République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université A.Mira de Béjaïa
Faculté des Sciences Exactes
Département de Mathématiques

Mémoire

En vue de l'Obtention du Diplôme de

Master en Mathématiques

Option : Probabilités Statistiques et Applications

Thème

Méthodes de Monte-Carlo par chaînes de Markov et leurs applications

Présenté par :
SALHI Zina

Devant le jury composé de :

Mme. *L. Bouraine*	*Présidente*	*U. A. Mira. Béjaïa*
Mlle. *L. Baiche*	*Examinatrice*	*U. A. Mira. Béjaïa*
Mr. *S. Ouazine*	*Examinateur*	*U. A. Mira. Béjaïa*
Mr. *Y. Boumzaid*	*Promoteur*	*U. A. Mira. Béjaïa*

Béjaïa, 24 juin 2018

Remerciements

Au premier lieu et avant tout je tiens à remercier Dieu qui m'a donné la force et la patience d'accomplir ce modeste travail.

Je tiens à exprimer aussi toute ma gratitude à mon encadreur M^r Y. Boumzaid qui m'a offert la possibilité de réaliser ce travail sous sa direction, je le remercie également pour ses discussions profitables, ses encouragements, son entière disponibilité, ses conseils judicieux, sa compréhension et ses suggestions, je lui suis reconnaissante pour la liberté et la confiance qu'il ma témoigné tout au long de ce travail.

J'adresse mes profonds remerciements aux membres de jury qui ont bien voulu évaluer et examiner mon travail :
M^{me} L. Bouraine qui ma fait l'honneur de présider mon jury de soutenance.
M^r S. Ouazine et M^{lle} L. Baiche qui m'ont fait l'honneur d'examiner mon travail.

Merci également à ma famille et en particulier mes parents, pour l'encouragement constant et leurs aides moraux sans retenue, et pour leurs soutiens inconditionnels tout au long de ces années d'études.
Enfin, mes remerciement vont à tous ceux et celles qui m'ont soutenus ou qui, d'une manière ou d'une autre, ont contribués à la réalisation de ce travail.

Z. Salhi

Résumé

Les méthodes de Monte Carlo notées «MC» et les méthodes de Monte Carlo par chaînes de Markov notées «MCMC» sont aujourd'hui utilisées pour simuler des phénomènes physiques complexes dans plusieurs domaines scientifiques et appliqués : radioactivité, physique des hautes énergies, réseaux, économétrie, logistique et bien d'autre domaines. Ce sont les deux aspects abordés dans ce mémoire.

Les travaux présentés dans ce document fournissent une revue simple, compréhensive et tutoriel de ces méthodes. Les travaux présentés dans le cadre des méthodes de Monte-Carlo décrivent une application classique de ces méthodes à savoir l'intégration Monte-Carlo. Tandis que les travaux présentés dans le cadre des méthodes de MCMC, décrivent deux techniques conçues pour créer des chaînes de Markov de loi stationnaire donnée, à savoir les algorithmes de Metropolis-Hastings et l'échantillonnage de Gibbs.

Mots clés : Simulation Monte-Carlo, Estimation Monte-Carlo, Intégration Monte-Carlo, Méthodes Monte-Carlo par chaînes de Markov, Metropolis-Hastings, Échantillonnage de Gibbs.

Abstract

Monte Carlo «MC» methods and Markov chain Monte Carlo methods «MCMC» are now used to simulate complex physical phenomena in several scientific and applied fields : radioactivity, high energy physics, networks, econometrics, logistics and many other fields. These are the two aspects addressed in this Master's thesis.

The work presented in this paper provides a simple, comprehensive and tutorial review of these methods. The work presented in the context of Monte-Carlo methods describes a classical application of these methods, namely Monte-Carlo integration. In other hard the work presented in the context of the MCMC methods, describe two techniques designed to create given stationary law Markov chains, namely the Metropolis-Hastings algorithms and the Gibbs sampling.

Keywords : Monte Carlo simulation, Monte Carlo estimate, Monte Carlo integration, Markov chain Monte Carlo, Metropolis-Hastings, Gibbs Sampling.

Table des matières

Notations

$X,\ Y$	Variables aléatoires.
U	Variable aléatoire distribuée selon $\mathcal{U}([0,1])$.
u	Réalisation de la variable aléatoire U.
F	Fonction de répartition de la densité f.
f	Densité de probabilité.
$\mathcal{U}([0,1])$	Distribution uniforme sur l'intervalle $[0,1]$.
$\mathcal{U}([a,b])$	Distribution uniforme sur l'intervalle $[a,b]$.
$\mathcal{N}(\mu,\sigma^2)$	Distribution gaussienne de moyenne μ et variance σ^2.
$\mathcal{N}(0,1)$	Distribution gaussienne centrée réduite.
$\varepsilon(\lambda)$	Distribution exponentielle de paramètre λ.
$\mathcal{L}(.)$	Loi d'une variable aléatoire.
$\mathbb{1}_A$	Fonction indicatrice qui vaut 1 sur l'ensemble A et 0 ailleurs.
P	Probabilité.
n	Nombre des échantillons.
n_0	Période de chauffe.
i	Indice des itérations d'un algorithme.
$i,\ j$	Indices généraux.
$x,\ y$	Etats d'une chaîne de Markov.
S	Espace des états d'une chaîne de Markov.
P	Matrice de transition.
q	Densité de proposition (instrumentale).
Q	Noyau de transition de la loi de proposition q.
π	Densité cible.
D	Dilution d'échantillon.
$\mathbb{R}$	Ensemble des nombres réels.
$\mathbb{N}$	Ensemble des entiers naturels.
$\mathbb{N}^*$	Ensemble des entiers naturels non nuls.
$\mathbb{R}^d$	Ensemble des nombres réels de dimension d.
log	Logarithme népérien.
$\sim$	Suivre la loi, de loi....
v.a	Variable aléatoire.
i.i.d	Indépendante identiquement distribuée.
p.s	Presque sûrement.
TCL	Théorème centrale limite.
I.C	Intervalle de confiance.
Var	Variance.
G.N.P.A	Générateur de nombre pseudo-aléatoire.
G.N.A	Générateur de nombre aléatoire.
MC	Monte-Carlo.
MCMC	Monte-Carlo Chaînes de Markov.

Introduction générale

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 » et les méthodes de Monte-Carlo par chaînes de Markov notées « MCMC » sont aujourd'hui utilisées pour simuler des phénomènes physiques complexes dans plusieurs domaines scientifiques et appliqués : radioactivité, physique des hautes énergies, réseaux, économétrie et logistique.
La technique de simulation Monte-Carlo s'appuie sur l'échantillonnage des distributions des quantités incertaines.

Dans la littérature, il n'existe pas de définition formelle des méthodes de Monte-Carlo. En effet, d'une référence à l'autre, ce terme est employé autant pour désigner les méthodes utilisant la simulation des phénomènes aléatoires pour résoudre des problèmes mathématiques, comme des problèmes d'optimisations ou d'intégrations. Ces méthodes utilisent l'échantillonnage répété de variables aléatoires pour déterminer le comportement d'une distribution.
L'idée de base est d'obtenir un échantillon aléatoire de la distribution d'intérêt et d'estimer par la suite les quantités voulues de façon empirique en se servant de l'échantillon généré.
Cette approche repose sur deux conditions qui sont la capacité de générer des valeurs de la distribution en question et la capacité de produire un gros échantillon afin d'obtenir des résultats fiables, mais il y a des cas où la loi d'intérêt n'est pas facilement simulable, la méthode de rejet peut aussi conduire à des simulations lentes si la probabilité de rejet est grande. Dans ce cas, les méthodes de Monte-Carlo par chaînes de Markov vont permettre de pallier à ces problèmes.

Les méthodes de Monte-Carlo par chaînes de Markov MCMC constituent une des approches les plus utilisées par la communauté statistique. Ces méthodes emploient une chaîne de Markov auxiliaire dont la distribution stationnaire est la distribution d'intérêt. L'algorithme de Metropolis-Hastings, introduit par Metropolis et al 1953 [14] et généralisé par Hastings 1970 [13] , est considéré comme l'une des premières méthodes MCMC. Depuis le nombre de tels algorithmes ainsi que le nombre des publications, parlant sur leurs convergence et leurs application à augmenter de façon remarquable.
Le principal attrait des approches MCMC et leurs facilité d'application à des distributions d'intérêts complexes et/ou en grandes dimensions. En plus, un grand avantage de l'application de ces méthodes réside dans le fait que la constante de normalisation de la densité d'intérêt ne doit généralement pas être spécifiée.

Dans ce mémoire, nous faisons une distinction entre les techniques de simulation permettant de générer des suites de variables aléatoires distribuées selon une densité donnée et les méthodes de Monte-Carlo qui utilisent ces suites. De plus, l'étude des méthodes Monte-Carlo est faite sous l'angle de l'utilisation des méthodes de simulations pour estimer des moments d'une distribution donnée. Ce choix se justifie par le fait que la méthodologie utilisée pour

résoudre ce type de problèmes peut-être employée pour résoudre plusieurs d'autres types de problématiques, comme l'estimation des quantités données sous forme d'intégrales ou encore la résolution des problèmes d'optimisations .
Chaque méthode présentée dans ce document utilise une technique de simulation particulière pour créer un échantillon, puis estime le moment à l'aide d'une méthode empirique. Donc, chacun des chapitres constituant ce document présente des techniques de simulation spécifique au contexte à étudier et discute des performances des méthodes de Monte-Carlo qui leur sont associées.

Ce mémoire est divisé en quatre chapitres :

Le premier chapitre, introduit les méthodes de simulation. Les méthodes présentées dans ce chapitre sont des méthodes de simulations simples à implémenter, mais inutilisables lorsque la suite des variables aléatoires générées par la distribution d'intérêt n'est pas indépendante identiquement distribuée.

Le deuxième chapitre, donne une application classique des méthodes de Monte-Carlo à savoir l'intégration Monte-Carlo. Dans un premier temps nous donnons les techniques de Monte-Carlo classiques utilisées pour l'estimation des quantités données sous forme d'intégration. Par la suite nous abordons quelques méthodes de réduction de variance permettant de réduire l'erreur de l'estimation.

Le troisième chapitre, introduit une classe de méthodes de Monte-Carlo utilisant les chaînes de Markov. Dans un premier temps, nous donnons quelques notions sur les chaînes de Markov qui seront nécessaires pour l'étude et la construction des algorithmes de Monte-Carlo par chaînes de Markov. Par la suite, nous donnons quelques algorithmes de Monte-Carlo utilisant ces chaînes afin de contourner les problèmes inhérents aux algorithmes de simulations et aux méthodes de Monte-Carlo classiques présentées dans les deux premiers chapitres.

Le quatrième chapitre en lui même est composé de deux parties. La première partie de ce chapitre énumère, via un exemple concret, les différentes variantes de l'implémentation des méthodes de Monte-Carlo par chaînes de Markov et comment celles-ci influencent sur la vitesse d'exécution de l'algorithme d'un coté et sur la qualité de la chaîne de Markov générée de l'autre. La deuxième partie est consacrée à une application réelle, à savoir la résolution d'un problème d'optimisation et dans lequel nous appliquons les méthodes MCMC.

Ce travail se termine par une conclusion générale qui synthétise les points essentiels abordés dans ce mémoire, ouvre une discussion et donne des perspectives à notre travail.

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 une 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. *[6]*
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. *[6]*
Un générateur algorithmique de nombres pseudo-aléatoires (G.N.P.A) 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 utilisable.

Dans la suite de ce document, un générateur de nombres uniformes sera un algorithme qui permet de produire une suite de nombres aléatoires issus 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 meilleure 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 congruences

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 :

$$\begin{cases} X_1 = & k \\ X_{i+1} = & (aX_i + b) \ mod \ m, \ i = 1, 2, \ldots, n-1, \end{cases} \tag{1.1}$$

où $a,\ b,\ m$ et X_1 sont des entiers positifs donnés. On dit que a est le multiplicateur, b est l'incrément, m le modulo, n le nombre des échantillons à générer et X_1 la valeur initiale appellée aussi graine.

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

$$U_i = X_i/m. \tag{1.2}$$

La longueur d'une suite quelconque générée par une relation récursive du type $(aX_i + b)\ mod\ m$ 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))[6]*
Soit $k \in \{0,\ 1, ...,\ m-1\}$. 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 :

$$\begin{cases} X_1 = & k \\ X_{i+1} = & (aX_i + b) \ mod \ m, \ i = 1, 2, \ldots n-1, \end{cases}$$

a un cycle de longueur m.

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 sous-section précédente. Dans cette sous-section, nous donnons les paramètres de quelques générateurs utilisés en informatique (voir table (1.1)).

Longage/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 informatiques.

Exemple 2. *L'algorithme ci-dessus, génère une suite de variables aléatoires produite par le générateur Scilab.*

Algorithme de génération des nombres aléatoires.

Debut
lire(n) ;
$a = 843314861$ *;*
$b = 453816693$ *;*
$m = 2^{31}$ *;*
$X(1) = 0$ *;* *%la graine*
$U(1) = \frac{X(1)}{m}$;
pour $i = 1 : n-1$ ***faire***
$X(i+1) = (aX(i)+b) mod\ m$ *;*
$U(i+1) = \frac{X(i+1)}{m}$ *;*
fin pour*;*
afficher (U) ;
Fin*.*

La figure (1.1), donne le nuage de points de 1000 *tirages de variables aléatoires de loi* $\mathcal{U}([0,1])$ *obtenues par le générateur Scilab.*

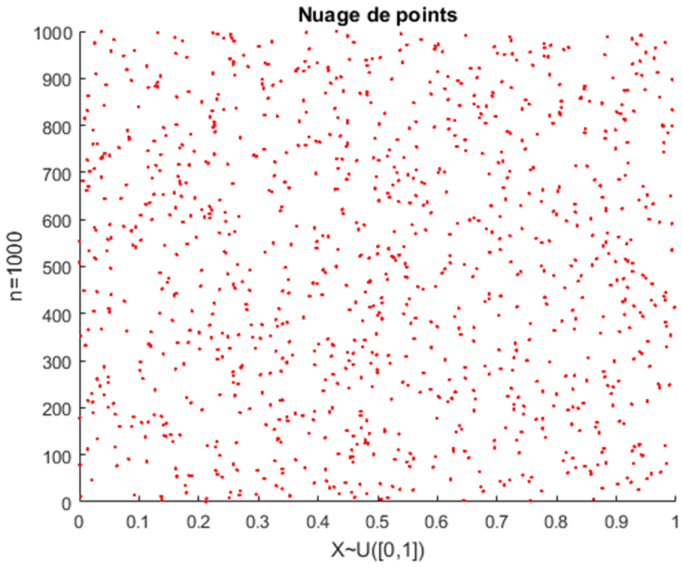

FIGURE 1.1 – Génération des nombres aléatoires.

Rappelons, qu'il existe beaucoup de tests permettant de s'assurer de la qualité de nombres aléatoires produits par un générateur algorithmique. Ces derniers permettent de vérifier la stochasticité et l'uniformité des séquences $(U_1, .., U_n)$ générées. Parmis 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 $(U_{i-1}, .., U_{i-n})$.

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 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-uniformes. Dans cette section, nous nous contentons par la présentation des trois méthodes principales à savoir :

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

1.3.1 Méthode 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 théorèmes 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 $F^{-1}(U)$ a 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 est continue, F est croissante et F^{-1} existe. On pose $X = F^{-1}(U)$. Alors pour $u \in [0,1]$, on a $U \sim \mathcal{U}([0,1])$ tout $x \in \mathbb{R}$,*

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

□

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 $U(i) \sim \mathcal{U}([0,1])$;
  $X(i) = F^{-1}(U(i))$;
fin pour ;
afficher (X) ;
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 généralisé)*
Soit X une v.a de fonction de répartition F, posons pour $0 \leq u \leq 1$,

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

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

Preuve 2. *Pour une preuve voir [15]*

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

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

C'est-à-dire,

$$X = \sum_{k=1}^{n} k.\mathbb{1}_{P_{k-1} \leq u < P_k},$$

avec,

$$\begin{cases} P_0 = & 0 \\ P_k = & \sum_{i \leq k} p_k. \end{cases}$$

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 un nombre aléatoire $u \in]0,1[$. D'après le théorème 3, la fonction pseudo-inverse est donnée par :

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

Soit

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

Donc l'échantillon de X à 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 :*

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

Et sa fonction de répartition est donnée par :

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

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

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

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

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

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

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

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(λ) ; % donner la valeur de λ
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) ;
Fin.

La figure ci-dessus, représente l'histogramme et la distribution d'un échantillon de taille $n = 1000$ généré à partir de la loi exponentielle de paramètre $\lambda = \frac{1}{2}$.

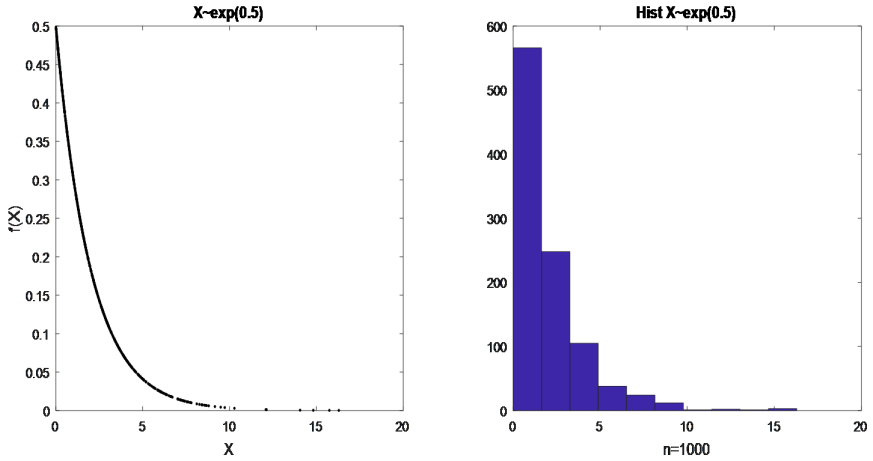

FIGURE 1.2 – Simulation de la loi exponentielle.

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 :*

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

Et sa fonction de répartition est donnée par :

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

Posons pour tout $0 \leq u \leq 1$, $u = F(x)$,
on trouve la fonction pseudo inverse suivante :

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

On peut mettre en place l'algorithme suivant :

Algorithme simulation de la loi Laplace.

Debut
lire(n) ; %nombre d'échantillon
lire(λ) ; % donner la valeur de λ
pour $i = 1 : n$ **faire**
  générer $U(i) \sim \mathcal{U}([0, 1])$;
  si $U(i) \leq \frac{1}{2}$;
  $X(i) = \frac{1}{\lambda} \log(2U(i))$;
  sinon
  $X(i) = -\frac{1}{\lambda} \log(2(1 - U(i)))$;
  fin si ;
fin pour ;
afficher (X) ;
Fin.

La figure ci-dessus, représente l'histogramme et la distribution d'un échantillon de taille $n = 1000$ généré à partir de la loi de Laplace de paramètre $(0, \frac{1}{\lambda})$, $\lambda = 0.5$.

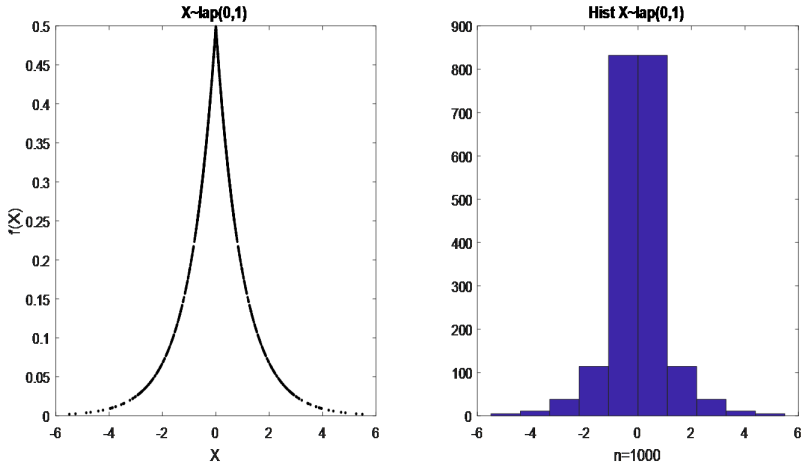

FIGURE 1.3 – Simulation de la loi de Laplace.

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. Ceci n'est pas toujours envisageable, il suffit de penser à la loi normale centrée réduite de densité,

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

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) $(Y_n)_{n\geq 1}$ i.i.d de densité g.

(b) $(U_n)_{n\geq 1}$ i.i.d 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 le pile ou face, à savoir pour tout $y \in \mathbb{R}^d$:

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

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 $\frac{1}{m}$.*

Preuve 3. *Pour une preuve voir [15]*

Remarque 1. *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. Dés 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 (grossièrement proche de f) et facile à simuler.

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 ;
  $r = \frac{f(Y(i))}{mg(Y(i))}$; % calculer le rapport d'acc-rejet ;
  si $U(i) \leq r$
    $X(j) = Y(i)$;
    $j = j + 1$;
  fin si ;
  $i = i + 1$;
fin Tantque ;
afficher (X) ;
Fin.

Afin de mieux expliquer 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 $\mathcal{N}(0,1)$,

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

Soit g la densité de la loi de Laplace de paramètre $(0, \frac{1}{\lambda})$,

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

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	λ	$+\infty$
$h'(x)$	$+$	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 $\forall\lambda \geq 0$,

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

D'où le tableau de variation de $m(\lambda)$ *donné dans la table (1.3).*
D'après ce tableau, la valeur de λ *qui permet de minimiser la probabilité d'acceptation-rejet est :* $\lambda = 1$.

λ	0	1	$+\infty$
$m^{'}(\lambda)$	$-$	0	$+$
$m(\lambda)$	$\searrow$	$m(1)$	$\nearrow$

TABLE 1.3 – Tableau de variation de m.

Donc pour $\lambda = 1$, *on touve* $m = \frac{2}{\sqrt{2\pi}}$ *et* $g(x) = e^{-|x|}$.
D'après les résultats précédents, on peut donc simuler f *avec un algorithme de rejet (puisqu'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 simulation de la loi normale par méthode de rejet.

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

La figure ci-dessus représente la simulation de la loi normale à partir de la loi de Laplace.

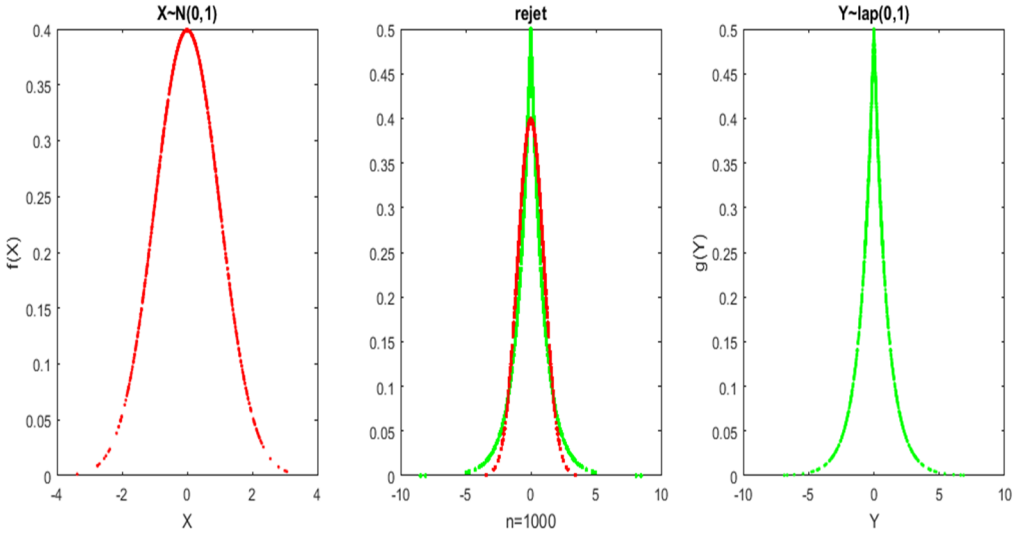

FIGURE 1.4 – Simulation d'une loi normale à partir de la loi de Laplace. À gauche, la simulation de la loi normale, au milieu la simulation de la loi normale par la méthode de rejet et à droite la simulation de la loi de Laplace.

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 connait la densité jointe de X et Y :

$$f(x,y) = \frac{1}{2\pi} e^{-\frac{x^2+y^2}{2}}, \ (X,Y) \in \mathbb{R}^2.$$

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 indépendentes et de même loi $\mathcal{N}(0,1)$.

Preuve 4. *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 :

$$\begin{cases} x = r\cos(\theta) \\ y = r\sin(\theta) \end{cases}$$

On a donc :

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

A partir de la dernière équation nous avons $\frac{1}{2\pi}$ représente la densité de la loi $\Theta \sim \mathcal{U}([0,2\pi])$. Par l'application de la méthode d'inversion vue précédemment on aura :

$$\Theta = 2\pi V,\ V \sim \mathcal{U}([0,1]).$$

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

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

Si on pose $R' = R^2$ alors on reconnais 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 4 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'ou la relation suivante :

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

□

Algorithme de Box-Müller.

Debut
lire(n) ;
pour $i = 1 : n$ **faire**
  générer $U(i) \sim \mathcal{U}([0,1])$;
  générer $V(i) \sim \mathcal{U}([0,1])$;
  $X(i) = \sqrt{-2\log(U(i))}\cos(2\pi V(i))$;
  $Y(i) = \sqrt{-2\log(U(i))}\sin(2\pi V(i))$;
fin pour ;
Afficher $(X,\ Y)$;
Fin.

Remarque 2. *1. Pour simuler une v.a $X \sim \mathcal{N}(0,1)$, il suffit de :*

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

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

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

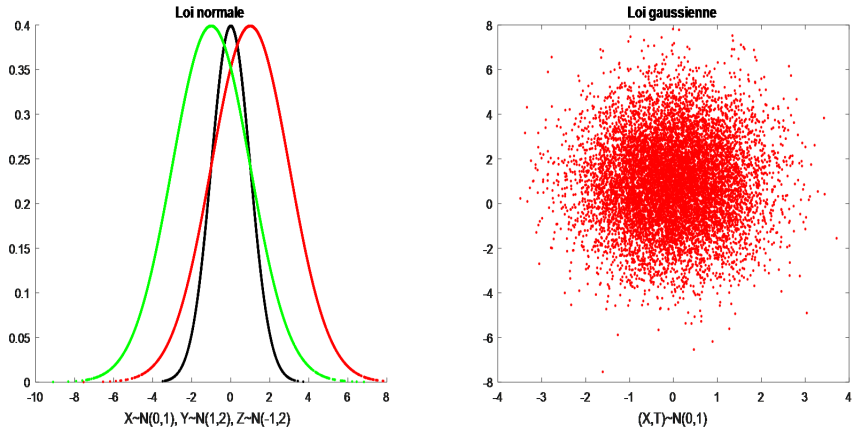

FIGURE 1.5 – À gauche la simulation de loi normale avec différentes paramètres et à droite la simulation d'un couple de v.a gaussien.

1.4 Conclusion

Dans ce chapitre, nous avons présenté quelques méthodes de simulation permettant de générer un échantillon de variables aléatoires distribuées selon une densité donnée. Ces méthodes vont être utilisées dans le chapitre suivant afin de résoudre des problèmes mathématiques concrets.

Chapitre 2

Intégration Monte-Carlo et erreur de l'estimation

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. 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 pécision. Rappelons ici que les méthodes présentées dans ce chapitre sont utilisables lorsque la suite de variables aléatoires de la distribution dont nous voulons déterminer la valeur des moments est 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 de type :

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

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 de Monte-Carlo de base est défini par :

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

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

Théorème 4. *Soit $(X_n)_{n\geq 1}$ une suite de v.a.i.i.d à valeur dans $\mathbb{R}^d$, $d \in \mathbb{N}^*$, telle que $E[X] = \mu < +\infty$.*
Pour tout $\varepsilon > 0$

$$\lim_{n \to +\infty} P\left\{ \left| \frac{1}{n} \sum_{i=1}^{n} \varphi(X_i) - \mu \right| \geq \varepsilon \right\} = 0.$$

Proposition 3. *La convergence en loi d'une suite de v.a* $X_n \xrightarrow[n\to+\infty]{loi} X$ *(*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 (intégrable* $E[\varphi(X)] < +\infty$*).*

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

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

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

avec f *une densité de probabilité.*

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 φ soit intégrable).

2.1.2 Loi forte des grands nombres

Théorème 5. *Soit* $(X_n)_{n\geq 1}$ *une suite de v.a.i.i.d à valeur dans* $\mathbb{R}^d,\ d \in \mathbb{N}^*,\ X_1 \sim \mathcal{L}(f)$*. On suppose que* $\varphi(x)$ *est une fonction intégrable, (*$E[\varphi(X)] < +\infty$*), alors :*

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

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

Sous les hypotèses 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 d'estimation.

Debut
lire(n) ;
$s = 0$; %initialiser une somme ;
pour $i = 1 : n$ **faire**
  générer $X(i) \sim \mathcal{L}(f)$;
  $s = s + \varphi(X(i))$; % calculer la somme ;
fin pour ;
$\hat{I}_n = \frac{s}{n}$;
afficher $(\hat{I}_n)$;
Fin.

2.1.3 Exemples d'application

Exemple 7. ***Cas Classique***

Estimation de la quantité $I = \int_a^b \varphi(x)dx,\ a, b \in \mathbb{R},\ b > a.$

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

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

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

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

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 d'estimation.
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 ;
$\hat{I}_n = \frac{s}{n}(b-a)$;
afficher $(\hat{I}_n)$;
Fin.

Exemple 8. ***Estimation de la valeur de*** π
Soit une suite de couple $(X_n, Y_n)_{n>1}$ *de v.a.i.i.d,* $(X_1, Y_1) \sim \mathcal{U}([-1, 1])$.
Posons $\varphi(x, y) = \mathbb{1}_{x^2+y^2\leq 1}(x, y)$.
L'aire du cercle d'unité est donnée par :

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

D'après la loi des grands nombres on aura,

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

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 π*, 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 π *est donné comme suit :*

Algorithme d'estimation de la valeur de π***.***

Debut
lire (n) ;
$s = 0$ *;*
pour $i = 1 : n$ ***faire***
générer $U(i) \sim \mathcal{U}([0,1])$ *;*
générer $V(i) \sim \mathcal{U}([0,1])$ *;*
$X(i) = 2U(i) - 1$ *;*
$Y(i) = 2V(i) - 1$ *;*
si $(X(i)^2 + Y(i)^2 \leq 1)$ *;*
$s = s + 1$ *;*
fin si*;*
fin pour*;*
$\pi = 4 * \frac{s}{n}$ *;*
Afficher (π) *;*
Fin*.*

Autrement dit, afin d'estimer la valeur de π*, 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'interieur du cercle est donnée par :*

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

D'où

$$P = \frac{\pi}{4} \Longrightarrow \pi = 4 * P.$$

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* π *et le nombre des points qui tombent à l'intérieur du cercle d'unité.*

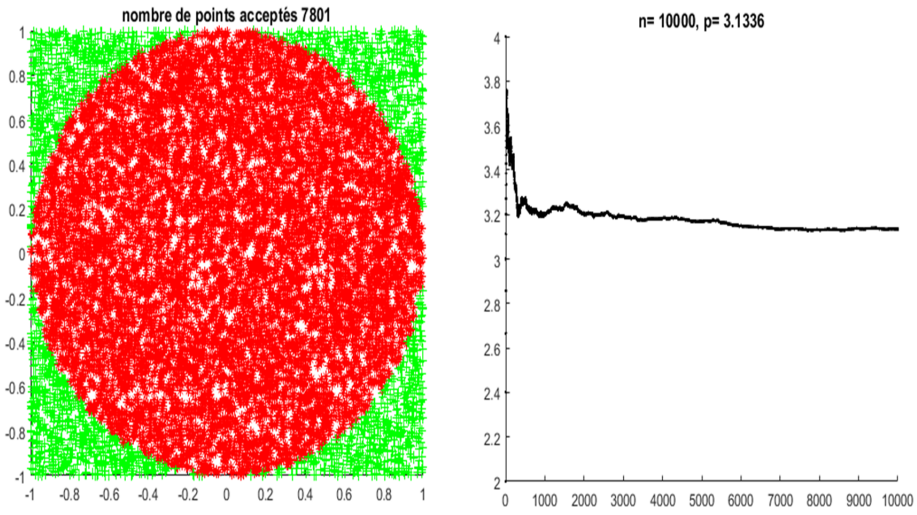

FIGURE 2.1 – Estimation de la valeur de π.

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.

2.2 Erreur de l'estimation Monte-Carlo

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 6. *TCL*
Soit $\varphi(X_i),\ i = 1...n$, *une suite de v.a.i.i.d à valeur dans* $\mathbb{R}$.
On suppose que $E[\varphi(X)^2] \leq +\infty$.

Soit $var[\varphi(X)] = \sigma^2$, *alors*

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

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

Sous les hypothèses du théorème centrale limite, on cherche à approcher $E[\varphi(X)]$ à β 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

$$P(|\varepsilon_n| \leq \beta) \geq (1-\alpha). \tag{2.3}$$

Nous avons

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

D'après TCL : $\frac{\sqrt{n}}{\sigma}\left(\frac{1}{n}\sum_{i=1}^{n}\varphi(X_i) - E[\varphi(X)]\right) \xrightarrow[n\to+\infty]{loi} \mathcal{N}(0,1)$.
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 ϕ^{-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 tirages nécessaires n pour atteindre un certain niveau d'erreur avec une certaine confiance est une fonction linéaire en fonction σ^2.

Proposition 4. *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 σ^2 pourrait ne pas être connue mais on peut l'estimer par une méthode de Monte-Carlo grâce au théorème suivant.

Théorème 7. *(Estimation de la variance)*
Si $\varphi(X_1)...\varphi(X_n)$ *Sont i.i.d avec* $E[\varphi(X)^2] < +\infty$ *et* $var[\varphi(X_1)] = \sigma^2$ *alors,*

$$\frac{1}{2n}\left[(\varphi(X_1) - \varphi(X_2))^2 + ... + (\varphi(X_{2n-1}) - \varphi(X_{2n}))^2\right] \underset{n\to+\infty}{\overset{P.S}{\longrightarrow}} \sigma^2.$$

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* π *ainsi que la construction de l'intervalle de confiance à* $(1-\alpha) = 95\%$.
Dans l'exemple de l'estimation de la valeur de π*, 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 d'estimation de la variance de π***.***

Debut
lire(n) ;
$var = 0$ *;*
pour $i = 1 : n$ ***faire***
  générer $U(i) \sim \mathcal{U}[0,1]$ *;*
  générer $V(i) \sim \mathcal{U}[0,1]$ *;*
  générer $T(i) \sim \mathcal{U}[0,1]$ *;*
  générer $W(i) \sim \mathcal{U}[0,1]$ *;*
  $X(i) = 2U(i) - 1$ *;*
  $Y(i) = 2V(i) - 1$ *;*
  $Z(i) = 2T(i) - 1$ *;*
  $R(i) = 2W(i) - 1$ *;*
  si $(X(i)^2 + Y(i)^2 \leq 1)$ ***et*** $(Z(i)^2 + R(i)^2 \leq 1)$ ***alors***
    $var = var + \left[(X(i)^2 + Y(i)^2) - (Z(i)^2 + R(i)^2)\right]^2$ *;*
  fin si*;*
fin pour*;*
$\hat{\sigma}_n^2 = var/(2 * n)$ *;*
afficher$(\hat{\sigma}_n^2)$ *;*
Fin*.*

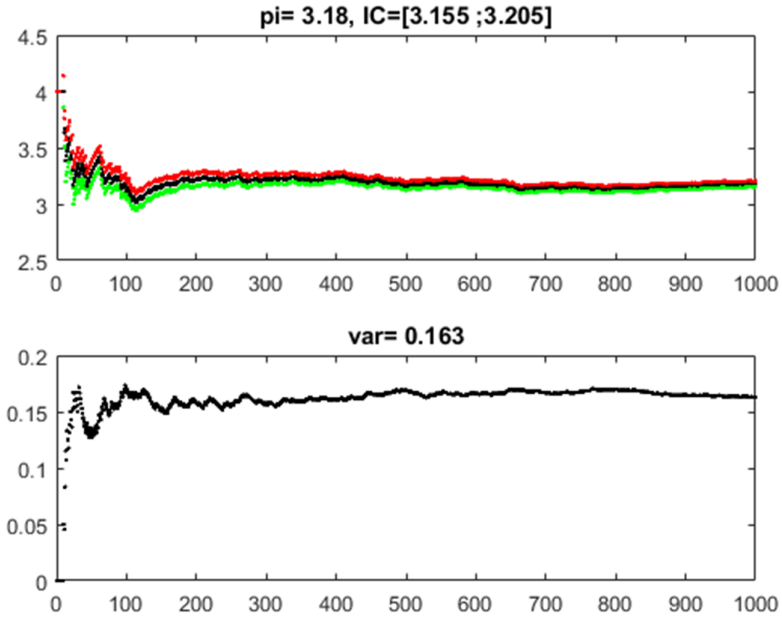

FIGURE 2.2 – Estimation de la valeur π, la variance de cette méthode et l'intervalle de confiance.

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 ε_n d'ordre $\frac{\sigma^2}{n} = var[\varphi(X)]/n$. 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 nécéssaires 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 $X_1, ..., X_n,\ X_1 \sim \mathcal{N}(0,1)$.
2. On montre que $E[\varphi(X)] < +\infty$.

Nous avons

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

D'après les étapes (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 β près avec une confiance de $(1-\alpha)$, nous aurons besoin de générer un nombre des échantillons qui dépend de la variance, i.e.

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

Calculons la variance de cette méthode.

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

avec $(E[\varphi(X)])^2 = e^{\gamma^2}$.
De la même façon, 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 γ.

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

TABLE 2.1 – Valeur de n en fonction de γ.

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 deux ou trois 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 de deux méthodes pour estimer $I = E[\varphi(X)]$.

Méthode de Monte Carlo Classique

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

Méthode d'échantillonnage préférentiel

$I \approx \hat{I}_n^2 = \frac{1}{n}\sum_{i=1}^{n} \frac{\varphi(Y_i)f(Y_i)}{h(Y_i)} \quad Y_i,\ i = 1 \dots n$ 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 $h : y \longmapsto \frac{\varphi(y)f(y)}{E[\varphi(X)]}$. Ce qui est bien une densité de probabilité.

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

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

Nous avons alors

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

Nous avons $h(y) = \frac{\varphi(y)f(y)}{E[\varphi(Y)]}$.
Par substitution dans l'équation (2.5), on trouve :

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

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 h_1 grossièrement proche de $|\varphi.f|$ telle que l'on sache simuler suivant la densité :
$$h(y) = \frac{h_1(y)}{\int\limits_{\mathbb{R}^d} h_1(y)dy}.$$
2. On suppose que $Y_i,\ i = 1 \dots n$ est une suite de v.a.i.i.d, $Y_1 \sim \mathcal{L}(h)$. Puis on compare $var\left[\frac{\varphi(Y)f(Y)}{h(Y)}\right]$ et $var[\varphi(X)]$. Pour cela, on utilise généralement le théorème (6) (estimation de variance).

Exemple 10. *Supposons qu'on cherche à approcher* $I = \int\limits_0^1 \sin(\pi x)dx$ *par une méthode Monte-Carlo.*
La quantité I *peut être estimer par deux méthodes différentes.*

Méthode de Monte Carlo Classique

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

Méthode d'échantillonnage préférentiel

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

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 *grossiérement proche de* $|\varphi.f|$.
Posons $h_1(y) = (1-y)\mathbb{1}_{[0,1]}(y)$, *donc*

$$h(y) = \frac{h_1}{\int\limits_0^1 h_1(y)dy} = \frac{1-y}{\int\limits_0^1 (1-y)dy} = 2(1-y),$$

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

Simulation de Y ***selon*** h
Par l'application de la méthode d'inversion (vue dans le chapitre 1), nous avons

$$\begin{aligned} H(y) &= \int\limits_0^y 2(1-t)dt \\ &= \left[(1-t)^2\right]_0^y \\ &= (1-y)^2 - 1. \end{aligned}$$

Pour tout $0 \le u \le 1$, *posons* $u = H(y) = (1-y)^2 - 1 \implies y = 1 - \sqrt{1+u}$.
Donc $Y_i = 1 - \sqrt{1+U_i}$ *avec* $U_1, ..., U_n \sim \mathcal{U}([0,1])$.

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

Algorithme d'estimation de variance des deux méthodes.

Debut

lire (n) ;

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

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

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

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

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

fin pour ;

%estimation de la variance de la méthode classique.

$var1 = 0$;

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

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

fin pour ;

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

%estimation de la variance de la méthode échantillonage préférentiel.

$var2 = 0$;

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

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

fin pour ;

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

Fin.

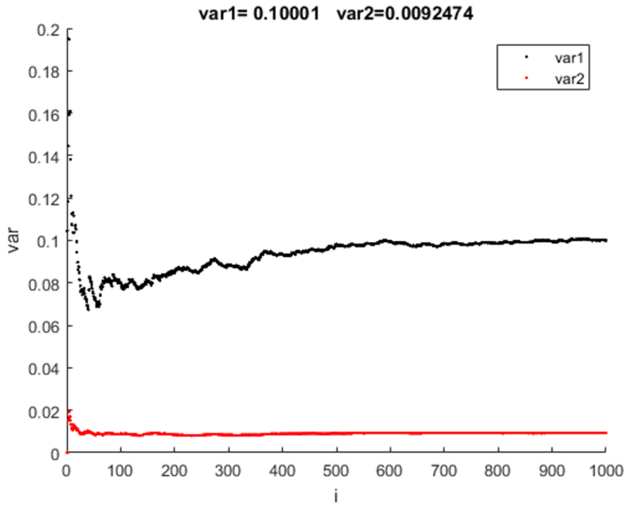

FIGURE 2.3 – La courbe en noir représente l'estimation de la variance de la méthode de MC classique. La courbe en rouge représente l'estimation de la variance de la méthode d'échantillonnage préférentiel.

D'après le programme de cet algorithme (voir figure(2.3)), nous trouvons $var1 = var[\varphi(X)] = 0.1$ *pour la méthode de Monte-Carlo classique et* $var2 = var[\frac{\varphi(Y)f(Y)}{h(Y)}] = 0.009$ *pour la méthode d'échantillonage préférentiel. La variance est donc réduite avec la méthode d'échantillonage préférentiel. De plus, le nombre des itérations nécéssaires 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'échontillonnage 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 trés 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 intérvalle de confiance* 2 *fois plus réduit que celui de Monte-Carlo classique.*

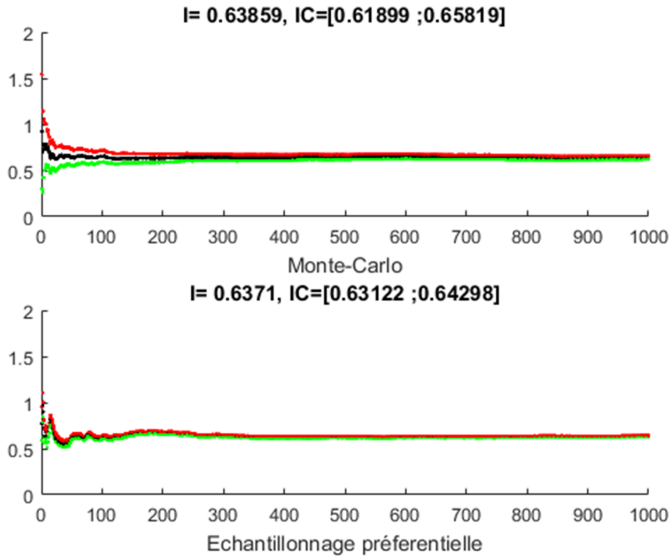

FIGURE 2.4 – Les valeurs approchées de la quantité I et les intervalles de confiance de la méthode MC classique et celle d'échantillonnage préférentiel.

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

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 φ, telle que l'on sache calculer $E[h(X)]$ (le fait que h soit proche de φ signifie que la quantité $var[\varphi(X) - h(X)]$ est petite).
2. Estimer les variances $var[\varphi(X)]$ et $var[\varphi(X) - h(X)]$ 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*

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

Méthode Monte-Carlo classique

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

Méthode variable de contrôle

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

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

Donc nous pouvons faire l'estimation suivante

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

Nous estimons les deux variances dans l'algorithme suivant.

Algorithme d'estimation de variances des deux méthodes.

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 la variance de la méthode classique ;
$var1 = 0$ *;*
pour $i = 1 : n$ ***faire***
  $var1 = var1 + [\exp(U(i)^2) - \exp(V(i)^2)]^2$ *;*
fin pour *;*
$var1 = \frac{var1}{(2*n)}$ *;*
% estimation de la variance de la 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 *;*
$var2 = \frac{var2}{(2*n)} + \frac{4}{3}$ *;*
Fin.

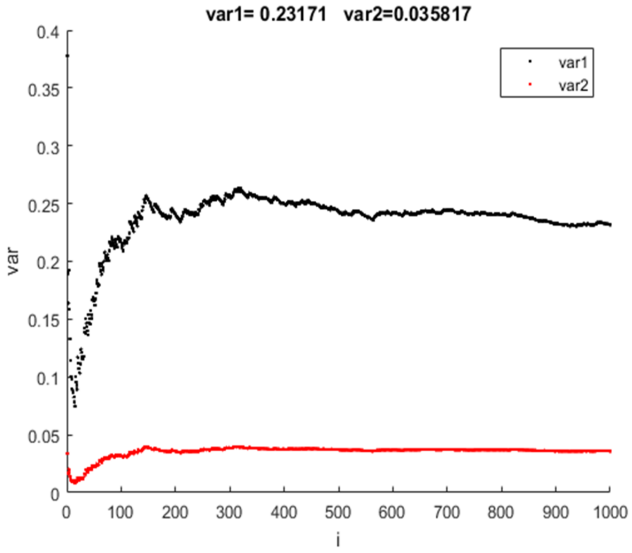

FIGURE 2.5 – La courbe en noir représente l'estimation de la variance de la méthode de Monte-Carlo classique. La courbe en rouge représente l'estimation de la variance de la méthode variable de contrôle.

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 près*que 7 *fois plus petite pour un temps de simulation inférieur à* 6 *fois et un intérvalle de confiance* 2 *fois plus réduit que celui de la méthode de Monte-Carlo classique.*

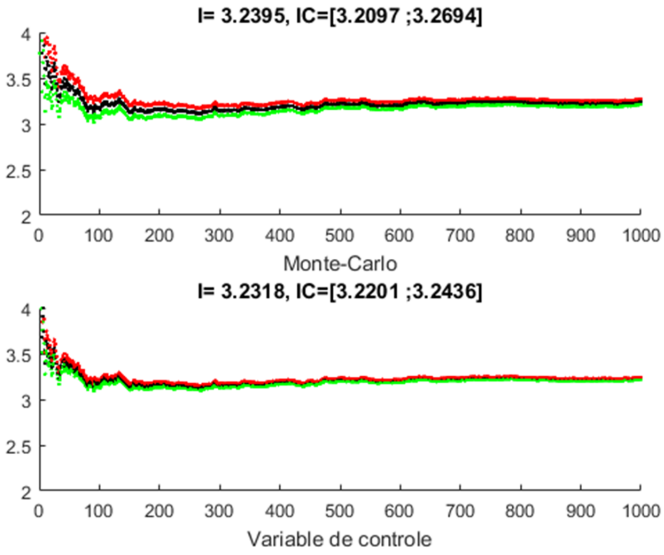

FIGURE 2.6 – Les valeurs approchées de la quantité I et les intervalles de confiance de la méthode Monte-Carlo classique et celle de variable de contrôle.

2.3.3 Méthode variable antithétique

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

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

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

Méthode variable antithétique

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

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

Ce qui donne l'idée pour une deuxième méthode.

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

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

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

La variance est donc toujours réduite.

Preuve 5. *Nous avons,*

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

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

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

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

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

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

Remarque 3. *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* $(1-u_1,...,1-u_d)\in[0,1]^d$ *par une transformation bijective* $g:E\to E$ *telle que pour tout* $x\in E$, $\varphi(g(x))=\varphi(x)$.

2.4 Conclusion

Dans ce chapitre, nous avons abordé les différentes méthodes de Monte-Carlo classique, ainsi que nous avons donné quelques méthodes de réduction de variance. Nous avons surtout vu que l'estimation d'une quantité de type $I=E[\varphi(X)]$ revient à la simulation d'une suite de variable aléatoire $X_1,...,X_n$ i.i.d selon une fonction de densité f donnée. Mais ceci, n'est pas toujours le cas. Dans le chapitre suivant, nous nous intéressons aux cas ou les suites $(X_n)_{n\geq1}$ ne sont pas i.i.d, mais constituent une chaine de Markov.

Chapitre 3

Méthode de Monte-Carlo par chaînes de Markov

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

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

peut-être approchée par un estimateur Monte-Carlo ;

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

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 chaînes 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 reconstruire 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 valeurs dans un ensemble S supposé fini, $S = \{1, 2, ..., M\}$ appelé espace des états, si pour tout $n \geq 1$, et pour toute suite $(x_1, ..., x_{n-1}, x, y)$ telle que $P(X_{n+1} = y | X_n = x, X_{n-1} = x_{n-1}, ..., X_1 = x_1) \geq 0$ alors,

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

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 en une étape, la probabilité d'aller de l'état x vers l'état y la quantité,

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

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 $x, y \in S \quad 0 \leq p_{xy} \leq 1$ (Encadrement des coefficients)
- pour tout $x \in S \quad \sum_{y \in S} p_{xy} = 1$ (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 étapes est notée,

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

et la matrice de transition en n coups est notée,

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

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

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

Proposition 5. *Soit $(X_n)_{n \geq 1}$ une chaîne de Markov de loi initiale μ et de matrice de transition P. Alors pour tout entier naturel $n \geq 1$, la loi de X_n est donnée par*

$$P(X_n) = \mu P^n. \tag{3.6}$$

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,

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

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

Récurrence et Transience

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

$$T_x = \inf \{n \geq 1,\ X_n = x\}. \tag{3.8}$$

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

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

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

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

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

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

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 :

$$p_{xx} = 1. \tag{3.12}$$

3.1.6 Distribution stationnaire

Une probabilité π 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,

$$\pi = \pi P. \tag{3.13}$$

Où $\pi = (\pi_1, \pi_2, ..., \pi_i, ...)$ avec $\sum_{i \in S} \pi_i = 1$.

Proposition 6. *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 π, si on a pour tous $x, y \in S$,

$$\pi_x p_{xy} = \pi_y p_{yx}. \tag{3.14}$$

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 π, alors π est une probabilité invariante.*

Preuve 6. *Pour montrer que π 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 8. *(Convergence)*
Soit $(X_n)_{n \geq 1}$ une chaîne de Markov irréductible, admet une unique distribution stationnaire π, alors, pour toute fonction $\varphi : S \to \mathbb{R}$ On a :

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

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 9. *(Vitesse de convergence(TCL))*
Supposons que S est fini et P est une matrice de transition irréductible, apériodique et admettant une probabilité invariante π, alors :

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

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

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

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

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 chaînes de Markov

La méthode MCMC est publiée en 1953 par Metropolis et ses co-auteurs [14], est étendue en 1970 par Hastings [13]. En 1984, les frères Geman [10] proposent l'échantillonneur de Gibbs pour la restauration bayésienne d'images, cet échantillon-neur 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 π : 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 π, qui garantit de plus la convergence en loi vers π.
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 π. Désormais, la démarche est inverse : on part d'une loi π donnée et on cherche à construire une chaîne de Markov dont π 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 a 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 π donnée, et on cherche à construire une chaîne de Markov dont π est sa distribution stationnaire. Nous commençons notre visite des algorithmes MCMC par celui de Metropolis présenté dans le paragraphe suivant.

Algorithme de Metropolis

L'algorithme de Metropolis a été proposé par Metropolis et al 1953 [14]. Cet algorithme consiste à construire une chaîne de Markov qui admet π comme unique distribution stationnaire et possède la propriété ergodique. Pour ce faire, on fixe une loi π suivant laquelle on aimerait simuler ou dont on voudrait estimer une quantité de type $\int_S \varphi(x)\pi_{dx}$, nous appellerons π la loi cible.
Soit P la matrice de transition ayant π 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 $q_x = [q_{x1}, ..., q_{xM}]$.

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 $q_x = [q_{x1}, ..., q_{xM}]$.
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
$$\begin{cases} X_{n+1} = Y = y & \text{si} \quad U \leq r_{xy}, \\ X_{n+1} = X_n = x & \text{sinon.} \end{cases}$$

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 $Y(i) \sim q_x$;
  $r = min\left\{1, \frac{\pi_y(i)}{\pi_x(i)}\right\}$;
  générer $U \sim \mathcal{U}([0,1])$;
  si $U \leq r$ **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 7. *La suite aléatoire $(X_n)_{n\geq 1}$ construite ici est une chaîne de Markov de matrice de transition P, définie par :*

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

De plus π est une probabilité réversible (donc invariante) pour P.

Proposition 8. *Si la probabilité π 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 7. *si $x \neq y$, $p_{xy} = q_{xy} r_{xy} > 0 \Rightarrow$ Tous les états communiquent.*

□

Proposition 9. *(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 π est réversible pour cette chaîne. En particulier, π est son unique mesure d'équilibre.

Preuve 8. *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,

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

$$\Rightarrow p_{yx} = p_{xy}\frac{\pi_x}{\pi_y} \Rightarrow \pi_y p_{yx} = \pi_x p_{xy}$$

□

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 ce qui suit.

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 de Metropolis.

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

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

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 $Y(i) \sim \mathcal{L}(q_x)$;
  $r = min\left\{1, \frac{\pi_y(i)q_{yx}(i)}{\pi_x(i)q_{xy}(i)}\right\}$;
  générer $U \sim \mathcal{U}([0,1])$;
  si $U \leq r$ **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

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

Par ailleurs, tout ce qui a été dit dans le cas d'un espace d'états 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 π 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 $q(.|x)$.
- 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 $Y(i) \sim q(.|X)$;
  $r = min\left\{1, \frac{f(Y(i))q(X(i)|Y(i))}{f(X(i))q(Y(i)|X(i))}\right\}$;
  générer $U \sim \mathcal{U}([0,1])$;
  si $U \leq r$ **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 Metropolis-Hastings

Il est à noter qu'il existe plusieurs autres algorithmes qui découlent de l'algorithme de Metropolis-Hastings, 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 $Y(i) \sim q(.)$;
  $r = min\left\{1, \frac{f(Y(i))q(X(i))}{f(X(i))q(Y(i))}\right\}$;
  générer $U \sim \mathcal{U}([0,1])$;
  si $U \leq r$ **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 $X_i + \varepsilon_i$, ε_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 $\varepsilon(i) \sim q(.)$;
$Y(i) = X(i) + \varepsilon(i)$;
$r = min\left\{1, \frac{f(Y(i))q(X(i)-Y(i))}{f(X(i))q(Y(i)-X(i))}\right\}$;
générer $U \sim \mathcal{U}([0,1])$;
si $U \leq r$ **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
$(X_1^{(n)}, X_2^{(n)}, ..., X_d^{(n)})$ 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 $q_i(.|x^{(n)})$, $i = 1, ..., d$.
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 $x_{-i}^{(n+1)} = (x_1^{(n+1)}, ..., x_{i-1}^{(n+1)}, x_{i+1}^{(n+1)}, ..., x_d^{(n+1)})$ (tel que $i = 1, ..., d$). Alors l'algorithme de cette méthode peut s'écrire sous la forme suivante :

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

Remarque 4. *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,*

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

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 utilisée pour la simulation de la densié f est la loi normale centré 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 $Y(i) \sim \mathcal{N}(X(i), \sigma)$;
  $r = min\left\{1, e^{X^2(i)-Y^2(i)} * \left(\frac{2+sin(Y^5(i))+sin(Y^2(i))}{2+sin(X^5(i))+sin(X^2(i))}\right)\right\}$;
  générer $U \sim \mathcal{U}([0,1])$;
  si $U \leq r$ ***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 $X_1 \sim \mathcal{U}([-2,2])$.

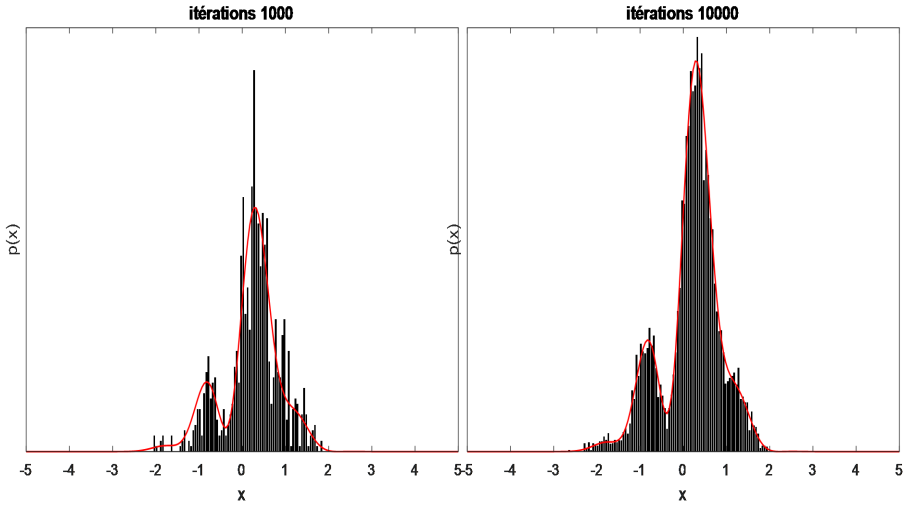

FIGURE 3.1 – Simulations de la loi cible f avec respectivement 1000 itérations et 10000 itérations.

Remarque 5. *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'echantillonneur de Gibbs.*

3.2.3 Echantillonneur de Gibbs

Il s'agit d'un cas particulier de Metropolis-Hastings où touts 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 $f(x) = f(x_1, x_2, ..., x_d)$ pour tout indice j entre 1 et d, on note l'ensemble
$x_{-j} = (x_1, ..., x_{j-1}, x_{j+1}, ..., x_d)$.
Ici x_{-j} correspond à x privé de sa j-ième élément donc la densité $f(x)$ peut être donnée par $f(x) = f(x_j, x_{-j})$.
Tandis que $f(.|x_{-j})$ est la densité conditionnelle correspondant à $(d-1)$ coordonnées gelées et $f(x)$ la densité marginale de ces $(d-1)$ coordonnées. Alors, pour tout j et pour toute suite x_{-j}, si on sait simuler suivant la densité conditionnelle $f(.|x_{-j})$, alors :

1. Partant de $X_n = x$, on tire successivement des coordonnées j de 1 à d.
2. On simule Y selon la loi $f(.|x_{-j})$.
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 $X_1^{i+1} \sim f_1(x_1|x_2^i, ..., x_d^i)$;
  générer $X_2^{i+1} \sim f_2(x_2|x_1^{i+1}, ..., x_d^i)$;
  $\vdots$
  générer $X_d^{i+1} \sim f_d(x_d|x_1^{i+1}, ..., x_{d-1}^i)$;
  $i = i + 1$;
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 6. *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 :
$Y \sim \mathcal{L}(q(.|x)) \quad et \quad Y = X + \varepsilon, \quad \varepsilon \sim \mathcal{U}([-a, a]),\ a \in \mathbb{R}_+^*$.

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 $\varepsilon(i) \sim \mathcal{U}([-a, a])$;
  $Y(i) = X(i) + \varepsilon(i)$;
  $r = min\left\{1, e^{\frac{X^2(i) - Y^2(i)}{2}}\right\}$;
  générer $U \sim \mathcal{U}([0, 1])$;
  si $U \leq r$ **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 précédent pour des valeurs initiales $X_1 = 10$ et $X_1 \sim \mathcal{U}[-a, a],\ a = 3$ respectivement.

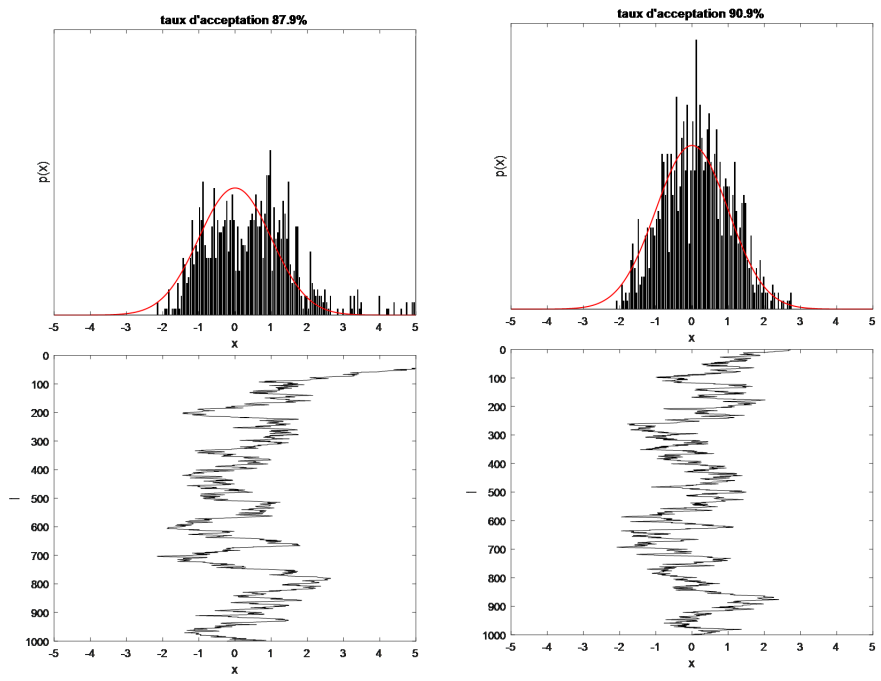

FIGURE 3.2 – Distributions et histogrammes des chaînes simulées suivant $\mathcal{N}(0,1)$ et les séries chronologiques des deux simulations.

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.

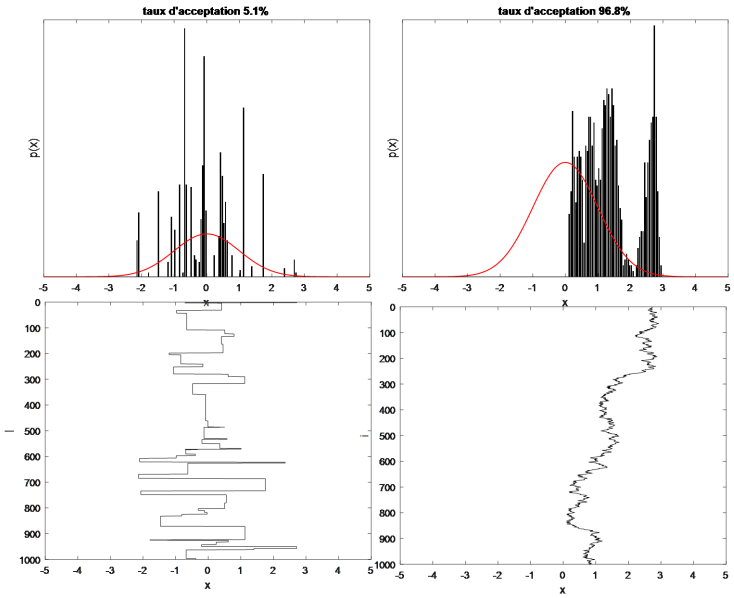

FIGURE 3.3 – Distributions et Histogrammes des chaînes simulées suivant $\mathcal{N}(0,1)$ et les séries chronologiques des deux simulations.

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 $[-30,30]$ (graphe de gauche) 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. A l'opposé avec une loi uniforme sur l'intervalle $[-0.1,0.1]$ (graphe de droite), 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.

3.4 Conclusion

Dans ce chapitre, après avoir donné quelques rappels sur les chaînes de Markov, nous avons abordé quelques méthodes de Monte-Carlo par chaînes de Markov. Comme nous avons présenté les différents algorithmes associés à ces méthodes, ainsi que la raison pour laquelle on les utilises. Nous avons vu aussi que la qualité de la chaîne générée par ces algorithmes dépend du choix de la valeur de l'état initial ainsi que le choix de la densité de la loi de proposition. Dans le chapitre suivant, nous nous intéressons à des techniques utilisées généralement pour améliorer la qualité de la chaîne de Markov générée par l'application des algorithmes de MCMC.

Chapitre 4

Implémentation et Applications

Dans le chapitre précédent, nous avons introduit les algorithmes MCMC et, bien que nous ayons omis la plus part des détails théoriques, nous avons affirmé que sous des conditions assez générales, ces algorithmes sont convergents car les chaînes qu'ils produisent sont ergodiques. Cependant ces résultats sont insuffisants lorsqu'on considère la mise en œuvre des méthodes MCMC. C'est bien l'objectif de ce chapitre. En effet, dans ce chapitre nous abordons les difficultés rencontrées lors de l'implémantation de ces algorithmes puis nous donnons quelques techniques permettant d'éviter ces difficultés, et nous terminons ce chapitre par un exemple d'application.

4.1 Détails d'implémentation

Les performances des méthodes de Monte-Carlo par chaînes de Markov sont grandement influencées par le choix fait par l'utilisateur. En effet outre le choix du point de départ (la valeur de l'état initial X_1), l'utilisateur doit aussi choisir une fonction instrumentale dans laquelle on génère des échantillons qui ajustent aux mieux la distribution cible. Donc afin d'illustrer l'effet de ces variantes, des échantillons de variables aléatoires distribuées selon la loi de mélanges suivante :

$$f(x) = (1/3)\frac{1}{\sqrt{2\pi}}e^{-\frac{1}{2}x^2} + (2/3)\frac{1}{\sqrt{2\pi}}e^{-\frac{1}{2}(x-7)^2}, \tag{4.1}$$

simulée par l'algorithme de Metropolis-Hastings avec différents choix de paramètres seront présentés à titre d'exemple.

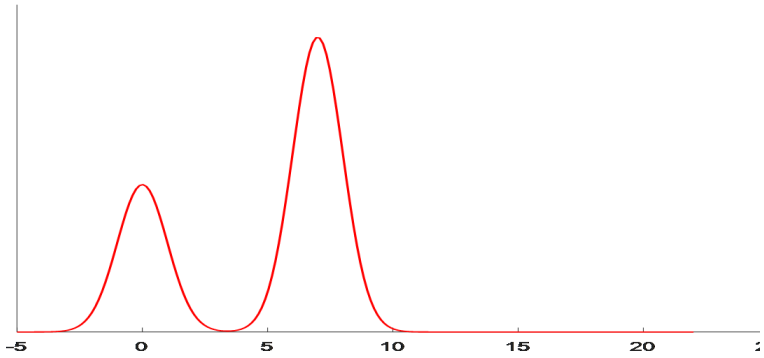

FIGURE 4.1 – Distribution de la loi de mélange $f(x)$.

Le premier paramètre à choisir est la distribution instrumentale qui a un impact considérable sur les performances de l'algorithme. Nous détaillons ceci dans la sous-section suivante.

4.1.1 Distribution instrumentale

Le choix de la distribution instrumentale (de proposition) est crucial mais difficile, car c'est ce choix qui influencera sur la qualité de l'algorithme. En effet, un choix judicieux favorisera une exploration rapide de l'espace d'états et accélérera en conséquence la vitesse de convergence de la chaîne vers la distribution cible. Donc pour des raisons de performance, on choisira évidement une distribution q facile à simuler et surtout proche le plus possible de la densité cible. La figure (4.2) nous montre l'effet du choix de la densité de proposition sur la simulation de loi cible donnée dans l'équation (4.1).

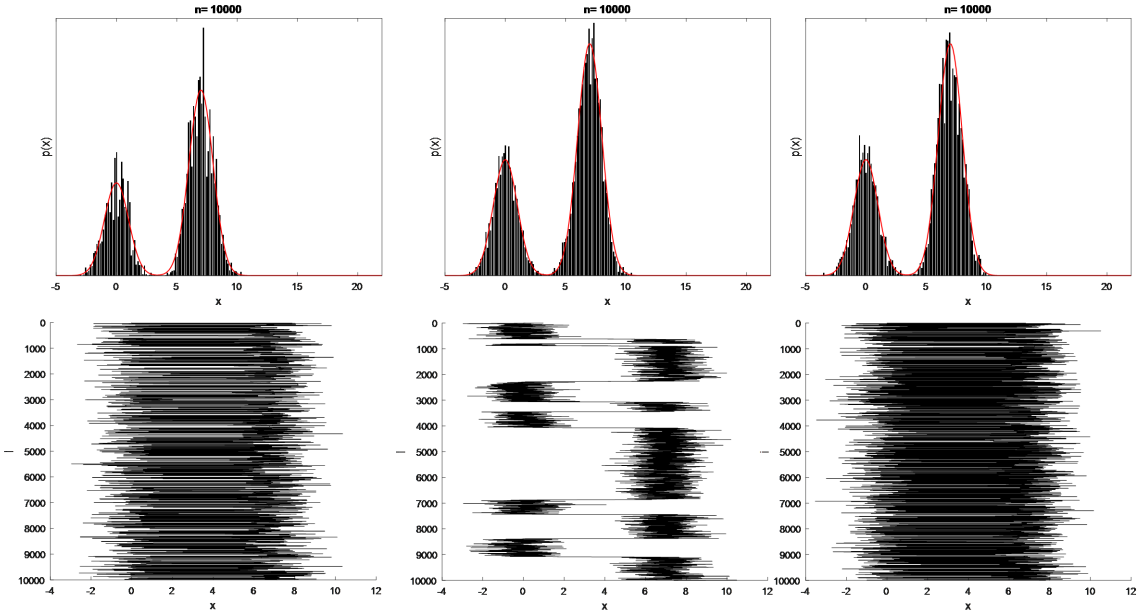

FIGURE 4.2 – Simulations de 10000 itérations avec $q(y|x) \sim \mathcal{N}(x, 5)$, $q(y|x) \sim \mathcal{N}(x, 1)$ et $q(y|x) \sim \mathcal{U}([x-10, x+1])$ respectivement (de gauche à droite).

A partir de la figure (4.2), nous pouvons voir que le meilleur choix de la loi de proposition est la loi normale $\mathcal{N}(x, 1)$ (graphe de milieu). En effet, pour $n = 10000$ cette dernière donne des meilleures approximations de la densité cible, par rapport aux deux autres choix (la loi $\mathcal{N}(x, 5)$ et la loi $\mathcal{U}([x-10, x+1])$).

Après le choix de la distribution instrumentale viens le tour pour le choix de la valeur de l'état initial de la chaîne à générer. Le choix de cette valeur à une influence sur la vitesse de convergence de l'algorithme.

4.1.2 État initial

Pour l'exècution des algorithmes de MCMC, il est nécessaire de choisir le premier état dans lequel se trouve la chaîne. Cette valeur initiale appelée aussi la graine notée X_1, influence fortement sur la vitesse de convergence de l'algorithme, donc nécessite d'être choisie judicieusement. Généralement, ce choix est fait au hasard, ou bien uniformément sur le support de définition (région d'intérêt) de la loi cible f, mais lorsque nous disposons de certaines informations sur la fonction cible, il est préférable de choisir la valeur de l'état initial en fonction de ces informations. La figure (4.3) illustre les résultats de trois simulations avec respectivement $X_1 = -20$, $X_1 = 0$ et $X_1 = 20$. Les graphiques de cette figure montrent le comportement de la fonction cible en fonction du choix de la valeur de l'état initial.

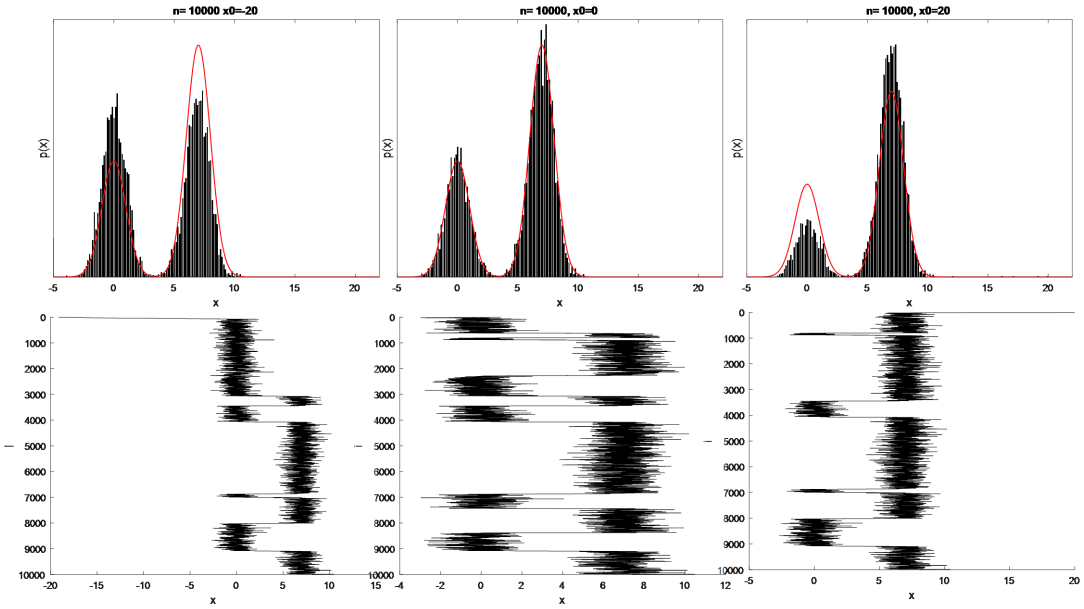

FIGURE 4.3 – Simulations de 10000 itérations avec respectivement $X_1 = -20$, $X_1 = 0$ et $X_1 = 20$.

A partir de cette figure, et en comparant les résultats obtenues, nous constatons que la meilleure valeur de l'état initial à choisir est $X_1 = 0$ (graphe du milieu). En effet, pour cette valeur et pour $n = 10000$, la chaîne simulé converge rapidement vers la loi cible, contrairement aux valeurs $X_1 = -20$ et $X_1 = 20$ qui prennent du temps pour se déplacer vers la région d'intérêt. Pour ces valeurs, les chaînes simulées convergent lentement vers la loi cible.

Toute fois, pour réduire au maximum l'influence du choix de X_1 sur la simulation de la densité cible, il est d'usage de ne pas tenir compte d'un certain nombre d'itérations, ceci signifie que les premières valeurs générées par l'algorithme ne seront pas prises en compte. Les auteurs anglophones font alors référence à la période de « burn in » ou « période de chauffe » .

4.1.3 Période de chauffe (période de transition)

Toute simulation MCMC peut-être divisée en deux périodes, soit la période de transition qui désigne les itérations dépendantes du choix de la valeur de l'état initial, ces itérations sont très instables, et la période de simulations ou la chaîne à atteint sa distribution stationnaire f. Or le nombre d'itérations de la période de chauffe, dont la durée est notée n_0 est difficile à déterminer. C'est donc à l'utilisateur de l'algorithme de vérifier empiriquement si la période de transition est terminée et s'il est donc possible de commencer l'échantillonnage de f. La figure (4.4)représente les résultats de trois simulations de la distribution cible f : la première est faite sans période de chauffe, la deuxième avec une période de chauffe $n_0 = 500$ et pour finir la troisième avec une période de chauffe $n_0 = 1000$.

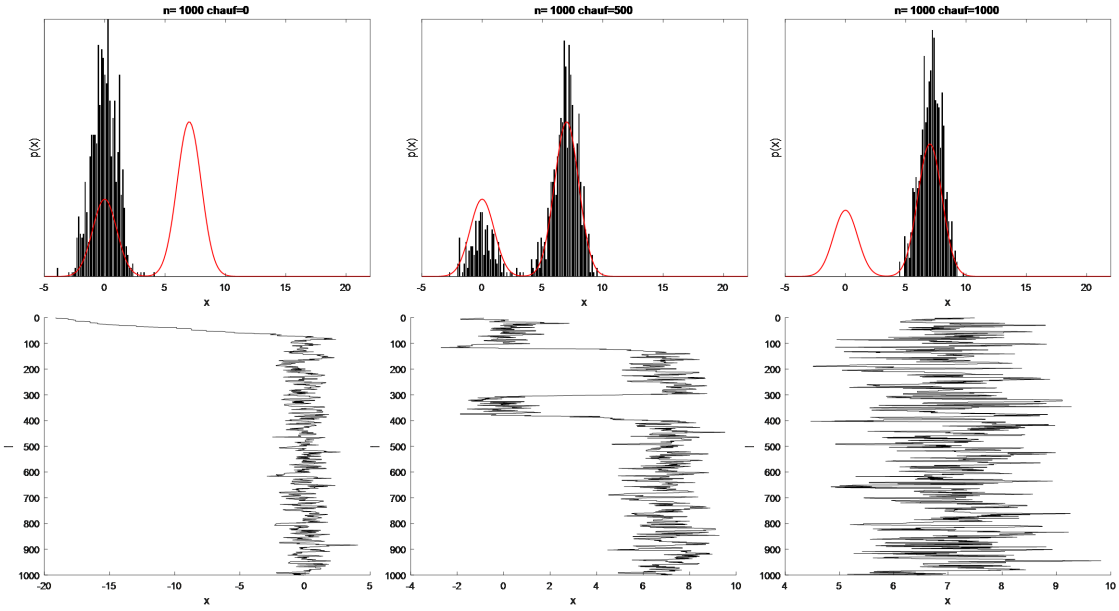

FIGURE 4.4 – Distribution et histogramme ainsi la série chronologique de trois simulations avec respectivement $n_0 = 0$, $n_0 = 500$ et $n_0 = 1000$.

Le premier graphe de le figure (4.4) illustre la simulation de la densité f, l'échantillon est généré sans période de chauffe, nous voyons ici que la plupart des propositions sont tombées dans la queue de la région d'intérêt de f, ceci est une conséquence de l'influence de la valeur de l'état initial qui est loin de la région d'intérêt. Le deuxième graphe visualise la simulation de f qui est faite avec la même valeur de l'état initial que le premier cas mais avec une période de chauffe $n_0 = 500$, nous voyons que l'échantillon généré enveloppe bien le support de définition de f, et nous remarquons que l'influence de la valeur de l'état initial est réduite en utilisant la période de chauffe. Dans le troisième cas, nous avons utilisé une période de chauffe grande $n_0 = 1000$, nous voyons d'après le graphe que la période de chauffe est dépassée, cela cause une perte d'information sur la loi cible donc il est inutile de prolonger la période de chauffe.

Cependant, cet exemple illustre bien l'importance de la période de chauffe, en effet elle réduit l'influence du choix de la valeur de l'état initial X_1 sur la simulation de la densité cible.

4.1.4 Dilution de l'échantillon

Lorsque les candidats acceptés par un algorithme MCMC sont très corrélés entre eux, chaque nouveau candidat accepté apporte très peu d'informations à l'échantillon. Par mesure d'économie, il peut être utile de conserver en mémoire seulement une fraction des candidats acceptés par l'algorithme. Cette opération de dilution de l'échantillon consiste à multiplier le nombre de valeurs à généré par D et d'enregistrer chaque D-ième valeur générée par l'algorithme. La figure (4.5) montre l'effet de la dilution de l'échantillonnage sur trois simulations de 1000 itérations, la première avec une période de transition $n_0 = 500$ itérations et une dilution $D = 5$ itérations, la deuxième avec une période de transition 500 itérations et avec une dilution $D = 1$ et la troisième sans période de transition et une dilution $D = 2$ itérations.

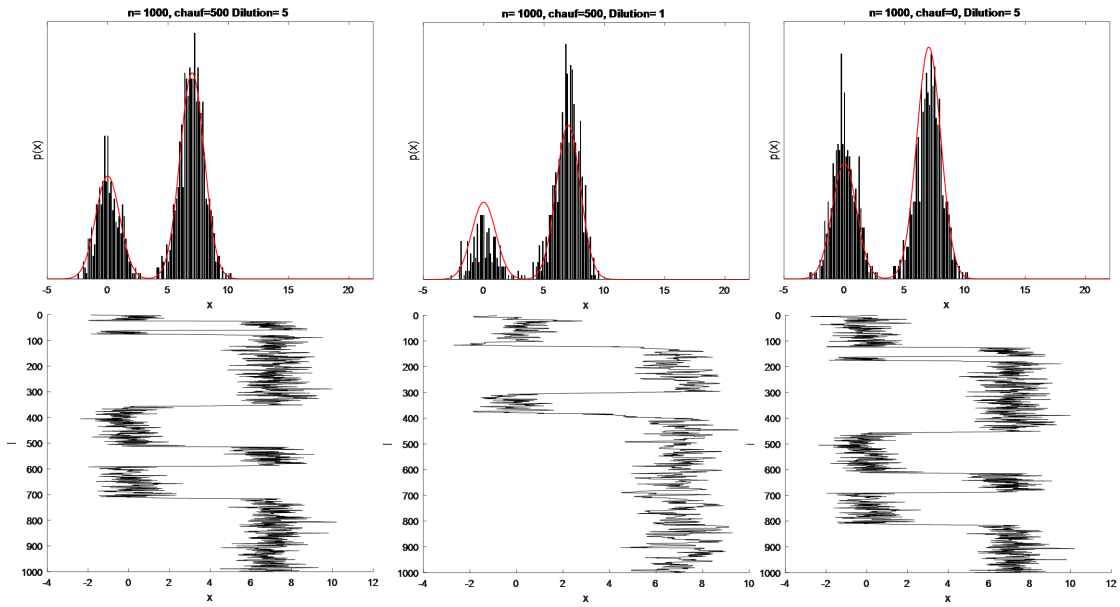

FIGURE 4.5 – Distribution et histogramme ainsi la série chronologique de trois simulations avec respectivement $n = 1000,\ n_0 = 500,\ D = 5$, $n = 1000,\ n_0 = 500,\ D = 1$ et $n = 1000,\ n_0 = 0,\ D = 5$.

A partir de la figure (4.5), nous pouvons voir que les graphes sur les cotés donnent des meilleurs résultats lorsqu'on a dilué l'échantillon, contrairement au graphe du milieu qui est fait sans dilution. En effet, la fonction instrumentale a l'occasion de parcourir plus souvent le support de définition de la distribution cible, donc les points conservés reflètent mieux la densité cible.

Un problème capital inhérent à l'utilisation de méthodes MCMC est l'évaluation de la convergence des chaînes construites. Plusieurs méthodes ont été proposées dans la littérature pour vérifier la convergence de ces méthodes [18], la section suivante présente quelques un de ces outils.

4.2 Contrôle de convergence des méthodes MCMC

En général, pour les méhodes MCMC, il existe deux formes de convergence qui nécessite l'évaluation :

1. Convergence vers la distribution stationnaire.
2. Convergence des moyennes.

4.2.1 Convergence vers la distribution stationnaire

La convergence en distribution stationnaire est nécessaire, car nous cherchons à approximer la densité cible f. Il faut donc s'assurer que notre chaîne atteigne sa distribution stationnaire. Même si la théorie nous indique que la chaîne convergera, il lui faudra un nombre infini d'itérations pour nous assurer. Dans un environnement de simulation, le facteur de réduction

d'échelle proposée par Gelman et Rubin (1992) [9] nous aidera à évaluer l'atteinte de la stationnarité de notre chaîne.
Ce critère consiste à construire M chaînes parallèles avec différentes valeurs initiales, puis calculer le facteur de réduction d'échelle en fonction de variances inter-chaînes et intra-chaînes.
Soit M le nombre de chaînes de Markov construites de longueur n.
Les variances inter-chaînes B_n et intra-chaînes W_n pour ces M chaînes de Markov sont définies respectivement par :

$$B_n = \frac{n}{M-1}\sum_{m=1}^{M}(\overline{X}_m - \overline{X})^2.$$

$$W_n = \frac{1}{M}\sum_{m=1}^{M} S_m^2.$$

avec

$$S_m^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_m^i - \overline{X}_m)^2.$$

$$\overline{X}_m = \frac{1}{n}\sum_{i=1}^{n} x_m^i.$$

$$\overline{X} = \frac{1}{M}\sum_{m=1}^{M} \overline{X}_m.$$

où x_m^i est le i-ème échantillon de la m-ième chaîne, $\overline{X}$ est la moyenne des M chaînes et $\overline{X}_m$ est la moyenne de la m-ième chaîne.

Puisque l'estimation de la variance totale de X_n peut s'estimer par une somme pondérée de la variance intra-chaîne et la variance inter-chaîne, nous avons la statistique :

$$\hat{var}(X_n) = \frac{n-1}{n}W_n + \frac{B_n}{n}.$$

Cette statistique à pour particularité de surestimer la valeur de $var(X_n)$, lorsque n est fini et que l'hypothèse que les valeurs des états initiales soient bien réparties sur le support de la fonction f est vérifiée. De plus $\hat{var}(X_n)$ sera sans biais pour $var(X_n)$ sous l'hypothèse que la chaîne ait atteint la stationnarité. Le facteur de réduction d'échelle associée à X_n est défini par :

$$\sqrt{\hat{\rho}_n} = \sqrt{\left[\frac{n-1}{n} + \frac{1}{n}\left(\frac{M+1}{M}\right)\frac{B_n}{W_n}\right]\left[\frac{d\hat{h}_n}{d\hat{h}_n - 2}\right]},$$

avec $d\hat{h}_n$ est le nombre de degrés de liberté d'une loi Student donnée par :

$$d\hat{h}_n = 2\frac{\hat{R}_n^2}{\hat{var}(\hat{R}_n)},$$

avec $\hat{R}_n = \hat{var}(X_n) + \frac{B_n}{n*M}$.

Le facteur de réduction d'échelle représente le facteur pour lequel l'échelle diminue si la chaîne poursuit son échantillonnage indéfiniment. Gelman et Rubin (1992) [9] montrent que $\sqrt{\hat{\rho}_n} \searrow 1$ lorsque $n \longrightarrow +\infty$. Ainsi, une valeur très proche de 1 indique que les chaînes sont très susceptibles d'avoir convergé en leur distribution stationnaire.
Il est souvent commun d'utiliser en plus un diagnostic graphique pour vérifier la convergence.

En superposant chacune des M chaînes exécutées en parallèle sur un graphique, nous examinons visuellement si les M chaînes semblent échantillonnées selon une distribution commune ou non.

4.2.2 Convergence des moyennes

La convergence des moyennes est toute aussi importante que la convergence en la distribution stationnaire. En quoi sert une chaîne de Markov qui a convergé si nous sommes incapables d'obtenir une bonne inférence à partir de celle-ci, La moyenne empirique,

$$\frac{1}{n}\sum_{i=1}^{n}\varphi(X_i),$$

doit converger vers $E[\varphi(X)]$ pour toute fonction φ telle que $E[\varphi(X)] < \infty$. Même si d'un point de vue théorique, le théorème de convergence, nous assure la convergence en moyenne de la chaîne de Markov, un nombre minimal d'itérations doit être respecté pour atteindre une approximation adéquate de $E[\varphi(X)]$ lors de nos simulations.
Une approche défendue par Raftery et Lewis (1992) [17] est de discrétiser la chaîne (X_n) par,

$$Z_n = \mathbb{1}_{(X_n \leq X)},$$

où X est choisi arbitrairement dans le support de f.
En utilisant une approximation markovienne de la loi (Z_n), avec

$$P(Z_n = 1 | Z_{n-1} = 0) = \alpha.$$

$$P(Z_n = 0 | Z_{n-1} = 1) = \beta.$$

Les auteurs ont déduit une évaluation de période de chauffe n_0 nécessaire pour approcher la loi stationnaire à ε prés

$$n_0 \geq \frac{\log\left(\frac{(\alpha+\beta)\varepsilon}{\alpha \vee \beta}\right)}{\log|1-\alpha-\beta|}.$$

Et du temps de simulation n nécessaire pour la convergence de la moyenne empirique des Z_n, δ_n au sens ou $n > n_0$

$$P(|\delta_n - \frac{\alpha}{\alpha+\beta}| < q) \geq \varepsilon'.$$

avec,

$$\delta_n = \frac{1}{n}\sum_{i=n_0}^{n_0+n} Z_i.$$

La valeur obtenue par Raftey et Lewis

$$n \geq \frac{\alpha\beta(2-\alpha-\beta)}{q^2(\alpha+\beta)^3}\phi^{-1}\left(\frac{\varepsilon'+1}{2}\right),$$

où $\phi^{-1}\left(\frac{\varepsilon'+1}{2}\right)$ désigne le quantile d'ordre $\left(\frac{\varepsilon'+1}{2}\right)$ de la loi normale centrée réduite.

Dans la première partie de ce chapitre, nous avons vu que les méthodes de Monte-Carlo par chaînes de Markov sont une classe d'algorithmes permettant de simuler des variables aléatoires provenant de distributions complexes. Dans la deuxième partie de ce chapitre nous donnons un exemple d'application de ces méthodes.

4.3 Recuit simulé et le voyageur de commerce

Le problème du voyageur de commerce est un problème d'optimisation classique. Ce dernier est utilisé pour minimiser le parcour d'un trajet et ainsi minimiser le coût de transport. Il permet de résoudre de nombreux problèmes que ce soit en planification (rammassage scolaire, circuits de transport), en logistique (distribution à partir d'un dépôt vers plusieurs distinations), ou encore dans des domaines plus éloignés comme la génétique.

Supposons qu'un commerçant doit se rendre en N villes, puis revenir à son point de départ ou non. Afin de diminuer son coût de transport, il faut lui trouver le trajet le plus court joignant les N villes. Une idée consiste à calculer la longueur de chaque trajet et choisir ainsi celui de longueur minimale. On peut identifier un trajet avec une permutation sur l'ensemble des N villes. Donc Il y a $N!$ façons possibles d'effectuer le trajet. Il est donc clairement impossible d'effectuer une recherche exhaustive si N est trés grand. En effet, pour $N = 10$, on aurait à explorer $10! = 3628800$ chemins pour déterminer le chemin optimal. On remarque donc que lorsque le nombre de villes augmente, le nombre de chemin à explorer augmente de manière exponentielle. Il est donc dans notre intérêt d'utiliser un algorithme d'optimisation pour résoudre ce problème.

Dans cette section, nous utilisons deux méthodes, la première méthode est basée sur l'algorithme de Metropolis-Hastings tandis que la deuxième est basée sur l'algorithme du recuit simulé.

Tout d'abord, on doit définir la fonction à minimiser. Il s'agit de la distance que parcourt le voyageur. Notons par S_N, l'ensemble des chemins que peut parcourir le voyageur. Un chemin $\sigma \in S_N$ peut-être donné sous plusieurs formes :

$$\begin{aligned} \sigma = &\ (1, 2, 3, ..., N, .., 1). \\ \sigma = &\ (1, 3, 5, ..., N, ..., 1). \end{aligned}$$

Autrement dit, le chemin σ est un vecteur contenant l'ensemble ordonné des villes que le voyageur visite. On peut donc noter :

$$\sigma = (\sigma(1), \sigma(2), ..., \sigma(N)),$$

où $\sigma(i)$ représente la i-ème ville que le commarçant visite.

Supposons qu'on connaît les distances entre les différentes villes et posons M la matrice de ces distances. Ainsi la distance d'un chemin σ est donnée par la relation :

$$D_\sigma = \sum_{i=1}^{N} d(M_{\sigma(i)}, M_{\sigma(i+1)}).$$

Notre objectif, consiste alors à minimiser la fonction suivante :

$$f : \sigma \in S_N \to D_\sigma.$$

4.3.1 Algorithme de Metropolis-Hastings

L'algorithme de Metropolis-Hasting est le suivant :

- **Etape 1 :** Choisir un noyau de transition Q. Choisir un chemin aléatoire σ_1 (Initialisation). Définir le nombre des itérations n.
- **Etape 2 :** Simuler σ_1 selon la transition $q(\sigma_1, \sigma_2)$.
- **Etape 3 :** Calculer le rapport d'acceptation/rejet :

$$r(\sigma_1, \sigma_2) = \frac{f(\sigma_1)q(\sigma_2, \sigma_1)}{f(\sigma_2)q(\sigma_1, \sigma_2))}.$$

– **Etape 4 :** Tirer U suivant une loi uniforme sur $[0,1]$. Si $U \leq r(\sigma_1, \sigma_2)$, poser $\sigma_1 = \sigma_2$, sinon conserver σ_1 ($\sigma_1 = \sigma_1$).
– **Etape 5 :** Répéter l'algorithme n fois.

La fonction f correspond à la loi stationnaire de la chaîne de Markov qu'on veut simuler par cet algorithme. Ici et afin de s'assurer de la convergence de l'algorithme vers le minimum, on choisira $f : \sigma = \exp(-D_\sigma)$.

Ici on choisira un noyau de transition symétrique c'est à dire que

$$q(\sigma_1, \sigma_2) = q(\sigma_2, \sigma_1), \ \forall \sigma_1, \sigma_2 \in S_N.$$

Notons ici qu'il y a plusieurs façons pour choisir le noyau de proposition, la littérature foisonne sur ce sujet.

Dans notre cas, nous définissons le noyau de transition comme une fonction de permutation. Ce noyau de proposition est défini comme suit : partant de σ, on tire i et j uniformément entre 1 et N, puis on permute entre $\sigma(i)$ et $\sigma(j)$ dans le vecteur définissant σ. Notons que cette transition est symétrique : la probabilité de passer de σ à $\sigma_{i,j}$ est égale à celle de passer de $\sigma_{i,j}$ à σ, à savoir $\frac{2}{N^2}$.

Le rapport de Metropolis devient donc : $r(\sigma, \sigma^{'}) = \exp(D_\sigma - D_{\sigma'})$.

On peut donc passer à l'implémentation de cet algorithme.
Pour 10 villes, voici un exemple de représentation :

Tout d'abord, on a besoin d'un ensemble de villes $N = 10$. Chaque ville correspond à un point du carré $[0,1] \times [0,1]$ (voir figure(4.6)).

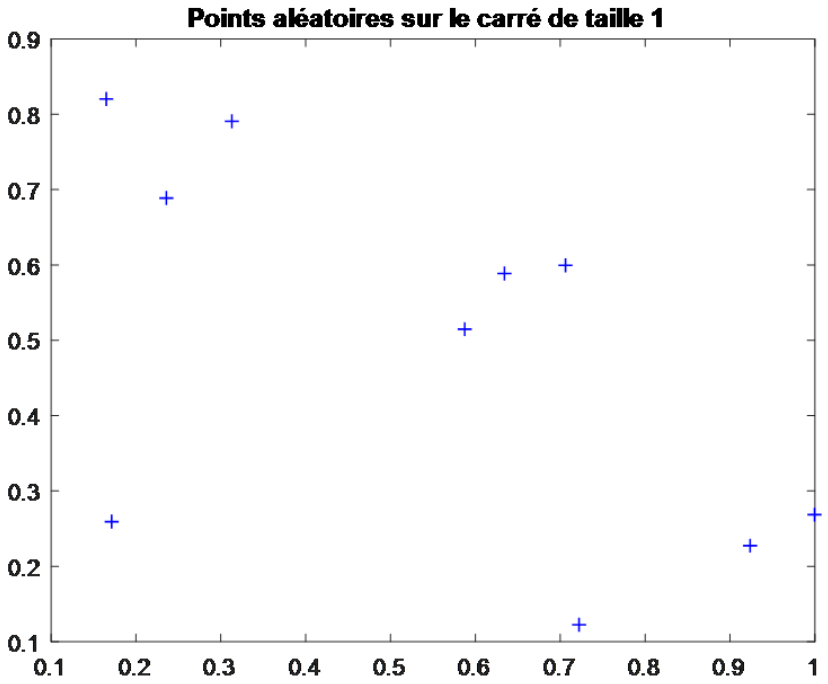

FIGURE 4.6 – La représentation graphique des 10 villes sur le carré $[0,1]^2$.

Pour les trajectoires entre les différentes villes, voir figure (4.7).

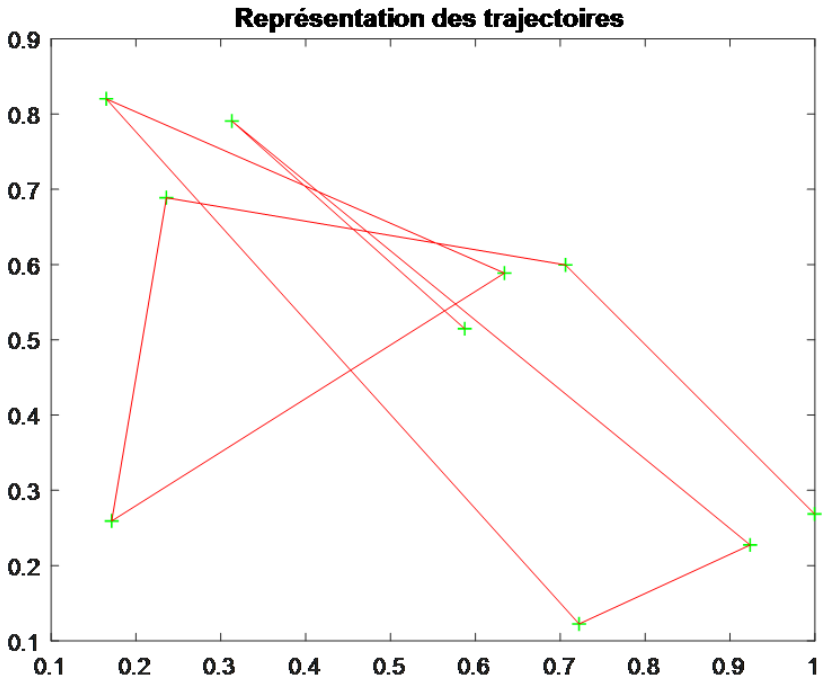

FIGURE 4.7 – Les trajectoires aléatoires entre les 10 villes.

Le parcours donné dans la figure (4.7) n'est clairement pas optimal.

Pour un nombre d'itérations $n = 10^4$, on applique l'algorithme de Metroplis-Hastings, le chemin optimal est donné dans la figure (4.8).

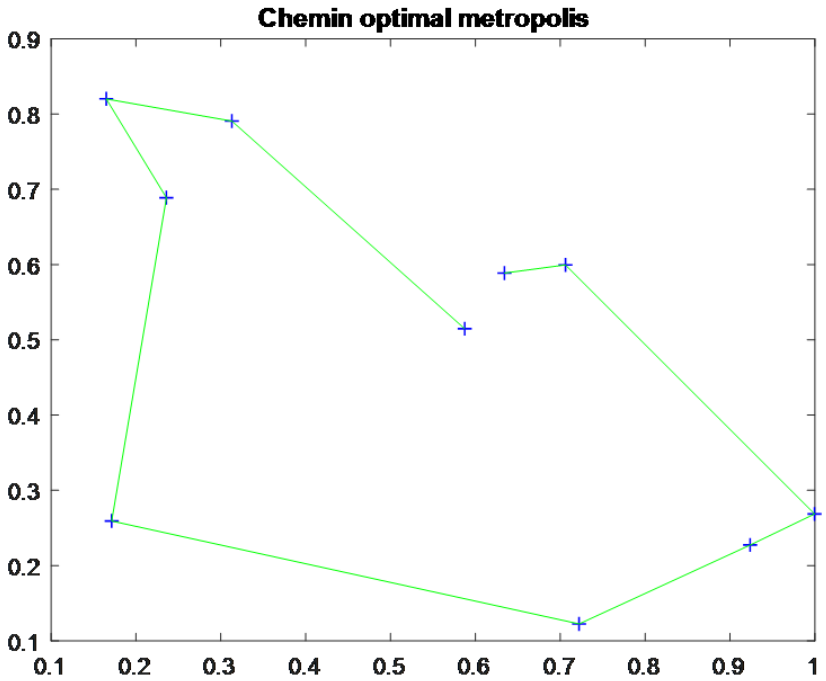

FIGURE 4.8 – Le parcours optimal de Metropolis pour 10 villes.

Et on obtient une distance optimale de 4.5275 en moins d'une seconde.

L'algorithme de Metropolis permet de résoudre le problème de voyageur de commerce dans le cas ou le nombre des villes n'est pas assez grand. Dans le cas contraire, l'algorithme de Metropolis reste piégé dans des minimas locaux. La figure (4.9), donne le chemin optimal de Metropolis pour 50 villes.

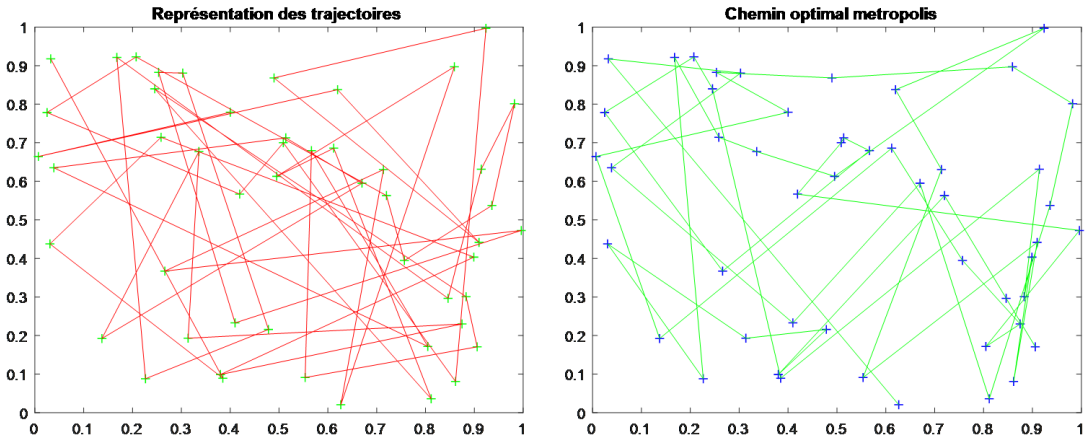

FIGURE 4.9 – Le parcours optimal de Metropolis pour 50 villes.

Le parcours donné dans la figure (4.9) n'est clairement pas optimal. C'est pour quoi il est préférable d'utiliser l'algorithme du recuit simulé pour résoudre ce type de problèmes.

4.3.2 Algorithme de recuit simulé

La méthode du recuit simulé tire son nom de la physique des matériaux et plus spécialement de la métallurgie. Le recuit est une opération qui consiste à laisser refroidir lentement un métal pour améliorer ses qualités. L'idée physique n'est qu'un refroidissement trop brutal capable de bloquer le métal dans un état peu favorable (alors qu'un refroidissement lent permettra aux molécules de s'agencer au mieux dans une configuration stable). C'est cette même idée qui est à la base du recuit simulé. Pour éviter que l'algorithme de Metropolis ne reste piegé dans des minimas locaux, on définit la loi stationnaire f en fonction d'une température $T = T_n$ de sorte que cette dernière décroit lentement en fonction du temps.

La température T permet de contrôler l'acceptation des dégradations de la façon suivante :

- Si la valeur de T est grande, les dégradations sont acceptées avec une probabilité plus grande.
- A la limite, quand T tend vers l'infini, tout voisin est systématiquement accepté.
- Inversement, pour $T = 0$, une dégradation n'est jamais acceptée.

La fonction qui spécifie l'évolution de la température est appelée le schéma de refroidissement.

L'algorithme du recuit simulé fonctionne de la même façon que celui de Metropolis, sauf qu'on doit tenir compte de la température T_n dans le rapport d'acceptation/rejet de Metropolis, celui-là devient donc à l'étape n :

$$r(\sigma, \sigma^{'}) = \exp((D_\sigma - D_{\sigma'})/T_n).$$

La figure (4.10) représente les positions des 50 villes sur le carré $[0, 1]^2$, les trajectoires aléatoires entre ces 50 villes, le chemin optimal obtenu par l'application de l'algorithme de Metropolis et le chemin optimal obtenu par l'application de l'algorithme du recuit simulé.

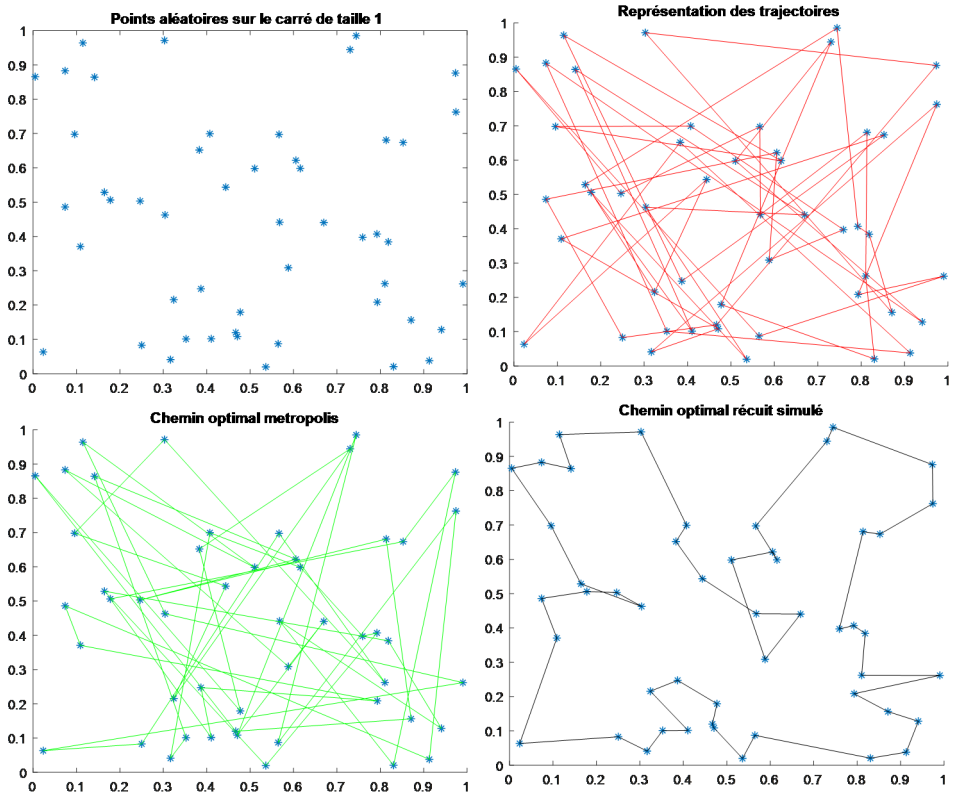

FIGURE 4.10 – De haute en bas et de gauche à droite : les trajectoires aléatoires entre ces 50 villes, le chemin optimal obtenu par l'application de l'algorithme de Metropolis et le chemin optimal obtenu par l'application de l'algorithme du recuit simulé.

Les résultats donnés dans la figure (4.10) sont incomparable. En effet, il est clair que le parcours donné par l'application de l'algorithme de Metropolis n'est clairement pas optimal. Tandis que l'algorithme du récuit simulé fournit un chemin optimal acceptable.

La figure (4.11) représente les positions des 150 villes sur le carré $[0,1]^2$, les trajectoires aléatoires entre ces 150 villes, le chemin optimal obtenu par l'application de l'algorithme du recuit simulé pour $n = 10^4$ itérations et le chemin optimal obtenu par l'application du même algorithme mais pour $n = 10^6$ itérations.

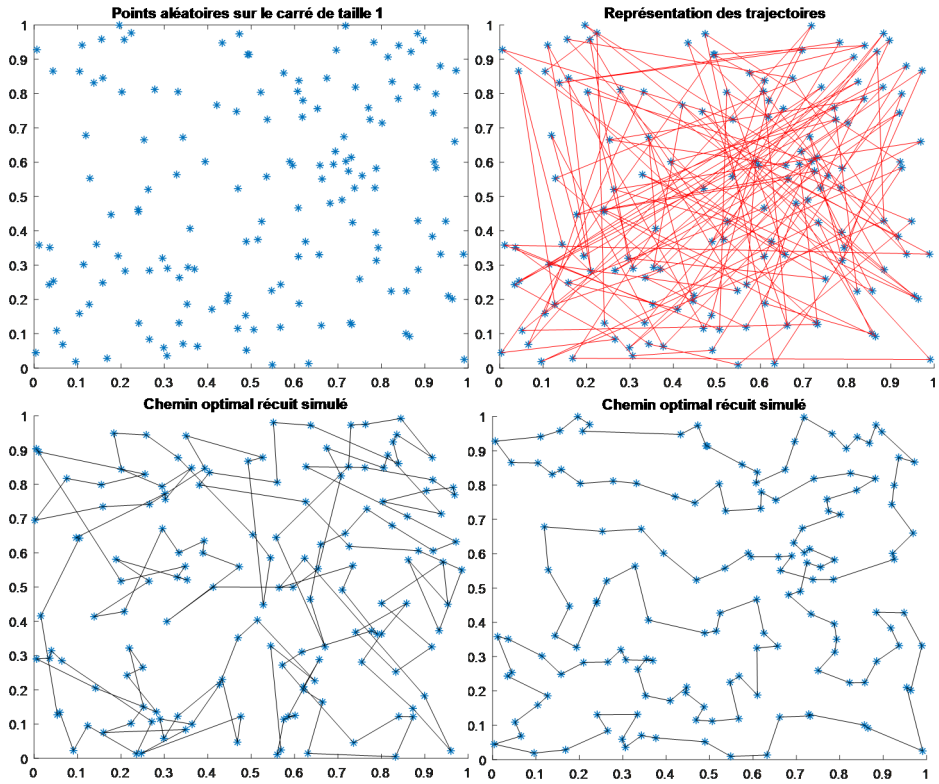

FIGURE 4.11 – Algorithme du recuit simulé pour 150 villes.

Pour $n = 10^6$, on obtient une distance optimale de 0.0582 pour un peu plus d'une minute. Tandis que pour $n = 10^4$, on obtient une distance optimale de 1.324 pour moins de 3 secondes.

Notons ici qu'il y a plusieurs façon pour choisir le schéma de refroidissement, la littérature foisonne sur ce sujet. Parmis ces schémas on peut citer :

1. Schéma géométrique ;

$$T = T_{init} \times (T_{final}/T_{init})^{nb\_ville/n\_iteration}.$$

2. Schéma linéaire ;

$$T = T_{init} + (T_{final} - T_{init}) \times (nb\_ville/n\_iteration).$$

3. Schéma logarithmique ;

$$T = 1/log(n\_iteration).$$

Il est à noter ici, qu'il faut en particulier choisir la température initiale T_{init} suffisamment élevée pour que l'équilibre thermique soit rapidement atteint. Il est à noter aussi que le choix de la température finale T_{final} et le schéma de refroidissement dépend de la nature du problème d'optimisation. Question d'essais multiples.

Pour notre application, nous avons utilisé un schéma géométrique pour une température initiale $T_{init} = 5$ et une température finale $T_{final} = 0.1$.

4.4 Conclusion

Dans la première partie de ce chapitre, nous avons abordé les différents paramètres qui interviennent lors de l'implémentation des algorithmes MCMC à savoir le choix de la valeur

de l'état initial et le choix de la loi de proposition. Par la suite, nous avons introduit la notion de contrôle de convergence des méthodes MCMC. Dans la deuxième partie de ce chapitre, nous avons donné un exemple concret de l'utilisation des méthodes MCMC afin de résoudre un problème d'optimisation complexe à savoir le problème de voyageur de commerce.

Chapitre 5

Conclusion, discussions et perspectives

5.1 Conclusion

Les méthodes de Monte-Carlo présentées dans ce document permettent de modéliser et de résoudre un grand nombre de problèmes pouvant être rencontrés dans des domaines différents tel que la finance, les mathématiques, la physique, la biologie moléculaire et génétique, les télécommunications, les réseaux, la recherche opérationnelle et bien d'autres encore. En effet à nos jours, l'utilisation de ces méthodes recouvre tous les domaines où l'utilisation des méthodes exactes se heurte à des difficultés. Dans ce contexte, on distingue deux grands domaines où la méthode de Monte-Carlo peut être utilisée avec succès :

1. **Problème déterministes ou numériques :** ce sont des problèmes de nature déterministe faisant appel aux calculs numériques, à titre d'exemple on peut citer :
   - Estimation des surfaces.
   - Résolution des équations différentielles.
   - Calculs d'intégrales multiples.
   - Résolution de problèmes d'optimisations combinatoires.
   - Résolution de systèmes d'équations algébriques.
2. **Problème non déterministes et processus aléatoires :** on cite comme exemple de ces problèmes :
   - Systèmes stochastiques de gestion de risque ou de production.
   - Reconnaissance de forme et analyse d'images.

Nous nous sommes intéressés dans ce mémoire à trois aspects différents : d'une part à la simulation des variables aléatoires et d'autre part aux méthodes de Monte-Carlo dite classiques et pour finir nous nous sommes intéréssés aux méthodes de Monte-Carlo par chaînes de Markov.

5.1.1 Simulation

Les travaux présentés dans le cadre de la simulation décrivent quelques techniques de simulation permettant de générer un échantillon de variables aléatoires distribuées selon une densité donnée. On a commencé 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 sont basées. Par la suite, nous avons les trois méthodes principales utilisées pour générées des nombres aléatoires non uniformes, à savoir la méthode d'inversion, la méthode de rejet et la méthode de Box-Müller.

5.1.2 Méthodes de Monte-Carlo classiques

Les travaux présentés dans le cadre des méthodes de Monte-Carlo décrivent une application classique de ces méthodes à savoir l'intégration Monte-Carlo.

Dans un premier temps, nous avons donné quelques 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 avons introduit la notion de l'erreur de l'estimation. Pour finir, nous avons donné quelques méthodes de réduction de variance permettant de minimiser l'erreur d'estimation et améliorer la pécision.

5.1.3 Méthodes de Monte-Carlo par chaînes de Markov

Dans le cadre des méthodes de MCMC, nous sommes intéréssés au cas où la suite des variables aléatoires n'est pas i.i.d mais constituent une chaîne de Markov d'une loi stationnaire donnée. Dans un premier temps, nous avons donné quelques rappel sur les chaînes de Markov, puis nous avons présenté deux 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. Pour finir nous avons donné un exemple d'application à savoir le problème de voyageur de commerce.

5.2 Discussion

Dans ce mémoire et jusqu'à maintenant, nous avons parlé que des avantages de l'utilistion des méthodes de Monte-Carlo. Dans cette section nous donnons quelques inconvénients issues de l'application de ces méthodes.

5.2.1 Intégration Monte-Carlo

Dans le cadre de l'intégration Monte-Carlo, la méthode de Monte-Carlo n'est avantageuse numériquement qu'à partir de la dimension $d = 3$. Même si la plupart des exemples de ce mémoire sont donnés en dimension au plus 2 par souci de simplicité, il convient de ne pas oublier ce fait dans la pratique. Il est à noter qu'il existe des méthodes déterministes dites **quasi Monte-Carlo** qui sont plus rapides que les méthodes de Monte-Carlo classiques mais qui demandent plus d'hypothèses et qui sont plus difficiles à implémenter.

5.2.2 Erreur de l'estimation

A l'opposé des méthodes déteministes (exactes, numériques), l'estimation d'une quantité par une méthode de Monte-Carlo génère une erreur aléatoire dont on ne peut pas la borner. Même si, on peut donner un intervalle de confiance au résultat.

5.2.3 Les paramètres d'implémentation

Comme nous avons vu dans le chapitre 3 et 4, la qualité de la suite générée par l'application des algorithmes de Monte-Carlo dépend du choix de la valeur de l'état initial ainsi que le choix de la distribution de proposition. Le choix de ces paramètre est généralement laissé aux utilisateurs. Donc afin de valider les résultats issus de l'application de ces algorithmes, il est de notre intérêt de faire une batterie d'essais.

5.3 Perspectives

Dans ce mémoire nous avons présenté les méthodes de Monte-Carlo les plus utiles, mais il est important de noter que plusieurs techniques développées dans ces dernières années pour simuler efficacement des processus plus exotiques, ont dû être laissées de côté, par mesure d'économie. Certaines de ces méthodes particulièrement intéressantes sont,

- Algorithme de Monte-Carlo avec plusieurs essaies, un algorithme permettant de faire diminuer l'auto-corrélation des échantillons obtenus lors de la simulation.
- Annealed Importance Sampling. Cette méthode permet de mieux parcourir le support d 'une distribution ayant des régions isolées.
- Echantillonnage séquentiel par bloc, un algorithme qui vise à simuler directement la suite des états à chaque itération.
- L'échantillonnage par tranche, qui est un algorithme facilitant l'échantillonnage d'une distribution connue et qui est très performant lorsque la dimension de cette dernière est élevée.

Table des figures

Liste des tableaux

Bibliographie

[1] Bégin. J. F, *Analyse MCMC de Certains de diffusion avec application au marché européen du carbone*, Technical report, Université de Montréal, Montréal, Canada, Août 2010.

[2] Belaid. N et Djerroud. L, *Les méthodes de Monte-Carlo : (MCMC et PMC). Application*, mémoire Master Recherche Opérationnelle, Béjaïa, Juin 2013.

[3] Bounhar. K et Smaili. N, *Méthodes de Monte-Carlo par chaînes de Markov et leurs applications*, mémoire Master Recherche Opérationnelle, Béjaïa, 2010/2011.

[4] Bergeron. L. V, *L'algorithme Metropolis-Hastings*, Université de Montréal, 2010.

[5] Deschenes. A, *Régression logistique bayésienne comparaison de densités a priori*, Université de Montréal, 2015.

[6] Dodge. Y et Melfi. G, *Premiers pas en simulations*, Springer, Paris, 2008.

[7] Droesbeke. J.J, Fin. J et Saporta. G, *Méthodes bayésiennes en statistique*, Editions TECHNIP, France, 2002.

[8] Gamerman. D, Hedibert. F. L, *Markov Chain Monte-Carlo*, Chapman and Hall/CRC, 2006.

[9] Gelman. A and Rubin. D.B, *Inference from iterative simulation using multiple sequences*, Statistical Science, 7(4) : 457 − 472, Novembre 1992.

[10] Geman. S and Geman. D, *Stochastic relaxation, gibbs distributions, and the bayesian restoration of images*, 6 : 721-741, 1984.

[11] Gilks. W. R, Richardson. S, Spiegelhalter. D. J, *Markov Chain Monte-Carlo in practice*, Springer Science+Business Media Dordrecht, 1996.

[12] Godbout. N.G, *Méthodes MCMC : amélioration d'un algorithme d'adaptation régionale et applications à la climatologie* , Rapport de Stage, Université de Montréal, 7 novembre 2015.

[13] Hastings. W.K, *Monte Carlo Sampling methods using Markov chains and their applications*, volume 57, Biometrica, pages 97-109, April 1970.

[14] Metropolis. N, Rosenbluth. A.W, Rosenbluth. M.N, Teller. A.H, and Teller. E, *Equation of state calculations by fast computings machines*, Chemical Physics, 21(6) :1087-1092, June 1953.

[15] Millet. A, *Méthodes de Monte-Carlo*, Spécialité Modélisation Aléatoire Laboratoire de Probabilités et Modéles Aléatoires, Universités Paris 7 et Paris 1.

[16] Parent. E et Bernier. J, *Le raisonnement bayésien Modèlisation et inférence*, INSEE, Paris, 2006.

[17] Raftery. A et Lewis. S, *[Practical Markov Chain Monte Carlo] : Comment : One Long Run with Diagnostics : Implementation Strategies for Markov Chain Monte Carlo*, Volume 7, 493-497, Number 4 (1992).

[18] Robert. C, *Méthodes de Monte-Carlo par chaînes de Markov*, Economica, 1996.

[19] Robert.P.C et Casella. G, *Méthodes de Monte-Carlo avec R*, Springer, Paris, 2011.

[20] Robert. P. Christian, *Le choix bayésien. Principes et Pratique*, Springer, 2006.

[21] Ruegg. A, *Processus stochastique*, Presse Polytechnique Romandes, 1989.

[22] Touzin. G, *Etude des méthodes de Monte-Carlo et leur efficacités relatives*, Université de Québec, Avril 2013.