 3 de ces méthodes.

### 2.3.1 Méthode d'échantillonnage préférentiel

Supposons qu'on cherche à estimer la valeur de  $E[\varphi(X)]$  où  $X$  est une variable à valeur dans  $\mathbb{R}^d$ , de densité  $f$ .

Pour toute densité  $h \geq 0$  nous pouvons écrire

$$\begin{aligned} E[\varphi(X)] &= \int_{\mathbb{R}^d} \varphi(x) f(x) dx \\ &= \int_{\mathbb{R}^d} \frac{\varphi(x) f(x)}{h(x)} h(x) dx \\ &= E \left[ \frac{\varphi(Y) f(Y)}{h(Y)} \right]. \end{aligned}$$

où  $Y$  est une variable à valeur dans  $\mathbb{R}^d$  de densité  $h$ .

Nous disposons donc deux méthodes pour estimer  $I = E[\varphi(X)]$ .

#### Méthode de Monte Carlo Classique

$$I \approx \hat{I}_n^1 = \frac{1}{n} \sum_{i=1}^n \varphi(X_i) \quad X_i, i = 1..n \text{ une suite de v.a.i.i.d de densité } f.$$

#### Méthode d'échantillonnage préférentiel

$$I \approx \hat{I}_n^2 = \frac{1}{n} \sum_{i=1}^n \frac{\varphi(Y_i) f(Y_i)}{h(Y_i)} \quad Y_i, i = 1..n \text{ une suite de v.a.i.i.d de densité } h.$$

Afin d'appliquer cette méthode, on doit choisir une densité  $h$ . Dans ce qui suit, nous verrons, comment peut-on choisir la densité  $h$ .

Supposons que  $\varphi > 0$ , le choix le plus simple de  $h$  est  $h : y \mapsto \frac{\varphi(y) f(y)}{E[\varphi(X)]}$ . Ce qui bien une densité

de probabilité.

Afin de comparer entre les deux méthodes, on doit comparer entre les variances de ces deux dernières. En effet, nous pouvons dire que la méthode d'échantillonnage préférentielle est plus intéressante que la méthode de Monte-Carlo classique si

$$\text{var} \left[ \frac{\varphi(Y)f(Y)}{h(Y)} \right] \leq \text{var}[\varphi(X)].$$

Nous avons alors

$$\text{var} \left[ \frac{\varphi(Y)f(Y)}{h(Y)} \right] = E \left[ \left( \frac{\varphi(Y)f(Y)}{h(Y)} \right)^2 \right] - E \left[ \left( \frac{\varphi(Y)f(Y)}{h(Y)} \right) \right]^2. \quad (2.5)$$

Nous avons  $h(y) = \frac{\varphi(y)f(y)}{E[\varphi(Y)]}$

Par substitution dans l'équation (2.5), on trouve :

$$\text{var} \left[ \frac{\varphi(Y)f(Y)}{h(Y)} \right] = E[(\varphi(Y))^2] - E[\varphi(Y)]^2 = 0.$$

Nous avons ici une méthode de Monte-Carlo de variance nulle, ce qui semble n'avoir aucun sens. Donc, il est de notre intérêt de choisir une autre densité  $h$ . L'idée donc serait de trouver une autre densité  $h$ .

La discussion ci-dessus nous donne une idée d'une démarche à suivre pour le choix de la densité  $h$  et ainsi réduire la variance.

1. Trouver une fonction grossièrement proche de  $|\varphi.f|$  telle que l'on sache simuler suivant la densité :

$$h(y) = \frac{h_1(y)}{\int_{\mathbb{R}^d} h_1(y) dy}.$$

2. On suppose que  $Y_i$ ,  $i = 1..n$  est une suite de v.a.i.i.d,  $Y_1 \sim \mathcal{L}(h)$ . Puis on compare  $\text{var} \left[ \frac{\varphi(Y)f(Y)}{h(Y)} \right]$  et  $\text{var}[\varphi(X)]$ . Pour cela, on utilise généralement le théorème (6) (estimation de variance).

**Exemple 10.** Supposons qu'on cherche à approcher  $I = \int_0^1 \sin(\pi x) dx$  par une méthode Monte-Carlo.

La quantité  $I$  peut être estimée par deux méthodes différentes.

### Méthode de Monte Carlo Classique

$$I \approx \hat{I}_n^1 = \frac{1}{n} \sum_{i=1}^n \varphi(X_i), \quad \varphi(X_i) = \sin(\pi X_i), \quad X_1, \dots, X_n \text{ i.i.d de loi } \mathcal{U}([0, 1]).$$

### Méthode échantillonnage préférentiel

$$I \approx \hat{I}_n^2 = \frac{1}{n} \sum_{i=1}^n \frac{\varphi(Y_i)f(Y_i)}{h(Y_i)}, \quad Y_i \sim \mathcal{L}(h), \quad i = 1..n.$$

avec :

$$\begin{cases} \varphi(y) = \sin(\pi y) \\ f(y) = \mathbb{1}_{[0,1]}(y) \end{cases}$$

Afin d'estimer la quantité  $I$  avec la méthode d'échantillonnage préférentiel, on définit une fonction  $h_1$  grossièrement proche de  $|\varphi.f|$ .

Posons  $h_1(y) = (1 - y)\mathbb{1}_{[0,1]}(y)$ , donc

$$h(y) = \frac{h_1}{\int_0^1 h_1(y) dy} = \frac{1 - y}{\int_0^1 (1 - y) dy} = 2(1 - y),$$

d'où

$$h(y) = 2(1 - y)\mathbb{1}_{[0,1]}(y).$$

### Simulation de $Y$ suivant $h$

Par l'application de la méthode d'inversion (vue dans le chapitre 1), nous avons

$$\begin{aligned} H(y) &= \int_0^y 2(1 - t) dt \\ &= \left[ (1 - t)^2 \right]_0^y \\ &= (1 - y)^2 - 1. \end{aligned}$$

Pour tout  $0 \leq u \leq 1$ , posons  $u = H(y) = (1 - y)^2 - 1 \implies y = 1 - \sqrt{1 + u}$ .

Donc  $Y_i = 1 - \sqrt{1 + U_i}$  avec  $U_1, \dots, U_n \sim \mathcal{U}([0, 1])$ .

L'algorithme suivant donne l'estimation de la variance des deux méthodes.

---

#### Algorithme

---

##### Debut

lire  $(n)$ ;

**pour**  $i = 1 : n$  **faire**

générer  $U(i) \sim \mathcal{U}([0, 1])$ ;

générer  $V(i) \sim \mathcal{U}([0, 1])$ ;

$$Y(i) = 1 - \sqrt{(U(i) + 1)};$$

$$Z(i) = 1 - \sqrt{(V(i) + 1)};$$

**fin pour**;

%estimation de variance méthode classique.

var1 = 0;

**pour**  $i = 1 : n$  **faire**

$$\text{var1} = \text{var1} + [\sin(\pi * U(i)) - \sin(\pi * V(i))]^2;$$

**fin pour**;

$$\text{var1} = \frac{\text{var1}}{(2 * n)};$$

%estimation de variance méthode échantillonnage préférentiel.

var2 = 0;

**pour**  $i = 1 : n$  **faire**

$$\text{var2} = \text{var2} + \left[ \left( \frac{\sin(\pi * Y(i))}{2(1 - Y(i))} \right) - \left( \frac{\sin(\pi * Z(i))}{2(1 - Z(i))} \right) \right]^2;$$

**fin pour**;

$$\text{var2} = \frac{\text{var2}}{(2 * n)};$$

**Fin.**

---

D'après le programme de cet algorithme (voir figure(2.3)), nous trouvons  $\text{var1} = \text{var}[\varphi(X)] = 0.1$  pour la méthode de Monte-Carlo classique et  $\text{var2} = \text{var}\left[\frac{\varphi(Y)f(Y)}{h(Y)}\right] = 0.009$  pour la méthode

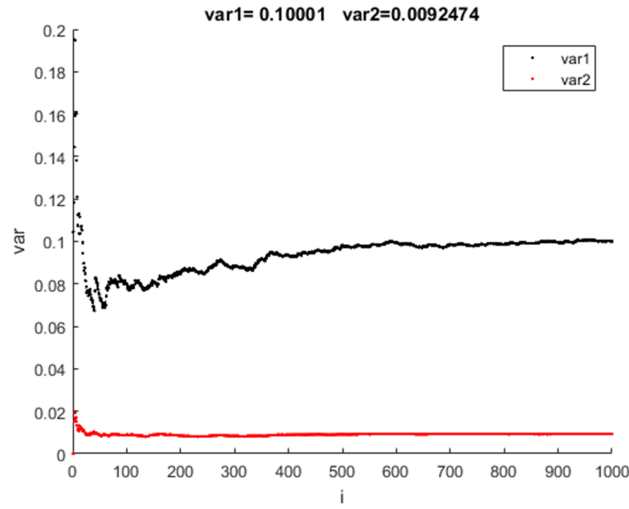


FIGURE 2.3 – La courbe en noir représente l'estimation de la variance de la méthode de M.C classique. La courbe en rouge représente l'estimation de la variance de la méthode d'échantillonnage préférentiel.

d'échantillonnage préférentiel. La variance est donc réduite avec la méthode d'échantillonnage préférentiel. De plus, le nombre des itérations nécessaires pour approcher la quantité  $\int_0^1 \sin(\pi x) dx$  à 0.01 près avec une confiance de 95% est de  $n = 3842$  pour la première méthode. Tandis qu'il est de  $n = 355$  pour la deuxième méthode.

Avec la méthode classique de Monte-Carlo et pour 1000 simulations on obtient un estimateur qui vaut 0.63859 pour un intervalle de confiance  $IC_{0.95} = [0.61899; 0.65819]$ . Tandis qu'avec la méthode d'échantillonnage préférentiel et pour le même nombre de simulations, on obtient un estimateur qui vaut 0.6371 pour un intervalle de confiance  $IC_{0.95} = [0.63122; 0.64298]$  (voir figure (2.4)). Donc, il est très avantageux d'utiliser la méthode d'échantillonnage préférentiel. La variance est en effet 11 fois plus petite pour un temps de simulation inférieur à 10 fois et un intervalle de confiance 2 fois plus réduit que celui de Monte Carlo.

### 2.3.2 Méthode de Variable de Contrôle

Nous cherchons à calculer  $E[\varphi(X)]$  avec  $X$  variable à valeur dans  $\mathbb{R}^d$  de densité  $f$ .

#### Méthode classique

Supposons que  $E[\varphi(X)] < +\infty$ , alors d'après les lois des grands nombres nous avons,

$$I \approx \hat{I}_n^1 = \frac{1}{n} \sum_{i=1}^n \varphi(X_i) \quad X_i, i = 1..n \text{ v.a i.i.d de densité } f.$$



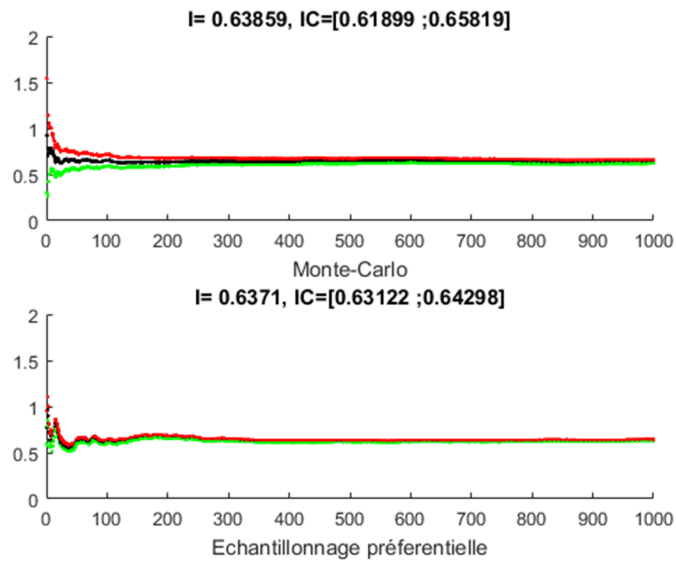


FIGURE 2.4 – Les valeurs approchées de la quantité  $I$  et les intervalles de confiance de la méthode M.C classique et celle d'échantillonnage préférentiel.

### Méthode variable de contrôle

La deuxième méthode est la suivante

Si on sait calculer (pas approcher) la quantité  $E[h(X)]$ , d'une certaine fonction  $h$ , alors on peut aussi faire l'estimation suivante :

$$E[\varphi(X)] \approx \hat{I}_n^2 = \frac{1}{n} \sum_{i=1}^n [\varphi(X_i) - h(X_i)] + E[h(X)].$$

En effet, nous avons

$$\begin{aligned} E[\varphi(X)] &= \int \varphi(x) f(x) dx \\ &= \int \varphi(x) f(x) - f(x) h(x) + f(x) h(x) dx \\ &= \int (\varphi(x) - h(x)) f(x) dx + \int f(x) h(x) dx \\ &= \frac{1}{n} \sum_{i=1}^n [\varphi(X_i) - h(X_i)] + E[h(X)]. \end{aligned}$$

Donc la méthode de variable de contrôle est la suivante.

1. Trouver une fonction  $h$  proche de  $\varphi$ , telle que l'on sache calculer  $E[h(X)]$  (le fait que  $h$  soit proche de  $\varphi$  signifie que la quantité  $\text{var}[\varphi(X) - h(X)]$  est petite).
2. Estimer les variances  $\text{var}[\varphi(X)]$  et  $\text{var}[\varphi(X) - h(X)]$  et les comparer.

**Exemple 11.** Dans cet exemple, nous donnons deux méthodes de Monte-Carlo différentes pour l'estimation de la quantité  $I = \int_0^1 e^{x^2} dx$ . Posons

$$\begin{cases} f(x) = \mathbb{1}_{[0,1]}(x), \\ \varphi(x) = \frac{1}{3} e^{x^2}. \end{cases}$$

## Méthode M.C classique

$$I \approx \hat{I}_n^1 = \frac{1}{n} \sum_{i=1}^n \varphi(X_i), \quad X_1, \dots, X_n, \text{ i.i.d, } X_1 \sim \mathcal{U}([0, 1]).$$

## Méthode variable de contrôle

Nous avons  $\varphi(x) = e^{x^2}$ , remarquons que  $h : x \mapsto 1 + x^2$  est grossièrement proche de  $\varphi$  sur  $[0, 1]$  (le développement limité de  $\varphi$  en 0 ).

Le choix de  $h$  est motivé par le fait, qu'on sache calculer d'une façon exacte la quantité  $E[h(X)]$ . En effet, nous savons,

$$E[h(X)] = \int_0^1 f(x)h(x)dx = \int_0^1 (1 + x^2)dx = \frac{4}{3}.$$

Donc nous pouvons faire l'estimation suivante

$$I \approx \hat{I}_n^2 = \frac{1}{n} \sum_{i=1}^n [\varphi(X_i) - h(X_i)] + \frac{4}{3}, \quad X_1, \dots, X_n, \text{ i.i.d, } X_1 \sim \mathcal{U}([0, 1]).$$

Nous estimons les deux variances dans l'algorithme suivant.

---

**Algorithme**

---

**Debut**  
lire (n);  
**pour** i = 1 : n **faire**  
    générer  $U(i) \sim \mathcal{U}([0, 1])$ ;  
    générer  $V(i) \sim \mathcal{U}([0, 1])$ ;  
**fin pour**;  
% estimation de variance méthode classique;  
var1 = 0;  
**pour** i = 1 : n **faire**  
    var1 = var1 +  $[\exp(U(i)^2) - \exp(V(i)^2)]^2$ ;  
**fin pour**;  
var1 =  $\frac{\text{var1}}{(2 * n)}$ ;  
% estimation de variance méthode variable de contrôle;  
var2 = 0;  
**pour** i = 1 : n **faire**  
    var2 = var2 +  $[(\exp(U(i)^2) - (1 + U(i)^2)) - (\exp(V(i)^2) - (1 + V(i)^2))]^2$ ;  
**fin pour**;  
var2 =  $\frac{\text{var2}}{(2 * n)} + \frac{4}{3}$ ;  
**Fin.**

---

Il n'est également pas surprenant de voir à l'aide des deux figures (figure(2.5) et figure(2.6)) que la variance est plus faible en utilisant la méthode de variable de contrôle. Avec cette méthode est pour 1000 échantillons, la variance est presque 7 fois plus petite pour un temps de simulation inférieur à 6 fois et un intervalle de confiance 2 fois plus réduit que celui de la méthode de Monte-Carlo classique.

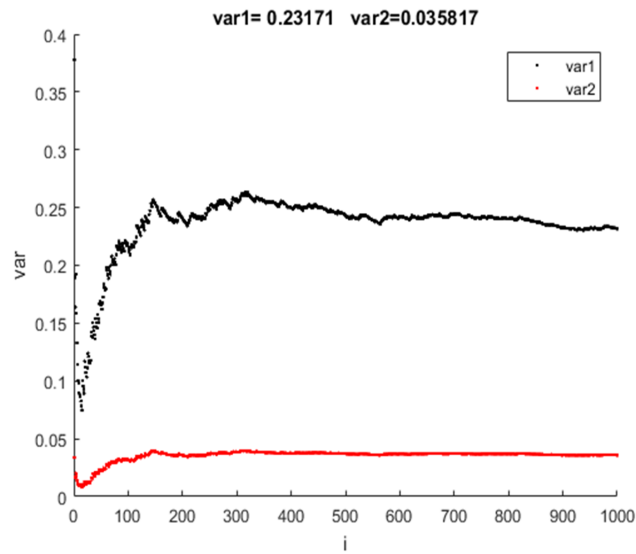


FIGURE 2.5 – La courbe en noir représente l'estimation de la variance de la méthode de M.C classique. La courbe en rouge représente l'estimation de la variance de la méthode variable de contrôle.

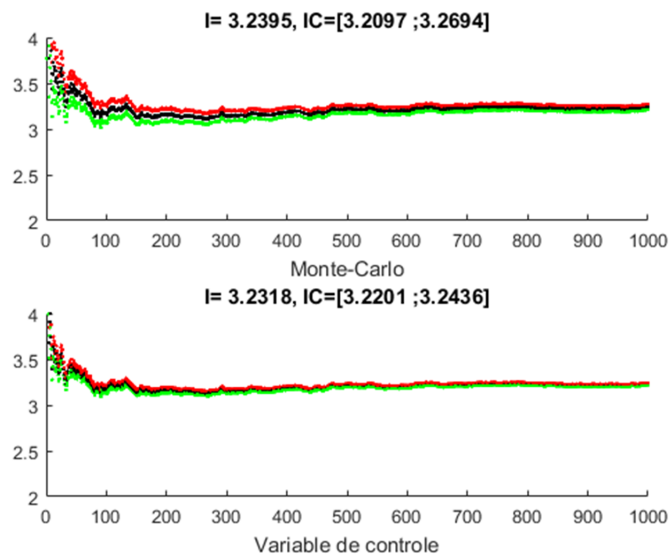


FIGURE 2.6 – Les valeurs approchées de la quantité  $I$  et les intervalles de confiance de la méthode M.C classique et celle de variable de contrôle.

### 2.3.3 Méthode variable antithétique

Supposons que l'on cherche à estimer la quantité suivante :

$$\begin{aligned}
 I &= \int_{[0,1]^d} \varphi(x_1, \dots, x_d) dx_1 \dots dx_d \\
 &= \int_{[0,1]^d} \varphi(x_1, \dots, x_d) f(x_1) \dots f(x_d) dx_1 \dots dx_d.
 \end{aligned}$$

avec  $f(x_d) = \mathbb{1}_{[0,1]}(x_d)$ .

### Méthode de Monte-Carlo classique

Pour  $E[\varphi(X)] < +\infty$  et d'après les lois des grands nombres la quantité  $I$  peut être approchée par l'estimateur Monte-Carlo  $\hat{I}_n^1$  de la façon suivante :

$$I \approx \hat{I}_n^1 = \frac{1}{n} \sum_{i=1}^n \varphi(X_i), \quad X_1, X_2, \dots, X_n, \text{ n v.a.i.i.d, } X_1 \sim \mathcal{U}([0, 1]).$$

### Méthode variable antithétique

Pour  $U \sim \mathcal{U}([0, 1])$ , nous avons  $(1, \dots, 1) - U$  est de même loi que  $U$ , alors nous pouvons écrire

$$I \approx E \left[ \frac{\varphi[(1, \dots, 1) - U] + \varphi(U)}{2} \right].$$

Ce qui donne l'idée pour une deuxième méthode.

$$I \approx \hat{I}_n^2 = \frac{1}{n} \sum_{i=1}^n \left[ \frac{\varphi[(1, \dots, 1) - X_i] + \varphi(X_i)}{2} \right].$$

**Lemme 1.** *Sous les hypothèses ci-dessus, nous avons,*

$$\text{var} \left[ \frac{\varphi[(1, \dots, 1) - U] + \varphi(U)}{2} \right] \leq \text{var}[\varphi(U)].$$

La variance est donc toujours réduite.

**Preuve 3.** *Nous avons,*

$$\begin{aligned} & \text{var} \left[ \frac{\varphi[(1, \dots, 1) - U] + \varphi(U)}{2} \right] \\ &= E \left[ \left( \frac{\varphi[(1, \dots, 1) - U] + \varphi(U)}{2} \right)^2 \right] - \underbrace{E \left[ \frac{\varphi[(1, \dots, 1) - U] + \varphi(U)}{2} \right]^2}_A \\ &= \frac{1}{4} E \left[ \varphi[(1, \dots, 1) - U]^2 + \varphi(U)^2 + 2\varphi[(1, \dots, 1) - U]\varphi(U) \right] - A \\ &= \frac{1}{4} E[\varphi[(1, \dots, 1) - U]^2] + \frac{1}{4} E[\varphi(U)^2] + \frac{1}{2} E[\varphi[(1, \dots, 1) - U]\varphi(U)] - A \end{aligned}$$

On a  $E(a \times b) \leq E(a^2)^{1/2} \times E(b^2)^{1/2}$  (Inégalité de Schwarz)

$$\begin{aligned} \text{var} \left[ \frac{\varphi[(1, \dots, 1) - U] + \varphi(U)}{2} \right] &\leq \frac{1}{4} E[\varphi[(1, \dots, 1) - U]^2] + \frac{1}{4} E[\varphi(U)^2] \\ &\quad + \frac{1}{2} \left[ E[\varphi[(1, \dots, 1) - U]^2]^{1/2} \times E[\varphi(U)^2]^{1/2} \right] - A \end{aligned}$$

nous avons  $(1, \dots, 1) - U \xrightarrow{\text{loi}} U$ .

$$\begin{aligned} &\leq \frac{1}{4} E[\varphi(U)^2] + \frac{1}{4} E[\varphi(U)^2] + \frac{1}{2} \left[ E[\varphi(U)^2]^{1/2} \times E[\varphi(U)^2]^{1/2} \right] - E[\varphi(U)]^2 \\ &\leq \frac{1}{2} E[\varphi(U)^2] + \frac{1}{2} E[\varphi(U)^2] - E[\varphi(U)]^2 \\ &\leq \text{var}[\varphi(U)]. \end{aligned}$$

**Remarque 4.** *Le même résultat est encore valable si on remplace l'intervalle  $[0, 1]^d$  par un domaine quelconque  $E$  de  $\mathbb{R}^d$ ,  $d \in \mathbb{N}^*$ , et  $(1 - u_1, \dots, 1 - u_d) \in [0, 1]^d$  par une transformation bijective  $g : E \rightarrow E$  telle que pour tout  $x \in E$ ,  $\varphi(g(x)) = \varphi(x)$ .*



## Chapitre 3

# Méthode de Monte-Carlo par chaîne de Markov

Dans le chapitre précédent, nous avons vu que toute quantité de type

$$I = E[\varphi(X)] = \int_{\mathbb{R}^d} \varphi(x) f(x) dx \quad d \in \mathbb{N}^*, \quad (3.1)$$

peut-être approchée par un estimateur Monte-Carlo ;

$$\hat{I}_n = \frac{1}{n} \sum_{i=1}^n \varphi(X_i). \quad (3.2)$$

Dans ce cas la suite  $(X_n)_{n \geq 1}$  est i.i.d  $X_1 \sim \mathcal{L}(f)$ .

Dans ce chapitre, nous nous intéressons au cas où la suite  $(X_n)_{n \geq 1}$  n'est pas i.i.d mais constitue une chaîne de Markov de loi stationnaire  $f$ . On parle alors de méthode Monte-Carlo par chaîne de Markov (MCMC). Nous commençons ce chapitre par un rappel sur les chaînes de Markov, puis nous présentons les deux types de techniques les plus importantes conçues pour créer des chaînes de Markov de loi stationnaire donnée, à savoir les algorithmes de Metropolis-Hastings et l'échantillonnage de Gibbs.

### 3.1 Généralités sur les chaînes de Markov

Dans ce qui suit, nous donnons quelques propriétés sur les chaînes de Markov qui seront utiles pour la suite.

#### 3.1.1 Chaînes de Markov

On appelle une chaîne de Markov à temps discret toute suite de v.a  $(X_n)_{n \geq 1}$  à valeur dans un ensemble  $S$  supposé fini,  $S = \{1, 2, \dots, M\}$  appelé espace des états, si pour tout  $n \geq 1$ , et pour toute suite  $(x_1, \dots, x_{n-1}, x, y)$  telle que  $P(X_{n+1} = y | X_n = x, X_{n-1} = x_{n-1}, \dots, X_1 = x_1) \geq 0$  alors

$$\begin{aligned} P(X_{n+1} = y | X_n = x, X_{n-1} = x_{n-1}, \dots, X_1 = x_1) &= P(X_{n+1} = y | X_n = x) \\ &= P(X_2 = y | X_1 = x). \end{aligned}$$

Autrement dit, sachant le présent, le futur est indépendant du passé.

De plus si  $P(X_{n+1} = y | X_n = x) = P(X_2 = y | X_1 = x)$  alors la chaîne de Markov  $(X_n)_{n \geq 1}$  est dite homogène.

#### 3.1.2 Probabilité de transition

On appelle probabilité de transition, la probabilité d'aller de l'état  $x$  vers l'état  $y$  la quantité,

$$p_{xy} = P(X_{n+1} = y | X_n = x) = P(X_2 = y | X_1 = x). \quad (3.3)$$

### 3.1.3 Matrice de transition (noyau, opérateur)

On appelle matrice de transition ou matrice stochastique la matrice  $P = (p_{xy})_{x,y \in S}$  qui vérifie les propriétés suivantes

- pour tout  $x, y \in S$   $0 \leq p_{xy} \leq 1$ .
- pour tout  $x \in S$   $\sum_{y \in S} p_{xy} = 1$  (Somme des lignes).

### 3.1.4 Propriétés fondamentales

#### Probabilités de transition en $n$ étapes

La probabilité d'aller de l'état  $x$  vers l'état  $y$  en  $n$  coups est notée,

$$p_{xy}^{(n)} = P(X_n = y | X_1 = x). \quad (3.4)$$

et la matrice de transition en  $n$  coups est notée,

$$P^n = (p_{xy}^{(n)})_{x,y \in S}. \quad (3.5)$$

#### Loi de probabilité de $X_n$

Soit  $X_1$  la position initiale, on définit la loi de  $X_1$  comme un vecteur ligne de taille  $M$

$$\begin{aligned} \mu &= [\mu(1), \dots, \mu(i), \dots, \mu(M)] \\ &= [P(X_1 = 1), \dots, P(X_1 = i), \dots, P(X_1 = M)]. \end{aligned}$$

**Proposition 4.** Soit  $(X_n)_{n \geq 1}$  une chaîne de Markov de loi initiale  $\mu$  et de matrice de transition  $P$ . Alors pour tout entier naturel  $n \geq 1$ , la loi de  $X_n$  est donnée par

$$P(X_n) = \mu P^n. \quad (3.6)$$

### 3.1.5 Classification des états d'une chaîne de Markov

Soit  $(X_n)_{n \geq 1}$  une chaîne de Markov d'espace d'états  $S$ .

#### Irréductibilité

Soit  $x, y \in S$ , deux états, on dit que  $y$  est accessible à partir de  $x$  si  $\exists n \in \mathbb{N}$  telle que,

$$p_{xy}^{(n)} = P(X_n = y | X_1 = x) > 0. \quad (3.7)$$

Autrement, une chaîne est dite irréductible si tous les états communiquent entre eux.

#### Réurrence et Transience

Soit  $x \in S$ , on définit le temps de retour à  $x$  lorsque la chaîne  $(X_n)$  part de  $x$ , par :

$$T_x = \inf \{n \geq 1, X_n = x\}. \quad (3.8)$$

Un état  $x \in S$  est dit récurrent si partant de  $x$  on y revient sûrement en temps fini, i.e. :

$$P(T_x < +\infty | X_1 = x) = 1. \quad (3.9)$$

L'état  $x$  est transitoire dans le cas contraire, i.e. :

$$P(T_x = +\infty | X_1 = x) > 0. \quad (3.10)$$

Autrement dit, un état  $x$  est transitoire si avec une probabilité strictement positive, on peut le quitter pour ne jamais y revenir.



## Apériodicité

On définit la période d'un état  $x$ ,  $x \in S$  par

$$d_x = \text{PGCD} \left\{ n \geq 1, p_{xx}^{(n)} > 0 \right\}. \quad (3.11)$$

Un état  $x$  est dit apériodique si sa période est 1, sinon il est périodique.

Une chaîne de Markov est dite apériodique si tous ses états sont apériodiques.

## Ergodicité

Une chaîne de Markov homogène, irréductible, récurrente positive et apériodique est dite ergodique.

## Absorption

Soit  $x \in S$  on dit que  $x$  est un état absorbant si :

$$p_{xx} = 1. \quad (3.12)$$

### 3.1.6 Distribution stationnaire

Une probabilité  $\pi$  est appelée probabilité invariante ou probabilité stationnaire d'une chaîne de Markov de matrice de transition  $P$  si elle satisfait l'équation de balance globale,

$$\pi = \pi P. \quad (3.13)$$

Où  $\pi = (\pi_1, \pi_2, \dots, \pi_x, \dots)$  avec  $\sum_{x \in S} \pi_x = 1$ .

**Proposition 5.** *Le fait qu'une chaîne de Markov homogène soit irréductible récurrente positive, c'est-à-dire ergodique assure l'existence d'une unique probabilité invariante (stationnaire).*

### 3.1.7 Réversibilité(Symétrie)

On dit qu'une chaîne de Markov de matrice de transition  $P$ , ou plus simplement la matrice  $P$  est réversible(Symétrique) par rapport à la probabilité  $\pi$  Si on a pour tous  $x, y \in S$ ,

$$\pi_x p_{xy} = \pi_y p_{yx}. \quad (3.14)$$

En sommant (3.14) Sur  $y$  on en déduit le lemme suivant.

**Lemme 2.** *Si une chaîne de Markov est réversible par rapport à la probabilité  $\pi$ , alors  $\pi$  est une probabilité invariante.*

**Preuve 4.** *Pour montrer que  $\pi$  est une probabilité invariante, il suffit de montrer que :*

$$\forall x \in S \quad \pi_x = \sum_{y \in S} \pi_y p_{yx}.$$

$$\begin{aligned} \sum_{y \in S} \pi_y p_{yx} &= \sum_{y \in S} \pi_x p_{xy} \\ &= \sum_{y \in S} \pi_x P(X_2 = y | X_1 = x) \\ &= \pi_x. \end{aligned}$$

□

### 3.1.8 Lois des grands nombres (convergence et vitesse de convergence)

#### **Théorème 7.** (Convergence)

Soit  $(X_n)_{n \geq 1}$  une chaîne de Markov irréductible, admet une unique distribution stationnaire  $\pi$ , alors, pour toute fonction  $\varphi : S \rightarrow \mathbb{R}$  On a :

$$\frac{1}{n} \sum_{i=1}^n \varphi(X_i) \xrightarrow[n \rightarrow +\infty]{p.s.} E[\varphi(X)] = \sum_{x \in S} \varphi(x) \pi_x.$$

Dans le cas continu on a  $E[\varphi(X)] = \int_{x \in S} \varphi(x) \pi_{dx}$ .

Autrement dit, ce théorème, nous permet d'approcher une intégrale (cas continu) par une moyenne empirique, dans ce cas, la suite  $(X_n)_{n \geq 1}$  n'est pas i.i.d mais constitue une chaîne de Markov.

#### **Théorème 8.** (Vitesse de convergence(TCL))

Supposons que  $S$  est fini et  $P$  est une matrice de transition irréductible, apériodique et admettant une probabilité invariante  $\pi$ , alors :

— Il existe deux réels  $\alpha \in [0, 1[$ ,  $\beta \in \mathbb{R}$  tels que pour tout  $x, y \in S$ ,

$$|P_{xy}^n - \pi_y| \leq \beta \alpha^n.$$

— Pour tout  $x \in S$  et toute fonction  $\varphi : S \rightarrow \mathbb{R}$ , si on appelle  $(X_n)_{n \geq 1}$  une chaîne de Markov de loi initiale  $\pi$  et de matrice de transition  $P$  alors,

$$\sqrt{n} \left[ \frac{1}{n} \sum_{i=1}^n \varphi(X_i) - E[\varphi(X)] \right] \xrightarrow[n \rightarrow +\infty]{loi} \mathcal{N}(0, \sigma^2).$$

Avec une variance  $\sigma^2 < \infty$ .

Après avoir présenté quelques notions et propriétés sur les chaînes de Markov, l'objectif est maintenant de construire des algorithmes MCMC qui produisent des chaînes de Markov vérifiant ces propriétés, ces algorithmes sont décrits dans la section suivante.

## 3.2 Méthode de Monte-Carlo par chaîne de Markov

La méthode MCMC est publiée en 1953 par Metropolis et ses co-auteurs, elle étendue en 1970 par Hastings. En 1984, les frères Geman proposent l'échantillonneur de Gibbs pour la restauration bayésienne d'images ; cet échantillonneur est développé par Gelfand et Smith.

C'est au début des années 90, après le développement et la démocratisation de l'outil informatique, que ces méthodes rencontrent un important succès.

L'idée de base de ces méthodes est la construction d'une chaîne de Markov ergodique de loi stationnaire  $\pi$  : partant d'une valeur arbitraire  $X_1$ , on génère une chaîne  $(X_n)_{n \geq 1}$  à partir d'un noyau de transition de loi stationnaire  $\pi$ , qui garantit de plus la convergence en loi vers  $\pi$ .

Autrement dit, dans la section précédente, on a considéré une chaîne de Markov définie d'une façon ou d'une autre, nous chercherons sa loi stationnaire  $\pi$ . Désormais, la démarche est inverse : on part d'une loi  $\pi$  donnée et on cherche à construire une chaîne de Markov dont  $\pi$  est la mesure stationnaire.

Parmi les méthodes MCMC, l'algorithme de Hastings-Metropolis (Hastings 1970) et l'algorithme de Gibbs (Geman et Geman 1984) sont les plus utilisés et ont donné lieu à de nombreux algorithmes dérivés.

### 3.2.1 Algorithme de Metropolis-Hastings

Les définitions données dans la section précédente permettent de trouver la loi stationnaire d'une chaîne de Markov donnée. Les algorithmes que nous allons présenter dans cette section font l'inverse. Autrement dit, on part d'une loi  $\pi$  donnée, et on cherche à construire une chaîne de Markov dont  $\pi$  est sa distribution stationnaire. Nous commencerons notre visite des algorithmes MCMC par celui de Metropolis présenté dans la sous section suivante.

#### Algorithme de Metropolis

L'algorithme de Metropolis a été proposé par Metropolis et al 1953. Cet algorithme consiste à construire une chaîne de Markov qui admet  $\pi$  comme unique distribution stationnaire et possède la propriété ergodique. Pour ce faire, on fixe une loi  $\pi$  suivant laquelle on aimerait simuler ou dont on voudrait estimer une quantité de type  $\int_S \varphi(x) \pi_{dx}$ , nous appellerons  $\pi$  la loi cible.

Soit  $P$  la matrice de transition ayant  $\pi$  comme loi stationnaire.

Afin de construire une chaîne de Markov  $(X_n)_{n \geq 1}$  on suit le même principe de la méthode de rejet.

Soit donc  $Q = (q_{xy})_{x,y \in S}$  une matrice de transition (appelée aussi matrice de proposition) telle que :

- Pour tout  $(x, y) \in S^2$ , on a  $q_{xy} = q_{yx}$  (symétrie).
- Partant de tout état  $x$ , on sait simuler facilement selon la loi  $q_x = [q_{x1}, \dots, q_{xM}]$ .

Comme pour la méthode de rejet, la méthode de Metropolis fonctionne en plusieurs étapes.

1. Partant de  $X_n = x$ , simulons  $Y = y$  selon la loi  $q_x = [q_{x1}, \dots, q_{xM}]$ .
2. Calculons le rapport d'acceptation rejet

$$r_{xy} = \min \left\{ 1, \frac{\pi_y}{\pi_x} \right\}.$$

3. On tire une loi uniforme  $U \sim \mathcal{U}([0, 1])$  et on pose

$$\begin{cases} X_{n+1} = Y = y & \text{si } U \leq r_{xy}, \\ X_{n+1} = X_n = x & \text{sinon.} \end{cases}$$

Donc l'algorithme de Metropolis s'écrit comme suit

---

#### Algorithme Metropolis

---

##### Debut

lire  $(n)$ ; %nombre d'itérations;

lire  $(X(1))$ ; %la valeur initiale;

$i = 1$ ;

##### Tantque $i \leq n - 1$ faire

générer  $Y(i) \sim q_x$ ;

$r_{xy} = \min \left\{ 1, \frac{\pi_y(i)}{\pi_x(i)} \right\}$ ;

générer  $U \sim \mathcal{U}([0, 1])$ ;

**si**  $U \leq r_{xy}$  **alors**

$X(i + 1) = Y(i)$ ;

**sinon**

$X(i + 1) = X(i)$ ;

**fin si**;

$i = i + 1$ ;

**fin Tantque**;

afficher  $(X)$ ;

**Fin.**

Dans la construction de la chaîne  $(X_n)_{n \geq 1}$ , il faut tout conserver, y compris les fois où l'on reste sur place ( $X_{i+1} = X_i$ ).

**Proposition 6.** La suite aléatoire  $(X_n)_{n \geq 1}$  construite ici est une chaîne de Markov de matrice de transition  $P$ , définie par :

$$\begin{cases} p_{xy} = q_{xy}r_{xy} & \text{si } x \neq y, \\ p_{xx} = 1 - \sum_{y \neq x} p_{xy} & \text{sinon.} \end{cases}$$

De plus  $\pi$  est une probabilité réversible (donc invariante) pour  $P$ .

**Proposition 7.** Si la probabilité  $\pi$  est strictement positive et si la matrice de transition  $Q$  est telle que pour tout  $x \neq y$ ,  $Q \neq 0$ , alors la matrice de transition  $P$  définie dans la proposition précédente est irréductible.

**Preuve 5.** si  $x \neq y$ ,  $p_{xy} = q_{xy}r_{xy} > 0 \Rightarrow$  Tous les états communiquent.

□

**Proposition 8.** (Réversibilité de Metropolis)

Si la matrice de transition  $Q$  est irréductible, alors la chaîne  $(X_n)_{n \geq 1}$  produite par l'algorithme de Metropolis est irréductible et  $\pi$  est réversible pour cette chaîne. En particulier,  $\pi$  est son unique mesure d'équilibre.

**Preuve 6.** Soit  $P = (p_{xy})_{x,y \in S}$  la matrice de transition de la chaîne  $(X_n)_{n \geq 1}$ .

On veut montrer que

$$\pi_x p_{xy} = \pi_y p_{yx},$$

pour tout  $(x, y) \in S^2$ , nous avons  $q_{xy} = q_{yx}$  et supposons que  $\pi_y \geq \pi_x$ .

Nous avons d'une part,

$$p_{xy} = q_{xy}r_{xy} = q_{xy} \min \left\{ 1, \frac{\pi_y}{\pi_x} \right\} = q_{xy}.$$

D'autre part,

$$\begin{aligned} p_{yx} &= q_{yx}r_{yx} = q_{yx} \min \left( 1, \frac{\pi_x}{\pi_y} \right) \\ &= q_{yx} \frac{\pi_x}{\pi_y} = q_{xy} \frac{\pi_x}{\pi_y} \\ &= p_{xy} \frac{\pi_x}{\pi_y}. \\ \Rightarrow p_{yx} &= p_{xy} \frac{\pi_x}{\pi_y} \Rightarrow \pi_y p_{yx} = \pi_x p_{xy} \end{aligned}$$

□

En 1970, quelques années après les travaux de Metropolis et ses collaborateurs, Hastings s'est penché sur le sujet. Il a étendu l'application de l'algorithme de Metropolis à des cas plus généraux. Nous présentons cette généralisation dans la section suivante.

### Généralisation : Méthode de Metropolis-Hastings

Dans la version originale de l'algorithme de Metropolis, la matrice de transition  $Q$  est symétrique, c'est pour quoi Hastings a proposé de généraliser l'algorithme de Metropolis. La méthode de Hastings fonctionne presque de la même façon que l'algorithme Metropolis.

— Partant de l'état  $x$ , on sait simuler suivant la loi  $q_x = [q_{x1}, \dots, q_{xM}]$ .

- Pour tout  $(x, y) \in S^2$ , on a  $q_{xy} > 0$  implique  $q_{yx} > 0$ .
- pour tout  $(x, y) \in S^2$ , tel que  $q_{xy} > 0$ , on sait calculer  $\frac{\pi_y q_{yx}}{\pi_x q_{xy}}$ .

Donc l'algorithme de Hastings fonctionne comme suit :

1. Partant de  $X_n = x$ , simulons  $Y = y$  selon la loi instrumentale  $q_x = [q_{x1}, \dots, q_{xM}]$ .
2. Calculons le rapport d'acceptation rejet,

$$r_{xy} = \min \left\{ 1, \frac{\pi_y q_{yx}}{\pi_x q_{xy}} \right\}.$$

3. On tire une loi uniforme  $U \sim \mathcal{U}([0, 1])$  et on pose

$$\begin{cases} X_{n+1} = Y = y & \text{si } U \leq r_{xy}, \\ X_{n+1} = X_n = x & \text{sinon.} \end{cases}$$

Alors l'algorithme de Metropolis-Hastings est le suivant :

---

#### Algorithme Metropolis-Hastings

---

**Debut**

lire  $(n)$ ; %nombre d'itérations;

lire  $(X(1))$ ; %la valeur initiale;

$i = 1$ ;

**Tantque**  $i \leq n - 1$  **faire**

générer  $Y(i) \sim \mathcal{L}(q_x)$ ;

$$r_{xy} = \min \left\{ 1, \frac{\pi_y(i) q_{yx}(i)}{\pi_x(i) q_{xy}(i)} \right\};$$

générer  $U \sim \mathcal{U}([0, 1])$ ;

**si**  $U \leq r_{xy}$  **alors**

$X(i + 1) = Y(i)$ ;

**sinon**

$X(i + 1) = X(i)$ ;

**fin si**;

$i = i + 1$ ;

**fin Tantque**;

afficher  $(X)$ ;

**Fin.**

---

De même que la méthode de Metropolis, la suite aléatoire  $(X_n)_{n \geq 1}$  ainsi générée est une chaîne de Markov de matrice de transition  $P$  avec

$$\begin{cases} p_{xy} = q_{xy} r_{xy}, & \text{si } x \neq y, \\ p_{xx} = 1 - \sum_{y \neq x} p_{xy}, & \text{sinon.} \end{cases}$$

Par ailleurs, tout ce qui a été dit dans le cas d'un espace d'état discret se généralise à un espace d'état continu.

#### Cas continu

Dans le cas d'un espace d'états continu disons  $\mathbb{R}^d$ ,  $d \in \mathbb{N}^*$ , la démarche menant au choix du noyau de transition à simuler est sensiblement la même que dans le cas discret. Rappelons que la loi cible notée  $\pi$  dans le cas discret sera remplacé par  $f$  dans le cas continu et la distribution

instrumentale  $q_{xy}$  sera remplacé par  $q(y|x)$ . Le but est de simuler selon la densité  $f(x)$ . La quantité  $q(y|x)$  s'interprète comme la densité de probabilité d'aller en  $y$  sachant qu'on part d'un point  $x$ . Les hypothèses pour l'algorithme de Metropolis-Hastings dans le cas continu sont l'interprétation de celles du cas discret à savoir :

- Pour tout  $x$ , on sait simuler selon la densité  $q(\cdot|x)$ .
- Pour tout  $(x, y)$ , on a  $q(y|x) > 0$  implique  $q(x|y) > 0$ .
- Pour tout  $(x, y)$ , on sait calculer  $\frac{f(y)q(x|y)}{f(x)q(y|x)}$ .

Alors l'algorithme de Metropolis-Hastings dans le cas continu s'écrit comme suit :

---

#### Algorithme Metropolis-Hastings

---

##### Debut

lire ( $n$ ); %nombre d'itérations;

lire( $X(1)$ ); %la valeur initiale;

$i = 1$ ;

##### Tantque $i \leq n - 1$ faire

générer  $Y(i) \sim q(\cdot|X)$ ;

$r(X, Y) = \min \left\{ 1, \frac{f(Y(i))q(X(i)|Y(i))}{f(X(i))q(Y(i)|X(i))} \right\}$ ;

générer  $U \sim \mathcal{U}([0, 1])$ ;

si  $U \leq r(X, Y)$  alors

$X(i + 1) = Y(i)$ ;

sinon

$X(i + 1) = X(i)$ ;

fin si;

$i = i + 1$ ;

fin Tantque;

afficher ( $X$ );

Fin.

---

Bien que cette méthode soit beaucoup plus efficace du point de vue computationnel que la méthode d'acceptation rejet, sa vitesse de convergence vers la fonction à simuler est directement liée au choix de la fonction instrumentale  $q$ . En pratique les différents choix de  $q$  permettent de définir plusieurs variantes de l'algorithme Metropolis-Hastings.

### 3.2.2 Typologies des algorithmes de Hastings-Metropolis

Il est à noter qu'il existe plusieurs autres algorithmes qui découlent de l'algorithme de Hastings-Metropolis, dans cette section nous nous contentons par présenter les plus souvent rencontré à savoir :

- Algorithme de Metropolis-Hastings indépendant.
- Algorithme de Metropolis-Hastings à marche aléatoire.
- Algorithme de Metropolis-Hastings à une variable à la fois.

#### Algorithme de Metropolis-Hastings indépendant

L'algorithme de Metropolis-Hastings indépendant peut être présenté comme une généralisation de l'algorithme d'acceptation rejet, ce type d'algorithme repose sur l'utilisation d'une loi instrumentale  $q$  indépendant de  $X$ , cet algorithme peut s'écrire comme suit :

---

**Algorithme Metropolis-Hastings indépendant**

---

**Debut**lire ( $n$ ); %nombre d'itérations;lire ( $X(1)$ ); %la valeur initiale; $i = 1$ ;**Tantque  $i \leq n - 1$  faire**générer  $Y(i) \sim q(\cdot)$ ;

$$r(X, Y) = \min \left\{ 1, \frac{f(Y(i))q(X(i))}{f(X(i))q(Y(i))} \right\};$$

générer  $U \sim \mathcal{U}([0, 1])$ ;**si  $U \leq r(X, Y)$  alors** $X(i + 1) = Y(i)$ ;**sinon** $X(i + 1) = X(i)$ ;**fin si**; $i = i + 1$ ;**fin Tantque**;afficher ( $X$ );**Fin.**

---

Malgré que les variables  $Y_i$  soient simulées indépendamment, l'échantillon résultant n'est pas i.i.d, en particulier parce que la probabilité d'acceptation des  $Y_i$  dépend de  $X_i$ .

**Algorithme de Metropolis-Hastings à marche aléatoire**

Cette variante prend en compte la valeur précédemment simulée pour générer la valeur suivante, la loi instrumentale utilisée est une distribution qui satisfait la propriété  $q(y|x) = q(y - x)$ , c'est-à-dire,  $Y_i$  peut s'écrire sous la forme  $X_i + \varepsilon_i$ ,  $\varepsilon_i$  et générer par tirage indépendants d'une loi fixée facile à simuler (en générale uniforme ou normale), indépendante de  $X$ , ainsi la chaîne de Markov générée associée à  $q$  est une marche aléatoire.

alors l'algorithme de Metropolis-Hastings à marche aléatoire s'écrit :

---

**Algorithme Metropolis-Hastings à marche aléatoire**

---

**Debut**lire ( $n$ ); %nombre d'itérations;lire( $X(1)$ ); %la valeur initiale; $i = 1$ ;**Tantque  $i \leq n - 1$  faire**générer  $\varepsilon(i) \sim q(\cdot)$ ; $Y(i) = X(i) + \varepsilon(i)$ ;

$$r(X, Y) = \min \left\{ 1, \frac{f(Y(i))q(X(i) - Y(i))}{f(X(i))q(Y(i) - X(i))} \right\};$$

générer  $U \sim \mathcal{U}([0, 1])$ ;**si  $U \leq r(X, Y)$  alors** $X(i + 1) = Y(i)$ ;**sinon** $X(i + 1) = X(i)$ ;**fin si**; $i = i + 1$ ;**fin Tantque**;afficher ( $X$ );**Fin.**

---

## Metropolis-Hastings à sauts réversibles

Appelé aussi à une variable à la fois, quand le paramètre à simuler est de grande dimension, on est contraint de trouver une densité instrumentale multidimensionnelle engendrant une chaîne ayant le comportement d'une chaîne de Markov. Pour cela, on peut utiliser un algorithme dit à une variable à la fois.

Le principe est de diviser le vecteur  $X^{(n)}$  en plusieurs composantes  $(X_1^{(n)}, X_2^{(n)}, \dots, X_d^{(n)})$  et de les simuler une par une. À chaque itération d'algorithme, on fait évoluer  $d$  composantes  $X_i^{(n)}$  en utilisant  $d$  étapes de l'algorithme de Metropolis-Hastings, ce qui signifie que pour obtenir le nouveau vecteur  $X^{(n+1)}$  il faudra utiliser  $d$  densités instrumentales  $q_i(\cdot | x^{(n)})$ ,  $i = 1, \dots, d$ .

Pour simuler une composante  $X_i^{(n)}$  il faut utiliser la loi instrumentale  $q_i(y_i | x_{-i}^{(n)})$  et la loi cible  $f_i(x_i | x_{-i}^{(n+1)})$  où  $x_{-i}^{(n+1)} = (x_1^{(n+1)}, \dots, x_{i-1}^{(n+1)}, x_{i+1}^{(n+1)}, \dots, x_d^{(n+1)})$  (tel que  $i = 1, \dots, d$ ). Alors l'algorithme de cette méthode peut s'écrire sous la forme suivante :

Algorithme Metropolis-Hastings à sauts réversibles
MH( $f_1(x_1   x_{-1}^{n+1}), q_1$ )
MH( $f_2(x_2   x_{-2}^{n+1}), q_2$ )
$\vdots$
MH( $f_d(x_d   x_{-d}^{n+1}), q_d$ )

**Remarque 5.** Les algorithmes de Metropolis-Hastings ne génèrent pas d'échantillon indépendant et identiquement distribué en particulier parce que la probabilité d'acceptation de  $Y_i$  dépend de  $X_i$ .

**Exemple 12.** Nous présentons dans cet exemple la méthode de simulation d'une chaîne de Markov de loi stationnaire  $f$ ,

$$f(x) = \frac{1}{C} e^{-x^2} (2 + \sin(x^5) + \sin(x^2)).$$

la difficulté venant du fait que cette loi n'est connue qu'à une constante de normalisation près :

$$C = \int_{\mathbb{R}} e^{-x^2} (2 + \sin(x^5) + \sin(x^2)) dx.$$

La loi de proposition (noyau de transition  $q$ ) utilisée pour la simulation de la densité  $f$  est la loi normale centrée réduite  $\mathcal{N}(x, \sigma^2)$ . Nous avons

$$\begin{aligned} q(y|x) &= \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\left(\frac{y-x}{\sigma}\right)^2} \\ &= \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\left(\frac{x-y}{\sigma}\right)^2} \\ &= q(x|y). \end{aligned}$$

Autrement dit, le noyau de proposition  $Q$  est symétrique. Donc, nous sommes dans le cadre d'application de l'algorithme de Metropolis, cas particulier de Metropolis-Hastings.

L'algorithme suivant donne une méthode de simulation Metropolis de noyau de proposition  $Q$  et de loi-cible  $f$ . On remarque qu'il n'est pas nécessaire de connaître la valeur de la constante  $C$  pour implémenter l'algorithme.



---

**Algorithme de simulation de la loi cible  $f$** 

---

**Debut**

lire  $(n)$ ; %nombre d'itérations;

lire  $(X(1))$ ; %la valeur initiale;

$i = 1$ ;

**Tantque**  $i \leq n - 1$  **faire**

générer  $Y(i) \sim \mathcal{N}(X(i), \sigma)$ ;

$$r(X, Y) = \min \left\{ 1, e^{X^2(i) - Y^2(i)} * \left( \frac{2 + \sin(Y^5(i)) + \sin(Y^2(i))}{2 + \sin(X^5(i)) + \sin(X^2(i))} \right) \right\};$$

générer  $U \sim \mathcal{U}([0, 1])$ ;

**si**  $U \leq r(X, Y)$  **alors**

$X(i + 1) = Y(i)$ ;

**sinon**

$X(i + 1) = X(i)$ ;

**fin si**;

$i = i + 1$ ;

**fin Tantque**;

afficher  $(X)$ ;

**Fin.**

---

La figure (3.1) présente les résultats de deux simulations de la distribution cible  $f$ , la première faite avec 1000 itérations, l'autre avec 10000 itérations pour un état initial  $X_1 \sim \mathcal{U}([-2, 2])$ .

**Remarque 6.** Les algorithmes de Metropolis-Hastings peuvent-être utilisés pour simuler un vecteur aléatoire de dimension  $d$  en utilisant une densité instrumentale multidimensionnelle (Metropolis-Hastings à sauts réversibles). Mais quand  $d$  est grand, ce choix est rarement fait en pratique car la convergence d'un tel algorithme serait extrêmement lente. En effet, plus la dimension de l'espace des paramètres est grande, plus la proportion de candidats rejetés est importante. Dans ce cas il est préférable d'utiliser plutôt l'algorithme ou l'échantillonneur de Gibbs.

### 3.2.3 Echantillonneur de Gibbs

Il s'agit d'un cas particulier de Metropolis-Hastings où toutes les états sont acceptées, il requiert néanmoins beaucoup plus de connaissances sur la loi cible  $f$ .

Plaçons nous dans un espace d'état continu  $\mathbb{R}^d$ ,  $d \in \mathbb{N}^*$ , supposons que la densité cible est donnée par  $f(x) = f(x_1, x_2, \dots, x_d)$  pour tout indice  $j$  entre 1 et  $d$ , on note l'ensemble  $x_{-j} = (x_1, \dots, x_{j-1}, x_{j+1}, \dots, x_d)$ .

Ici  $x_{-j}$  correspond à  $x$  privé de sa  $j^{\text{ième}}$  élément donc la densité  $f(x)$  peut être donnée par  $f(x) = f(x_j, x_{-j})$ .

Tandis que  $f(\cdot | x_{-j})$  est la densité conditionnelle correspondant à  $(d - 1)$  coordonnées gelées et  $f(x)$  la densité marginale des ces  $(d - 1)$  coordonnées. Alors, pour tout  $j$  et pour toute suite  $x_{-j}$ , si on sait simuler suivant la densité conditionnelle  $f(\cdot | x_{-j})$ , alors :

1. Partant de  $X_n = x$ , on tire successivement des coordonnées  $j$  de 1 à  $d$ .
2. On simule  $Y$  selon la loi  $f(\cdot | x_{-j})$ .
3. On pose  $X_{n+1} = (Y, X_{-j})$ .

Donc l'algorithme de Gibbs s'écrit comme suit :

---

**Algorithme de Gibbs**

---

**Debut**

lire ( $n$ ); %nombre d'itérations;

lire( $X(1)$ ); %la valeur initiale;

$i = 1$ ;

**Tantque  $i \leq n - 1$  faire**

générer  $X_1^{i+1} \sim f_1(x_1|x_2^i, \dots, x_d^i)$ ;

générer  $X_2^{i+1} \sim f_2(x_2|x_1^{i+1}, \dots, x_d^i)$ ;

$\vdots$

générer  $X_d^{i+1} \sim f_d(x_d|x_1^{i+1}, \dots, x_{d-1}^i)$ ;

$i = i + 1$ ;

**fin Tantque;**

afficher ( $X$ );

**Fin.**

---

La méthode que nous avons présentée dans cette section est appelée échantillonnage de Gibbs par balayage déterministe (ou systématique) car l'indice  $j$  est tiré successivement à chaque étape. Le balayage aléatoire consiste à tirer au hasard uniformément une coordonnée  $j$  entre 1 et  $d$ .

**Remarque 7.** L'échantillonneur de Gibbs correspond à un algorithme de Metropolis-Hastings où toutes les transitions sont acceptées. C'est-à-dire  $r(x, y) = 1$  à chaque étape.

### 3.3 La qualité de la simulation

Dans les sections précédentes, nous avons vu que le choix de la valeur de l'état initial ainsi que le choix de la distribution de proposition sont laissés aux utilisateurs. Dans cette section nous donnons un exemple pour illustrer l'influence de ces choix sur la qualité de la simulation.

Pour cela, supposons que l'on veuille simuler la loi normale centrée réduite  $\mathcal{N}(0, 1)$ , Soit  $f(x)$  la densité cible de la loi normale centrée réduite,

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}, \quad x \in \mathbb{R}.$$

Pour ce faire, nous utilisons la loi de proposition suivante :

$$Y \sim \mathcal{L}(q(\cdot|x)) \quad \text{et} \quad Y = X + \varepsilon, \quad \varepsilon \sim \mathcal{U}([-a, a]), \quad a \in \mathbb{R}_+^*.$$

Nous avons  $Y \sim \mathcal{U}([x - a, x + a])$ .

Alors la densité de proposition est donnée par :

$$\begin{aligned} q(y|x) &= \begin{cases} 1 & \text{si } x - a \leq y \leq x + a, \\ 0 & \text{sinon.} \end{cases} \\ &= \begin{cases} 1 & \text{si } |y - x| \leq a, \\ 0 & \text{sinon.} \end{cases} \\ &= \begin{cases} 1 & \text{si } |x - y| \leq a, \\ 0 & \text{sinon.} \end{cases} \\ &= q(x|y). \end{aligned}$$

D'après la dernière relation on constate que la loi de proposition est symétrique ( $q(y|x) = q(x|y)$ ). Donc, nous sommes dans le cadre d'application de l'algorithme de Metropolis. L'algorithme qui permet de simuler la loi de densité cible  $f$  est donné comme suit :

---

**Algorithme de simulation de loi normal**

---

**Debut**

lire ( $n$ ); %nombre d'itérations;

lire( $X(1)$ ); %la valeur initiale;

$i = 1$ ;

**Tantque**  $i \leq n - 1$  **faire**

générer  $\varepsilon(i) \sim \mathcal{U}([-a, a])$ ;

$Y(i) = X(i) + \varepsilon(i)$ ;

$r(X, Y) = \min \left\{ 1, e^{\frac{X^2(i) - Y^2(i)}{2}} \right\}$ ;

générer  $U \sim \mathcal{U}([0, 1])$ ;

**si**  $U \leq r(X, Y)$  **alors**

$X(i + 1) = Y(i)$ ;

**sinon**

$X(i + 1) = X(i)$ ;

**fin si**;

$i = i + 1$ ;

**fin Tantque**;

afficher ( $X$ );

**Fin.**

---

### 3.3.1 Choix de la valeur de l'état initial

La figure (3.2) permet de visualiser l'évolution des séries chronologiques, ainsi que les distributions et les histogrammes de la chaîne obtenue à partir de la simulation de la loi normale centrée réduite  $\mathcal{N}(0, 1)$  pour 1000 itérations en utilisant l'algorithme ci-dessus pour des valeurs initiales  $X_1 = 10$  et  $X_1 \sim \mathcal{U}[-a, a]$ ,  $a = 3$  respectivement.

Les graphiques de la figure (3.2) illustrent bien les résultats de simulations pour  $X_1 = 10$  et  $X_1 \sim \mathcal{U}([-3, 3])$ , nous voyons que pour une valeur de l'état initial  $X_1 = 10$  (graphe de gauche) la chaîne simulée prend plus de temps pour oublier la valeur de l'état initial. Autrement dit, plus de temps pour se rapprocher du ventre de la loi normale centrée réduite, par ailleurs pour  $X_1 \sim \mathcal{U}([-3, 3])$  (graphe de droite) la chaîne converge rapidement vers la loi cible et donne des meilleurs résultats par rapport au premier choix.

### 3.3.2 Choix de la loi de proposition

La figure (3.3) décrit deux échantillons de 1000 itérations produit par l'algorithme de Metropolis-Hastings pour une valeur de l'état initial  $X_1 \sim \mathcal{U}([-3, 3])$  et les lois de propositions  $\mathcal{U}([-30, 30])$  et  $\mathcal{U}([-0.1, 0.1])$  respectivement.

La figure (3.3) illustre les résultats des simulations pour une valeur initiale  $X_1 \sim \mathcal{U}([-3, 3])$  fixe et des lois de propositions différentes. Lorsque nous choisissons la loi uniforme sur l'intervalle  $[-30, 30]$  comme loi de proposition le taux d'acceptations tombe à 5.1%, les pas de la marche aléatoire sont trop grands, pour une loi normale centrée en 0. En effet, dans l'intervalle  $[-3, 3]$  la plus parts des propositions tombent dans la queue de la loi normale. Dans ce cas, une chaîne de  $n = 1000$  approche mal la densité cible. À l'opposé avec une loi uniforme sur l'intervalle  $[-0.1, 0.1]$ , le taux d'acceptations est de l'ordre de 96.8% mais la convergence est trop lente. La marche aléatoire fait des petits pas, donc met trop de temps à explorer la région d'intérêt, ici l'intervalle  $[-3, 3]$ . La chaîne générée à partir de cette loi pour  $n = 1000$  approche mal la densité cible.

Nous constatons d'après ces résultats que le choix de la valeur de l'état initial ainsi que le choix de la distribution instrumentale influencent sur la qualité de la simulation de la loi cible. En effet, un

choix judicieux favorisera une exploration rapide de l'espace des états et accélérera en conséquence la convergence de la chaîne vers la distribution cible.

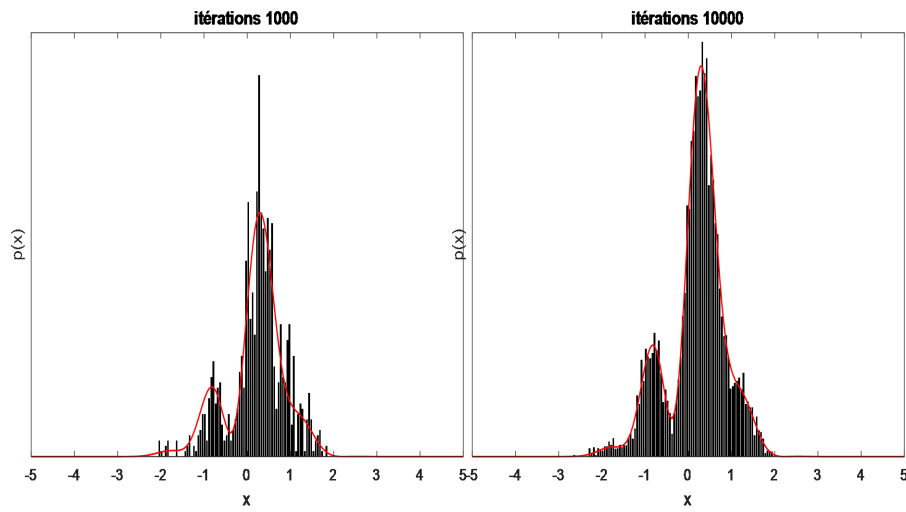


FIGURE 3.1 – Simulations de la loi cible  $f$  avec respectivement 1000 itérations et 10000 itérations.

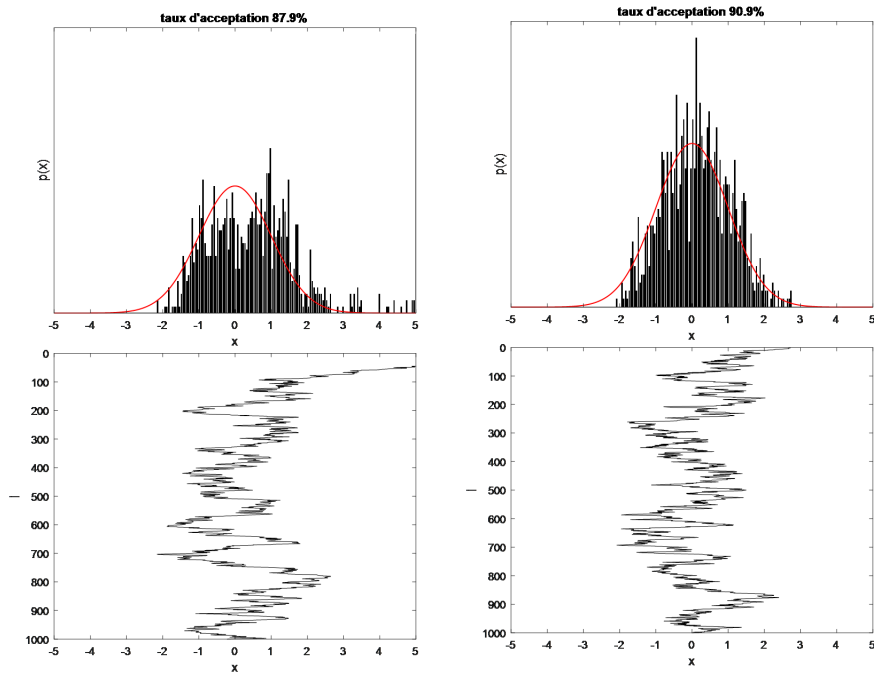


FIGURE 3.2 – Distributions et histogrammes des chaînes simulées suivant  $\mathcal{N}(0, 1)$  et les séries chronologiques des deux simulations.

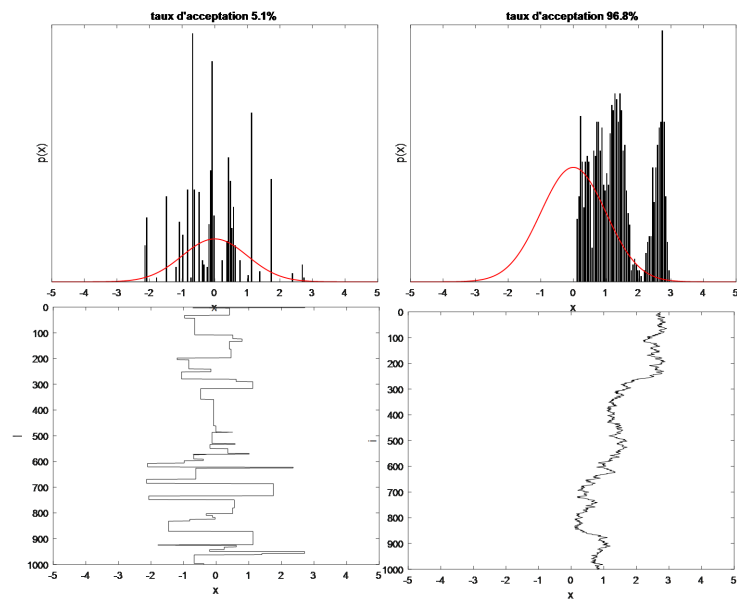


FIGURE 3.3 – Distributions et Histogrammes des chaînes simulées suivant  $\mathcal{N}(0,1)$  et les séries chronologiques des deux simulations.