Méthodes de Monte-Carlo

B. Ycart

UFR Mathématiques et Informatique
Université René Descartes, Paris
ycart@math-info.univ-paris5.fr

DEA Statistiques et modèles aléatoires
en économie et en finance

Université Denis Diderot
2000

Table des matières

1 Introduction

1.1 Ce qui ne sera pas traité

Bien que des milliers de publications soient parues sur le sujet, la dénomination "Méthodes de Monte-Carlo" reste assez imprécise dans la littérature, et varie selon les spécialités (mathématiques, recherche opérationnelle, automatique, physique, informatique). Chez certains auteurs, elle tend à englober tout ce qui a trait à l'utilisation du hasard dans les programmes informatiques, c'est-à-dire à la fonction Random. Ceci représente un champ beaucoup trop vaste pour le cadre de ce cours (le livre récent de Fishman [35] malgré ses 700 pages n'est pas exhaustif), et recouvre de fait des sujets assez distincts.

Commençons par une liste de domaines plus ou moins interconnectés, liés à l'utilisation du hasard sur ordinateur, que nous n'aborderons pas ou peu (chacun d'eux justifierait un cours complet en soi).

- **La construction des générateurs pseudo-aléatoires**, ou comment coder efficacement une fonction Random. C'est un problème que nous considèrerons arbitrairement comme résolu par Marsaglia et Zaman [61, 62], bien qu'une littérature importante continue à se développer sur la question. Quatre références de base sont les livres de Knuth [56], Dudewicz et Ralley [29], Fishman [35], Gentle [38]. L'article de Ripley [75] est une bonne introduction.
- **La simulation des variables aléatoires**, ou comment transformer un appel de Random (réalisation d'une variable aléatoire de loi uniforme sur $[0, 1]$) en une variable aléatoire de loi donnée. Ce sujet est abordé à niveau élémentaire dans de nombreux manuels, par exemple les livres de Bouleau [12], Snell [89] ou Berger [9]. Il est traité à fond par Devroye dans [26]. Nous nous contenterons de quelques indications sur les trois principes généraux que sont l'inversion, le rejet et la décomposition.
- **La simulation des processus stochastiques** ou comment transformer une suite de variables aléatoires indépendantes et de même loi (suite d'appels de Random) en un processus quelconque (martingale, chaîne ou processus de Markov, champ aléatoire, processus de diffusion ...). Plusieurs livres traitent de la simulation des processus, parmi lesquels celui de Bouleau et Lépingle [14]. Pour les processus de diffusion les livres de Kloeden et Platen [54, 55] sont la référence indispensable. Nous nous limiterons au schéma de discrétisation le plus simple, le schéma d'Euler-Maruyama.
- **La construction et la simulation de modèles stochastiques**, par exemple en recherche opérationnelle ou automatique (réseaux de files d'attente, systèmes à évènements discrets... : voir entre autres [80, 82, 84, 83, 86, 92, 96]). C'est un sujet suffisamment important pour avoir suscité le développement de langages de programmation spécialisés comme SIMULA ou plus récemment MODLINE et QNAP (sur les aspects algorithmiques voir aussi Watkins [94]). Nous ne traiterons pas non plus de l'utilisation du calcul stochastique en analyse financière, ni des utilisations de la simulation dans ce contexte (voir [46, 58, 69, 59]).

- **L'analyse probabiliste d'algorithmes**. Etudier la complexité d'un algorithme (déterministe) dans le pire ou le meilleur des cas, reflète rarement son comportement sur des données courantes. On a donc souvent recours à une analyse "en moyenne" où les données d'entrée de l'algorithme sont tirées au hasard, en un sens qui dépend du type de problème étudié. Sur cette question, plusieurs références de la littérature informatique sont accessibles au mathématicien appliqué, parmi lesquelles Graham *et al.* [41] ou le livre de Hofri [43], plus spécialisé.
- **Les algorithmes randomisés**. Ce sujet s'est développé ces dix dernières années sous l'impulsion d'informaticiens théoriciens. Parmi les problèmes dont la solution peut être programmée, on distingue ceux qui sont résolubles en un temps qui ne dépasse pas une certaine puissance de la taille du problème de ceux qui ne le sont pas (NP-complets). Pour certains de ces derniers, on a pu trouver des algorithmes de résolution approchée en temps polynômial, à base essentiellement de chaînes de Markov. Leur étude devient un domaine important de l'informatique théorique (voir Sinclair [88] ou Motwani et Raghavan [68]).
- **Les méthodes de Monte-Carlo en statistique**. De nombreuses questions d'estimation, de tests ou de représentation de données se ramènent à des problèmes numériques d'optimisation ou de résolution de systèmes de grande taille. Les méthodes qui seront présentées dans ce cours connaissent un grand succès auprès des statisticiens, qui en ont déduit des versions adaptées à leurs types de problème (échantillonnage de Gibbs, algorithmes EM, SEM... : voir Robert [78, 79] et McLachlan [64]). Il est très artificiel de couper, comme nous le ferons, les méthodes de résolution de problèmes déterministes de leurs applications naturelles en statistique. Duflo [30, 31] traite d'ailleurs sans distinction les deux types d'applications. Robert [77, 78, 79] montre bien l'importance et l'intérêt des méthodes de Monte-Carlo en statistique, en particulier bayésienne.
- **Les algorithmes de filtrage**. Il existe de nombreuses méthodes adaptées aux cas où les données du problème à traiter ne sont pas connues exactement, soit qu'elles proviennent d'un calcul numérique entaché d'erreurs importantes, soit qu'elles soient calculées à partir d'un échantillon statistique. Filtrage de Kalman, méthodes de fonctions splines [32] ou ondelettes [4], algorithmes de Robbins-Monro ou Kiefer-Wolfowitz, toute une panoplie de techniques permettent de traiter des données bruitées (voir Benvéniste *et al.* [8] pour une référence générale, et [95] pour le cas de l'analyse d'images). Là aussi notre séparation entre les méthodes où le hasard provient du modèle et celles où il est apporté par l'utilisation de la fonction Random est tout à fait artificielle. Les processus stochastiques sous-jacents sont essentiellement les mêmes, comme le montre bien Duflo [31] (voir aussi [57]).

1.2 Ce qui sera traité

Que reste-t-il donc ?

Nous désignerons surtout dans ce cours par *méthodes de Monte-Carlo* des algorithmes utilisant la fonction Random pour résoudre un problème numérique *déterministe* (calcul d'intégrale, résolution de système, résolution d'équation différentielle, optimisation).

Les méthodes numériques déterministes ayant connu le développement que l'on sait, il est légitime de se poser la question de l'intérêt d'en rajouter qui fassent appel au hasard, là où il n'a rien à faire a priori. L'objectif à atteindre étant la solution approchée d'un certain problème avec une précision ε fixée, un critère de choix important dans le choix d'une méthode numérique est le temps d'exécution, mesuré en nombre d'opérations élémentaires (la complexité de l'algorithme). Intervient alors la notion de taille du problème, qui est un entier n dont la signification est habituellement claire en fonction du contexte. C'est la dimension de l'espace pour une intégrale multiple, la taille de la matrice pour la résolution d'un système, etc... Les complexités des méthodes numériques déterministes dépendent habituellement de la taille de façon polynomiale, typiquement en $O(n^2)$ ou $O(n^3)$. Tout aussi typiquement, une méthode de Monte-Carlo résoudra le même problème en $O(n)$. Cela semble miraculeux : où est donc le piège ? Il est dans la constante cachée par l'expression $O(n)$. Cette constante dépend de la précision ε à atteindre. Pour une dimension donnée, il est fréquent qu'une méthode numérique atteigne la précision ε en un temps $O(\log(\varepsilon^{-1}))$ (en idéalisant, cela signifie que chaque nouveau pas d'itération fait passer la précision de 10^{-k} à $10^{-(k+1)}$). Pour une raison liée au théorème central limite, les méthodes de Monte-Carlo atteignent la précision ε en $O(\varepsilon^{-2})$, ce qui est très lent (typiquement 10^6 itérations pour une précision de 10^{-3}).

Déterministe	Monte-Carlo
$K \log(\varepsilon^{-1})\, n^2$	$K \varepsilon^{-2}\, n$

C'est donc dans des cas où la taille n du problème est très grande et l'exigence sur la précision faible (ε grand) qu'il faut envisager les méthodes de Monte-Carlo. Au début, pour des raisons de clarté, les exemples traités seront en basses dimensions. En pratique, on ne calcule pas une intégrale double ou triple, on ne résoud pas un système 2×2 (ni même 100×100) avec une méthode de Monte-Carlo. L'utilisation du hasard ne devient concurrentielle qu'à partir de dizaines de dimensions pour une intégrale ou de milliers d'équations pour un système. Il peut même se faire que la taille du problème soit telle que les méthodes déterministes échouent pour des raisons de place en mémoire. Par exemple, la taille maximale des systèmes linéaires que l'on peut résoudre actuellement est de l'ordre de 10^6 (voir Stewart [90]). Dans certains cas, une méthode de Monte-Carlo ira bien au-delà. Remarquons également que les deux approches peuvent être complémentaires. De nombreuses méthodes numériques demandent à être initialisées par une valeur déjà assez proche de la solution (par exemple Newton-Raphson). Une méthode de Monte-Carlo pourra fournir à peu de frais cette initialisation.

Pour résumer, on peut voir les méthodes de Monte-Carlo comme des méthodes relativement peu précises, mais faciles à implémenter, robustes, et destinées en priorité aux

problèmes de très grande taille. Signalons enfin que dans de nombreux cas les méthodes de Monte-Carlo se prêtent bien à la parallélisation, sujet que nous n'aborderons pas (voir Shonkwiler et Van Vleck [87], Del Corso [25] et Trouvé [93]).

1.3 Prérequis et plan

Ce cours a un objectif résolument pratique. Il se veut accessible à tout étudiant de DEA, ayant reçu une formation minimale en probabilités. Aucune connaissance particulière, autre qu'un peu de bon sens, ne sera supposée acquise. Les seules notions de probabilité qui seront utilisées sont la notion de suite de variables indépendantes (fonctions d'appels de Random successifs), le théorème central limite, qui est l'outil de base pour déterminer la précision d'un algorithme de Monte-Carlo, et les chaînes de Markov, vues comme des algorithmes itératifs où chaque nouveau pas est calculé en fonction du précédent et d'un nouveau tirage aléatoire. Les connaissances probabilistes correspondantes figurent dans tous les manuels classiques, qui vont en général bien au-delà (par exemple Berger [9], Bouleau [12, 13], Breiman [15], Feller [33, 34], Snell [89]...). Sur les chaînes de Markov plus particulièrement, on pourra se reporter au livre récent de Brémaud [16]. Les références de base restent Chung [22] et Kemeny et Snell [50]. Un point de vue plus appliqué est celui de Barucha-Reid [7] et Karlin et Taylor [47, 48]. Çinlar [23] est particulièrement clair. Les livres de Neuts [70, 71] proposent une vision systématiquement tournée vers l'outil informatique.

La deuxième partie traite des méthodes à tirages indépendants pour les calculs d'intégrales, exprimées comme des espérances de variables aléatoires. C'est à cette occasion que nous aborderons les générateurs pseudo-aléatoires et la simulation des lois de probabilité usuelles, pour en donner quelques principes de base. Les calculs d'intégrales par Monte-Carlo sont traités de manière plus ou moins détaillée dans de nombreux manuels, comme ceux de Kennedy et Gentle [51], Hammersley et Handscomb [42], Gentle [38], Morgan [67], Rubinstein [81], Ripley [74], Kleijnen [52, 53]. Sans surprises sur le plan théorique, elles fourniront surtout l'occasion de rappeler un certain nombre de bases probabilistes et algorithmiques, l'objectif principal étant de développer l'état d'esprit assez particulier qui préside à une implémentation efficace des méthodes de Monte-Carlo. Il s'agit en effet de s'habituer à considérer que la vitesse d'exécution de l'algorithme est ce qui conditionne avant tout la précision du résultat. On illustrera ce point de vue à l'aide de plusieurs astuces de programmation, habituellement regroupées sous l'appellation de "méthodes de réduction de la variance".

Au delà des calculs d'intégrales, les méthodes les plus répandues font appel à la simulation de chaînes de Markov. La troisième partie traite des méthodes markoviennes qui, comme dans la partie précédente, utilisent la loi des grands nombres pour calculer une espérance. Cette espérance est celle d'une variable aléatoire qui est fonction de la trajectoire d'une chaîne de Markov sur un intervalle de temps borné. Elle est approchée par une moyenne des valeurs prises par la variable sur un grand nombre de trajectoires indépendantes. L'application aux systèmes linéaires nous servira surtout à introduire les méthodes de résolution d'équations aux dérivées partielles. Comme référence de

base sur le sujet, nous utiliserons le livre de Lapeyre *et al.* [59].

La quatrième partie traite d'un autre type de méthodes markoviennes. Ces méthodes explorent un espace d'états soit de manière homogène pour approcher une mesure d'équilibre donnée (méthodes MCMC [78]), soit de manière dirigée à la recherche d'un point (extrémum d'une fonction [17]). Dans ce dernier cas, il s'agit de suivre une trajectoire d'une chaîne de Markov, qui visite avec une probabilité croissante un voisinage de la cible à atteindre. Parmi ces techniques, on peut ranger les méthodes neuronales, que nous n'aborderons pas (voir [3, 63, 76]). Nous traiterons surtout le recuit simulé [5, 10, 18] et décrirons l'heuristique des algorithmes génétiques [6, 19, 20, 21, 39, 44, 65, 66], et de l'algorithme MOSES [36]. Ces algorithmes peuvent être vus comme des méthodes de descente de gradient, "bruitées" afin d'éviter les pièges d'éventuels minima locaux. Les questions théoriques de convergence et de précision des méthodes d'exploration markovienne sont souvent très difficiles. Elles ont donné lieu à une intense activité de publication ces 15 dernières années (voir Saloff-Coste [85]). Nous n'aborderons ces questions que de manière assez superficielle dans le cadre des chaînes réversibles. Les deux livres de Duflo [30, 31] constituent une référence de base, d'un niveau sensiblement supérieur à celui de ce cours.

2 Méthodes à tirages indépendants

2.1 Générateurs pseudo-aléatoires

2.1.1 Postulats

Tous les langages (ou presque) disposent d'un générateur pseudo-aléatoire. Les syntaxes varient : `ran`, `gran`, `rand`, `Random`... Ce sont des fonctions, qui au dire des manuels d'utilisation, "retournent des nombres au hasard". Ce que l'on entend par "nombres au hasard" dépend d'abord du type des nombres (booléens, entiers, réels). Nous conviendrons de noter Random la fonction qui "retourne un réel au hasard dans $[0,1]$". Ceci recouvre en fait deux propriétés distinctes, que nous admettrons comme postulats.

Postulats

1. $\forall a, b\,,\ 0 \leq a < b \leq 1 \qquad Prob[\mathsf{Random} \in]a,b]\ ] = b - a.$
2. *Les appels successifs de* Random *sont des variables aléatoires indépendantes.*

En d'autres termes, on décide que la suite des appels de Random est une suite de variables aléatoires indépendantes, de loi uniforme sur $[0,1]$. Dans ce qui suit, la notation $Prob$ désigne la loi de cette suite, à savoir le produit indicé par $\mathbb{N}$ de copies de la mesure de Lebesgue sur $[0,1]$.

Interprétation : Dans le langage courant "au hasard" ne signifie pas seulement aléatoire mais en plus uniformément réparti. Choisir au hasard, c'est donner les mêmes chances à tous les résultats possibles (équiprobabilité). On attend d'un réel "au hasard" dans $[0,1]$ qu'il tombe entre 0.4 et 0.5 avec probabilité $1/10$, de même qu'entre 0.8 et 0.9. Deux intervalles inclus dans $[0,1]$ ont la même probabilité d'être atteints s'ils ont même longueur, et cette probabilité est la longueur des intervalles. Les postulats ci-dessus ont une autre conséquence. Si on considère des couples successifs d'appels de Random comme des coordonnées de points du plan, ces points sont des "points au hasard" dans $[0,1]^2$. Le sens précis étant que ces points sont indépendants et

$$\forall a, b\ ,\ 0 \leq a < b \leq 1\ ,\ \forall c, d\ ,\ 0 \leq c < d \leq 1$$

$$Prob[\ (\mathsf{Random}_1, \mathsf{Random}_2) \in]a,b]\times]c,d]\ ] = (b-a)(d-c)\ .$$

La probabilité qu'un point dont les deux coordonnées sont des appels de Random tombe dans un rectangle est la surface de ce rectangle. Ceci peut évidemment être étendu en dimension quelconque. Des triplets d'appels de Random successifs sont les coordonnées de "points au hasard" dans le cube $[0,1]^3$, etc...

Proposition 2.1 *Pour tout $k \in \mathbb{N}^*$, soient $(R_1, \ldots, R_k)$ k appels successifs de* Random*. Les postulats 1) et 2) entraînent que pour tout rectangle*

$$D =]a_1, b_1] \times \cdots \times]a_k, b_k], \quad 0 \leq a_i < b_i \leq 1\ , \quad i = 1, \ldots, k\ ,$$

$$Prob[\ (R_1, \ldots, R_k) \in D\ ] = (b_1 - a_1) \cdots (b_k - a_k)\ .$$

La probabilité pour Random de tomber sur n'importe quelle valeur particulière est nulle :

$$\forall a \in [0,1] \;\; Prob[\text{ Random} = a\ ] = 0,$$

car

$$\forall \varepsilon > 0\,,\; Prob[\text{ Random} = a\ ] \leq Prob[\text{ Random} \in]a-\varepsilon,\; a]\ ] = \varepsilon\ .$$

En conséquence,

$$\begin{aligned} Prob[\text{ Random } \in]a,b]\ ] = &\; Prob[\text{ Random} \in [a,b]\ ] \\ = &\; Prob[\text{ Random} \in [a,b[\ ] \\ = &\; Prob[\text{ Random} \in]a,b[\ ]\ . \end{aligned}$$

Il y a un paradoxe à admettre à la fois que Random peut prendre une infinité de valeurs distinctes et ne prend jamais aucune valeur particulière. Nous verrons en pratique la situation est légèrement différente, sans que cela remette en cause les postulats de définition de Random.

Une fois admis ces deux postulats, toute analyse d'algorithmes contenant Random est une démonstration mathématique qui ne doit contenir aucun à-peu-près.

Exemple 1 : (Int($\cdot$) désigne la partie entière).

$$X \longleftarrow (\text{Int}(\underbrace{\text{Random}}_{R_1} * 3\ )) * (\text{Int}(\underbrace{\text{Random}}_{R_2} * 2\ ))\ .$$

La variable aléatoire X prend les valeurs 0, 1 ou 2.

$$\begin{aligned} Prob[X=2] &= Prob[\text{ Int}(\ R_1 * 3\ ) = 2 \text{ et Int}(\ R_2 * 2\ ) = 1\ ] \\ &= Prob[\ R_1 \in [2/3,\ 1[\text{ et } R_2 \in [1/2,\ 1[\ ] \\ &= \left(1-\frac{2}{3}\right)\left(1-\frac{1}{2}\right) \\ &= \frac{1}{6}\ . \end{aligned}$$

On montre de même que $Prob[X=1] = 1/6$ et $Prob[X=0] = 4/6$.

Remarque : Contrairement aux apparences, l'algorithme ci-dessus ne retourne pas de valeurs entre 3 et 6. En théorie, la probabilité que Random retourne la valeur 1 est nulle. En pratique, les générateurs sont le plus souvent codés de manière à ne retourner ni 0 ni 1.

Exemple 2 : Lancer de dé.

$$D \longleftarrow \text{Int}(\text{Random} * 6) + 1$$

Pour tout $k \in \{1, \ldots, 6\}$,

$$\begin{aligned} Prob[D=k] &= Prob[\text{ Int}(\text{ Random} * 6\ ) = k-1\ ] \\ &= Prob[\text{Random} * 6 \in [k-1, k[\ ] \\ &= Prob[\text{Random} \in [(k-1)/6,\ k/6[\ ] \\ &= \frac{k}{6} - \frac{k-1}{6} = \frac{1}{6}\ . \end{aligned}$$

Il ne faut pas déduire de cet exemple que Int(Random $* k$) est la bonne manière de tirer un entier au hasard entre 0 et k–1. Les générateurs courants permettent différents types de sorties : réelles uniformes sur l'intervalle $[0, 1]$, mais aussi booléennes, entières, et même complexes.
Nous noterons Random$(\{x_1, \ldots, x_k\})$ la fonction qui retourne une valeur "au hasard" sur l'ensemble fini $\{x_1, \ldots, x_k\}$. Ses appels successifs sont des variables aléatoires indépendantes et le postulat 1) est remplacé par :

$$Prob[\text{Random}(\{x_1, \ldots, x_k\}) = x_i] = \frac{1}{k} , \quad \forall i = 1, \ldots, k .$$

Les générateurs permettent de simuler des suites d'expériences aléatoires indépendantes, et donc de calculer de façon approchée des probabilités par application de la loi des grands nombres.

$n_A \leftarrow 0$
Répéter n fois
    expérience
    Si A réalisé alors $n_A \leftarrow n_A + 1$
    finSi
finRépéter
$f_A \leftarrow n_A/n$.

La variable f_A est la fréquence expérimentale de l'évènement A au cours des n expériences.

Exemple :

$n_A \leftarrow 0$
Répéter n fois
    $D \leftarrow$ Random$(\{1, \ldots, 6\})$ *(Lancer d'un dé)*
    Si $D \geq 4$ alors $n_A \leftarrow n_A + 1$
    finSi
finRépéter
$f_A \leftarrow n_A/n$.

En sortie de cet algorithme, f_A contient un nombre d'autant plus proche de 0.5 que n est grand. Plus le nombre d'expériences est grand, plus précis est le résultat. Il est donc essentiel qu'une boucle de simulation soit la plus rapide possible. Il faut en éliminer toutes les opérations coûteuses ou inutiles.

2.1.2 Suites uniformes

Au vu de la suite de réels retournée par un générateur particulier, comment décider que ce générateur convient ? Le seul moyen pratique d'estimer une probabilité est de l'exprimer comme une limite de fréquences expérimentales.

Définition 2.2 *Une suite* (x_n), $n \in \mathbb{N}$, *à valeurs dans* $[0,1]$ *est dite* k*-uniforme si pour tout rectangle*

$$D =]a_1, b_1] \times \cdots \times]a_k, b_k], \quad 0 \leq a_i < b_i \leq 1 \ , \quad i = 1, \ldots, k \ ,$$

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n-1} \mathbb{1}_D((x_{ki}, x_{ki+1}, \ldots, x_{k(i+1)-1})) = (b_1 - a_1) \cdots (b_k - a_k) \ .$$

La notation $\mathbb{1}_D$ désigne la fonction indicatrice de l'ensemble D.

$$\mathbb{1}_D(y) = \begin{cases} 1 & \text{si } y \in D \, , \\ 0 & \text{sinon} \ . \end{cases}$$

Le vecteur $(x_{ki}, x_{ki+1}, \ldots, x_{k(i+1)-1})$ est le i-ième k-uplet d'éléments consécutifs de la suite. La somme $\sum_{i=0}^{n-1} \mathbb{1}_D((x_{ki}, x_{ki+1}, \ldots, x_{k(i+1)-1}))$ est le nombre de k-uplets d'éléments consécutifs de la suite qui appartiennent à D parmi les n premiers.

La définition ci-dessus dit donc que parmi les k-uplets d'éléments consécutifs de la suite, la proportion de ceux qui tombent dans un rectangle donné doit tendre vers le volume de ce rectangle (en dimension k). Cette définition est passablement utopique. En particulier elle entraîne que la probabilité que Random tombe sur un point donné est nulle. Comme qu'il n'y a qu'une quantité dénombrable de rationnels dans $[0,1]$, la probabilité de tomber sur un rationnel est également nulle. Or l'ordinateur ne connaît que les décimaux, et même seulement un nombre fini d'entre eux, donc la fonction Random ne peut retourner que des rationnels... Qu'à cela ne tienne, si on sait construire une suite uniforme de chiffres entre 0 et 9, on pourra en déduire des réels au hasard dans $[0,1]$, approchés à la k-ième décimale, en considérant des k-uplets consécutifs de chiffres de la suite initiale. C'est le principe des "tables de nombres au hasard" que l'on trouve encore dans certains livres. La même remarque vaut bien sûr en base 2. Il suffirait donc de savoir définir ce qu'est une suite de bits au hasard. Voici la définition de k-uniformité pour les suites de booléens.

Définition 2.3 *Une suite* $x = (x_n)$ *de booléens dans* $\{0,1\}$, *est dite* k*-uniforme si pour tout* $(\varepsilon_1, \ldots, \varepsilon_k) \in \{0,1\}^k$ *:*

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n-1} \mathbb{1}_{\{(\varepsilon_1,\ldots,\varepsilon_k)\}}((x_{ki}, x_{ki+1}, \ldots, x_{k(i+1)-1})) = \frac{1}{2^k} \ .$$

Ce que l'on attend en fait de la suite des appels de Random, c'est qu'elle soit k-uniforme, pour tout entier k (on dit ∞-uniforme). C'est évidemment illusoire, car l'infini n'existe pas pour un ordinateur. Tout ce que l'on peut faire, c'est vérifier que rien d'invraisemblable ne se produit pour la suite finie observée. C'est précisément l'objet des tests statistiques que de distinguer le plausible de ce qui est trop peu vraisemblable.

"Définition" *Un* $N-$*uplet de nombres dans* $[0,1]$ *sera dit pseudo-aléatoire s'il passe avec succès une série de tests statistiques, chacun étant destiné à vérifier une conséquence de la* k*-uniformité. Le nombre de ces tests ainsi que l'entier* k *sont fonction croissante de l'exigence de l'utilisateur.*

Des quantités de tests ont été imaginés pour mettre les générateurs à l'épreuve.

2.1.3 Implémentation

Parmi les différentes formalisations du hasard qui ont pu être proposées, la notion de suite ∞-uniforme est la seule utilisable en pratique : elle est la seule que l'on puisse tester pour un générateur donné et elle suffit à justifier toutes les applications des générateurs pseudo-aléatoires. Les générateurs conseillés ci-dessous, comme ceux qui sont livrés avec les compilateurs courants, sont issus d'une longue expérimentation statistique, et n'ont survécu au temps que parce qu'ils ont donné satisfaction à de nombreux utilisateurs, ce qui n'empêche pas de rester vigilant.

Comment sont-ils programmés ? Même les suites récurrentes les plus simples peuvent être chaotiques et donc constituer des exemples de suites apparemment aléatoires. Et ce, même si le sens intuitif de aléatoire (impossible à prévoir) est contradictoire avec la définition d'une suite récurrente (chaque terme fonction connue du précédent). Il n'est pas trop difficile d'engendrer des suites qui aient un comportement aléatoire. Pour un ordinateur, il n'existe qu'un nombre fini de valeurs. Les valeurs de Random sont toujours calculées à partir d'entiers répartis dans $\{0, 1, \ldots, M-1\}$ où M est un grand nombre (de l'ordre 10^8 au moins pour les générateurs usuels). Pour retourner un réel dans $[0, 1]$, il suffit de diviser par M. En pratique, un nombre fini de valeurs peuvent seules être atteintes, et elles le sont avec une probabilité positive. Ceci contredit les postulats de définition de Random mais ne constitue pas un inconvénient majeur dans la mesure où M est très grand. Pour obtenir un autre type de réalisations, comme par exemple des entiers uniformes sur $\{0, \ldots, k\}$ ou des booléens (pile ou face), passer par les appels de Random est inutile, il vaut mieux partir du générateur sur $\{0, \ldots, M-1\}$ sans diviser au préalable par M. Ceci est pris en compte par la plupart des langages courants.

La valeur retournée par un générateur est une fonction de la valeur précédente ou des valeurs obtenues précédemment. Dans le premier cas la suite calculée est une suite récurrente. Une "graine" u_0 étant choisie dans $\{0, \ldots, M-1\}$, les valeurs successives de la suite sont définies par $u_{n+1} = g(u_n)$, où g est une fonction de $\{0, \ldots, M-1\}$ dans lui-même, suffisamment simple pour être calculée rapidement.

On se heurte alors à un problème important : toute suite récurrente sur un ensemble fini est *périodique*. La suite de valeurs retournée par Random bouclera forcément. Peut-on considérer comme aléatoire une suite périodique ? Oui peut-être, si la période est suffisamment élevée par rapport au nombre de termes que l'on utilise. La prudence s'impose en tout cas et il faut se méfier de l'idée intuitive que plus le passage de u_n à u_{n+1} sera compliqué, plus leurs valeurs seront indépendantes et meilleur sera le générateur.

Les générateurs les plus simples sont les générateurs par congruence. Ils sont de la forme suivante :

$$g(u) \;=\; (Au + C) \text{ modulo } M \;.$$

Divers résultats mathématiques permettent de justifier les "bons" choix de A, C et M. Ils sont tombés en désuétude du fait de l'apparition de nouveaux générateurs plus performants. Le générateur suivant était très répandu et donnait en général satisfaction.

Il peut encore servir comme dépannage.

$$g(u) \;=\; 16807\,u \;\text{ modulo }\; 2147483647\,.$$

Tout générateur nécessite une initialisation. Pour une même valeur de la graine u_0, c'est la même suite de valeurs qui sera calculée à chaque fois. Pour obtenir des résultats différents d'une exécution à l'autre, il est nécessaire de changer la graine au début de chaque exécution. Selon les langages, cette initialisation peut être laissée au choix de l'utilisateur, ou être réalisée à partir du compteur de temps (fonction `randomize` en Turbo Pascal, `seed` en C...). **L'instruction de randomisation doit figurer une seule fois, en début de programme principal**. La solution de l'initialisation par le compteur de temps peut poser un problème sur les gros systèmes où l'horloge est d'accès réservé.

On trouve sur le réseau un ensemble de procédures proposées par Marsaglia et Zaman. Ces procédures sont présentées sous forme de fichier compressé, selon les systèmes d'exploitation (par exemple `FSULTRA1.ZIP` pour DOS). Une fois décompressé on obtient un ensemble de procédures en Assembleur, Pascal, C, Fortran qui implémentent le générateur ULTRA, dont les qualités sont très supérieures à celles des générateurs classiques. C'est ce générateur que nous conseillons d'utiliser dans sa version Assembleur, de préférence aux générateurs des langages courants.

La qualité d'une simulation est largement conditionnée par sa rapidité d'exécution. La manière de programmer joue donc un rôle essentiel. En particulier on prendra garde à évacuer de la boucle principale du programme toute opération inutile. De même il faut systématiquement chercher à remplacer les opérations coûteuses par d'autres plus rapides, même si cela donne un programme moins élégant à lire. Il est bon d'avoir en tête un ordre de grandeur des coûts relatifs de certaines opérations : affectations, tests, Random, additions en entier et en réel, multiplications en entiers et en réels, fonctions "chères" (exp, log, sqrt, cos...). Le problème est qu'il est difficile d'attribuer des coûts précis aux opérations de base, dans la mesure où ces coûts dépendent très fortement non seulement du processeur, mais aussi du langage, et même du compilateur utilisé pour ce langage. A titre d'exemple, voici des temps en secondes, mesurés sur un PC avec processeur 486DX33, pour des boucles de 10^7 itérations contenant chacune une opération élémentaire, écrites en TurboPascal. Ces résultats ne peuvent pas être pris comme des évaluations de durées pour des opérations booléennes, entières ou réelles quelconques. Il est impossible de savoir en général quelles optimisations locales un compilateur donné est capable de réaliser.

Le seul conseil que l'on puisse donner est de tester systématiquement les différentes options possibles sur des boucles de taille réduite, et de conserver la plus rapide pour le calcul en vraie grandeur. Un peu de bon sens et de pratique suffisent en général à éviter les erreurs les plus grossières.

version	4.0	4.0	5.5	5.5	7.0	7.0
coprocesseur	sans	avec	sans	avec	sans	avec
Boucle vide	2.69	2.69	2.69	2.70	2.69	3.30
B :=true	3.30	3.29	3.30	3.29	3.90	3.29
K :=1234	3.90	3.30	3.29	3.30	3.29	3.30
X :=1.234	6.32	32.73	4.51	4.50	4.51	4.50
B :=(1234<2345)	3.29	3.90	3.29	3.30	3.29	3.30
B :=(1.234<2.345)	22.52	18.90	3.30	3.29	3.30	3.29
K :=1234+2345	3.30	3.29	3.35	3.35	3.29	3.90
K :=1234−2345	3.29	3.90	3.30	3.90	3.30	3.30
X :=1.234+2.345	41.36	38.40	5.05	4.45	4.50	4.50
X :=1.234−2.345	52.23	38.99	4.50	4.50	4.51	4.51
K :=123*234	3.30	3.30	3.30	3.30	3.29	3.29
X :=1234/2345	367.12	53.06	5.06	4.50	4.51	5.11
X :=1.234*2.345	292.75	38.39	4.50	4.51	4.50	4.51
X :=1.234/2.345	364.76	53.44	5.11	5.11	4.50	4.50
X :=Random	45.59	82.61	43.33	79.64	42.90	82.33
X :=sqrt(1.234)	2233.16	80.41	2302.15	55.20	2172.25	55.48

2.1.4 Complexité et hasard

Parmi les différentes formalisations du hasard qui ont pu être proposées, la notion de suite uniforme est la seule utilisable en pratique : elle est la seule que l'on puisse tester pour un générateur donné et elle suffit à justifier toutes les applications de la fonction Random. Une "vraie" suite aléatoire doit être uniforme. Mais peut-on considérer comme aléatoire toute suite uniforme ? Nous allons voir que non, malheureusement.

Comme l'ordinateur ne peut donner que des approximations décimales des réels, il suffirait de savoir construire une suite aléatoire de chiffres entre 0 et 9, pour en déduire des réels au hasard dans $[0, 1]$, approchés à la k-ième décimale, en considérant des k-uplets consécutifs de chiffres de la suite initiale. La même remarque vaut bien sûr en base 2. Il suffirait donc de savoir définir ce qu'est une suite de bits au hasard.

Limitons-nous donc aux suites de bits dans $\{0, 1\}$. On attend d'une telle suite qu'elle soit le résultat typique d'une suite de tirages de Pile ou Face. Voici trois séquences particulières :

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1

0 1 1 0 1 0 0 0 1 1 0 1 1 1 0 0

La troisième a meilleur aspect que les deux premières. Elle a pourtant exactement la même probabilité de sortir telle quelle à Pile ou Face : $1/2^{16}$. Parmi les 1000-uplets de bits, tous ont a priori la même probabilité de sortir : $1/2^{1000}$. Toute suite aléatoire de bits doit contenir nécessairement une infinité de fois 1000 zéros à la suite. Accepterions-nous qu'un générateur de bits retourne ne serait-ce que 100 zéros à la suite ? Cela paraît peu vraisemblable.

Sur les 3 exemples ci-dessus, nous écarterions le premier car il semble violer la 1-uniformité. Nous écarterions le second au nom de la 2-uniformité. Mais que dire alors de la suite concaténée de tous les entiers en base 2 (suite de Champernowne) ?

$$0\ 1\underbrace{1\ 0}_{2}\underbrace{1\ 1}_{3}\underbrace{1\ 0\ 0}_{4}\underbrace{1\ 0\ 1}_{5}\underbrace{1\ 1\ 0}_{6}\underbrace{1\ 1\ 1}_{7}\underbrace{1\ 0\ 0\ 0}_{8}\underbrace{1\ 0\ 0\ 1}_{9}\underbrace{1\ 0\ 1\ 0}_{10}\underbrace{1\ 0\ 1\ 1}_{11}\underbrace{1\ 1\ 0\ 0}_{12}\underbrace{1\ 1\ 0\ 1}_{13}\ \ldots$$

On montre qu'elle est ∞-uniforme. Mais comment qualifier d'aléatoire une suite aussi parfaitement prévisible ?

La définition la plus intuitive du hasard, celle des dictionnaires, n'a aucun rapport avec l'uniformité. Est aléatoire ce qui est imprévisible. Une suite serait donc aléatoire si on ne pouvait pas prévoir son $(n+1)$-ième terme connaissant les n premiers. La suite

$$0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ \ldots\ 0\ 1\ \ldots$$

n'est pas aléatoire car sans avoir écrit les 999 premiers termes on sait que le 1000-ième sera "1" et le suivant "0". La suite

$$0\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 0\ 0\ 1\ 0\ 1$$

en revanche, ne semble pas présenter de régularité, de configuration qui permettrait d'en prévoir les termes suivants. Une suite est donc aléatoire si elle n'a pas de règle de construction simple.

On peut voir une règle de construction comme un moyen de compresser une suite en un *algorithme* qui l'engendre.

Exemple :

Répéter n fois
    écrire 0 puis 1
finRépéter

Traduit en chaîne de bits, la seule chose qui dépend de n dans cet algorithme est le test. La longueur de la chaîne de bits sera $\log_2 n$+constante, pour engendrer à l'exécution une chaîne de longueur $2n$ ($\log_2 n$ est le nombre de bits nécessaires pour écrire n, à 1 près).

Notons X^* l'ensemble de toutes les chaînes de bits de longueur finie. Si $x \in X^*$, sa longueur (nombre de bits) sera notée $l(x)$. Un algorithme est une application de X^* dans lui-même.

$$\begin{array}{ccc} X^* & \stackrel{\Phi}{\longrightarrow} & X^* \\ y & \longrightarrow & x \\ \text{input} & & \text{output} \end{array}$$

Soit $\tau(\phi)$ la taille de ϕ exprimée en bits (hors input).

Définition 2.4 *On appelle complexité de x relative à ϕ l'entier :*

$$K_\phi(x) = \inf\ \{\ l(y),\ \phi(y) = x\ \} + \tau(\phi)\ \ (= +\infty\ si\ x \notin \phi(X^*))\ .$$

En d'autres termes $K_\phi(x)$ est une mesure de la taille de l'information qu'il faut donner à l'algorithme ϕ pour produire x. Dans l'exemple ci-dessus, une chaîne de $\log_2 n$ bits suffisait à produire la chaîne de longueur $2n$

$$\underbrace{0\ 1\ 0\ 1\ \cdots\cdots\ 0\ 1}_{n \text{ fois}} .$$

Définition 2.5 *On appelle complexité de x l'entier :*

$$K(x) = \inf_{\phi} K_\phi(x) .$$

$K(x)$ est la quantité minimale d'information nécessaire pour produire x.

Remarque : Bien que ceci n'ait pas la prétention d'être un cours de complexité, on doit tout de même signaler que les définitions ci-dessus n'ont de sens que si on restreint quelque peu la notion d'algorithme. Si on en reste aux applications quelconques de X^* dans X^*, on se heurte à des paradoxes du type de celui de R. Berry :

"Le plus petit nombre qu'on ne puisse pas définir en moins de 20 mots".

Le bon cadre est celui des fonctions récursives, calculables par machines de Turing.

Pour engendrer une suite donnée de n bits, l'algorithme brutal consiste à les écrire tous les uns après les autres. Il correspond à l'application identique I de X^* dans lui-même. Relativement à cet algorithme la complexité de toute suite de n bits est n. Donc la complexité de toute suite est majorée par sa longueur :

$$\forall x \quad K_I(x) = l(x) \Longrightarrow K(x) \leq l(x) .$$

Dire qu'une suite est aléatoire, c'est dire qu'on ne peut pas faire mieux que l'algorithme brutal pour l'engendrer.

Définition 2.6 *Soit $x = (x_n)_{n \in I\!N^*}$ une suite de bits. Elle est dite aléatoire si il existe une constante c telle que pour tout n :*

$$\exists c \quad \forall n \quad K((x_1, \ldots, x_n)) \geq n - c .$$

D'après la proposition suivante, la plupart des suites sont aléatoires.

Proposition 2.7 *Le cardinal de l'ensemble des suites de bits de longueur n dont la complexité est minorée par $n-c$ est au moins :*

$$2^n(1 - 2^{-c}) .$$

Démonstration : Parmi les inputs susceptibles d'engendrer les suites de longueur n, seuls ceux dont la longueur est inférieure à $(n - c)$ nous intéressent. Il y en a au plus :

$$2^0 + 2^1 + \cdots + 2^{n-c-1} = 2^{n-c} - 1 .$$

Chaque couple (input+algorithme) engendre au plus une suite de longueur n. Il y a donc au plus 2^{n-c} suites de longueur n dont la complexité est inférieure à $n - c$.
□

Concrètement, parmi toutes les suites de longueur n, une proportion de $1/2^{10} \simeq 10^{-3}$ d'entre elles seulement sont de complexité $< n - 10$.

Il devrait donc être facile de trouver des suites aléatoires puisque la plupart d'entre elles le sont. Paradoxalement, le problème de démontrer qu'une suite particulière est aléatoire est indécidable en général.

2.2 Théorème central limite

2.2.1 Enoncé

En simulation, la situation typique est celle où on exécute un très grand nombre de fois une boucle, en calculant à chaque passage des réalisations de variables aléatoires indépendantes. Le résultat recherché est une espérance, que l'on estime par la moyenne empirique des variables simulées. Pas plus en simulation qu'en physique ou en biologie on ne donnera un résultat sans indication sur sa précision. C'est le théorème central limite qui permet de calculer cette précision.

Théorème 2.8 *Soit $(X_n), n \in \mathbb{N}^*$ une suite de variables aléatoires indépendantes de même loi, d'espérance μ et de variance σ^2 finies. Posons*

$$\forall n \in \mathbb{N}^* \ , \quad \overline{X}_n = \frac{X_1 + \cdots + X_n}{n} \quad et \quad Z_n = \frac{\sqrt{n}}{\sigma}\left(\overline{X}_n - \mu\right) \ .$$

La suite $(Z_n)_{n\in\mathbb{N}^*}$ converge en loi vers la loi normale $\mathcal{N}(0,1)$, c'est-à-dire :

$$\forall a, b \ -\infty \leq a < b \leq +\infty \qquad \lim_{n\to\infty} Prob[a < Z_n < b] \ = \ \int_a^b \frac{1}{\sqrt{2\pi}} e^{-x^2/2}\, dx \ .$$

Interprétation La fonction $\frac{1}{\sqrt{2\pi}} e^{-x^2/2}$ joue un rôle extrêmement important en probabilités (courbe de Gauss). C'est une fonction paire, qui décroît rapidement à l'infini. Voici quelques valeurs pour :

$$g(a) \ = \ \int_{-a}^{a} \frac{1}{\sqrt{2\pi}} e^{-x^2/2}\, dx \ .$$

a	1	1.96	2	2.5758	3	4	$+\infty$
$g(a)$	0.6826	0.95	0.9544	0.99	0.9973	0.999994	1

Dans le théorème central limite, μ est la valeur à estimer. Les n valeurs $X_1, \ldots, X_n$ constituent un échantillon de mesures aléatoires indépendantes d'espérance μ. La quantité $(X_1 + \cdots + X_n)/n$ est la moyenne empirique de l'échantillon, qui d'après la loi des grands nombres doit converger vers l'espérance μ. Le théorème central limite donne la précision de cette approximation. Il faut le lire intuitivement comme suit. Si n est assez grand alors Z_n est très probablement compris entre -3 et 3. Soit encore :

$$\frac{X_1 + \cdots + X_n}{n} - \mu \ \in \ \left[-\frac{3\sigma}{\sqrt{n}} \ ; \ +\frac{3\sigma}{\sqrt{n}}\right] \ ,$$

ou bien $\overline{X}_n$ (moyenne empirique) est égale à μ à $3\sigma/\sqrt{n}$ près. On formalise ceci par la notion d'intervalle de confiance.

Le théorème central limite est utilisé pour des valeurs finies de n. L'idée concrète est la suivante. Si n est assez grand, la variable centrée réduite (espérance 0, variance 1) Z_n associée à la somme de n variables indépendantes suit approximativement la loi $\mathcal{N}(0,1)$. Si on réalise suffisamment de simulations de Z_n et que l'on trace un histogramme des valeurs obtenues, celui-ci ne sera pas très loin de la courbe $\frac{1}{\sqrt{2\pi}}e^{-x^2/2}$. Pas plus loin en tout cas que si on avait simulé des variables aléatoires de loi $\mathcal{N}(0,1)$. Si Z suit la loi $\mathcal{N}(0,1)$, alors $Y = \sigma Z + \mu$ suit la loi $\mathcal{N}(\mu, \sigma^2)$. On peut aussi dire que pour n assez grand une somme de n variables aléatoires indépendantes suit approximativement une loi normale, dont l'espérance et la variance sont respectivement la somme des espérances et la somme des variances des variables que l'on ajoute. Le problème est de savoir à partir de quelle valeur n est "assez grand", pour la précision désirée. Cela dépend beaucoup de la loi des X_n. L'approximation est d'autant meilleure que la loi des X_n est plus symétrique. En particulier, le bon comportement de la loi uniforme vis à vis du théorème central limite conduit à un algorithme approché de simulation pour la loi $\mathcal{N}(0,1)$, qui peut être plus rapide que l'algorithme polaire que nous verrons plus loin.

$X \longleftarrow -6$
Répéter 12 fois
    $X \longleftarrow X+$Random
finRépéter

Justification Si (R_n) désigne la suite des appels de Random (suite de variables indépendantes de loi uniforme sur $[0,1]$), on a :

$$\frac{R_1 + \cdots + R_n - n/2}{\sqrt{n}\sqrt{\frac{1}{12}}} \xrightarrow{\mathcal{L}} \mathcal{N}(0,1) \ .$$

On évite une division et on obtient une approximation déjà correcte en prenant $n = 12$. Cet algorithme n'est cependant pas conseillé avec les générateurs classiques. Son principal inconvénient est de consommer trop d'appels de Random, ce qui pose le problème de la dépendance des réalisations successives.

Pour des lois plus dissymétriques comme la loi exponentielle, l'approximation normale n'est pas valable pour des sommes de quelques dizaines de variables. On peut la considérer comme justifiée à partir de quelques centaines. En simulation, ce sont des milliers, voire des millions de variables qui sont engendrées, et l'approximation normale est tout à fait légitime.

2.2.2 Intervalles de confiance

L'idée de l'estimation par intervalle de confiance est de définir, autour de la moyenne empirique, un intervalle aléatoire (dépendant des n expériences) qui contienne μ avec une forte probabilité. C'est l'amplitude de cet intervalle qui mesure la précision de l'estimation.

Théorème 2.9 *Soit $(X_n), n \in \mathbb{N}^*$ une suite de variables aléatoires indépendantes de même loi, d'espérance μ et variance σ^2 finies. Posons :*

$$\forall n \in \mathbb{N}\,, \quad \overline{X}_n = \frac{X_1 + \cdots + X_n}{n} \quad et \quad S_n^2 = \frac{X_1^2 + \cdots + X_n^2}{n} - \overline{X}_n^2 \,.$$

Soit α un réel > 0 (petit). Soit z_α le réel > 0 tel que :

$$\int_{-z_\alpha}^{+z_\alpha} \frac{1}{\sqrt{2\pi}} e^{-x^2/2}\, dx = 1 - \alpha \,.$$

Posons

$$\begin{aligned} T_1 &= \overline{X}_n - \frac{z_\alpha \sigma}{\sqrt{n}} \quad , \qquad T_1' &= \overline{X}_n - \frac{z_\alpha \sqrt{S_n^2}}{\sqrt{n}} \,, \\ T_2 &= \overline{X}_n + \frac{z_\alpha \sigma}{\sqrt{n}} \quad , \qquad T_2' &= \overline{X}_n + \frac{z_\alpha \sqrt{S_n^2}}{\sqrt{n}} \,. \end{aligned}$$

Alors :

$$\lim_{n\to\infty} Prob[\ \mu \in [T_1, T_2]\ ] \ = \ \lim_{n\to\infty} Prob[\ \mu \in [T_1', T_2']\ ] \ = \ 1 - \alpha \quad .$$

On dit que les intervalles aléatoires $[T_1, T_2]$ et $[T_1', T_2']$ sont des intervalles de confiance pour μ, de niveau de confiance asymptotique $1 - \alpha$.

Interprétation La valeur μ étant inconnue, il n'y a pas de raison a priori pour que l'écart-type σ soit connu. S'il est inconnu, on l'estime par l'écart-type empirique $\sqrt{S_n^2}$. C'est la raison pour laquelle nous donnons deux intervalles de confiance. La valeur de z_α est lue dans une table, ou retournée par un module de calcul numérique. Les valeurs les plus courantes sont les suivantes :

α	0.01	0.02	0.05
z_α	2.5758	2.3263	1.96

Les intervalles $[T_1, T_2]$ et $[T_1', T_2']$ sont aléatoires. A l'issue de la série de n expériences, T_1 et T_2 auront pris des valeurs particulières t_1 et t_2. On ne pourra pas dire qu'il y a une probabilité $1 - \alpha$ pour que μ appartienne à $[t_1, t_2]$. Aussi bien μ que t_1 et t_2 sont des réels fixés et le résultat $\mu \in [t_1, t_2]$ sera soit vrai soit faux mais ne dépendra plus du hasard. Ce qu'on pourra dire, c'est que cet encadrement est obtenu à l'issue d'une expérience qui avait un fort pourcentage de chances de donner de bons résultats. Pour $\alpha = 0.01$, si on répète 100 fois la série de n expériences pour obtenir 100 intervalles, on peut s'attendre à ce que l'un d'entre eux soit faux.

Il faut comprendre un intervalle de confiance comme une précision donnée sur la valeur estimée de μ :

$$\mu = \overline{X}_n \pm \frac{z_\alpha \sigma}{\sqrt{n}} \qquad \text{ou} \qquad \mu = \overline{X}_n \pm \frac{z_\alpha \sqrt{S_n^2}}{\sqrt{n}} \,.$$

Selon le niveau de confiance, z_α varie en gros entre 2 et 3. Seulement deux facteurs influent vraiment sur la précision : le nombre d'expériences n et la variance σ^2. Pour ce qui est du nombre d'expériences, la précision est de l'ordre de $n^{-1/2}$. C'est une "mauvaise" précision, mais on n'y peut rien. En revanche, on aura intérêt à tenter le plus possible de réduire σ. C'est l'origine de l'expression "*méthodes de réduction de la variance*". Malheureusement la réduction de la variance s'accompagne en général d'une augmentation du temps d'exécution de l'algorithme. Si en divisant l'écart-type par 2 on double en même temps le temps d'exécution, on n'aura rien gagné. Plus que la variance, le critère d'évaluation à retenir pour un calcul par simulation est la précision atteinte pour un temps d'exécution donné (mesuré évidemment sur la même machine, avec le même compilateur).

Le calcul d'un intervalle de confiance s'effectue grâce à des variables cumulantes dans la boucle de simulation.

Somme $\longleftarrow 0$
Somme2 $\longleftarrow 0$
Répéter n fois
    Expérience
    calcul de X
    Somme $\longleftarrow$ Somme $+X$
    Somme2 $\longleftarrow$ Somme2 $+X * X$
finRépéter
Moyenne $\longleftarrow$ Somme $/n$
Variance $\longleftarrow$ Somme2 $/n-$ moyenne$*$moyenne
Amplitude $\longleftarrow z_\alpha$*Sqrt(Variance$/n$)
$T_1 \longleftarrow$ Moyenne $-$ Amplitude
$T_2 \longleftarrow$ Moyenne $+$ Amplitude

Nous avons supposé jusque-là que les valeurs après chaque opération sont "exactes". En pratique, le nombre d'itérations est souvent très grand. Une somme de 10^6 termes peut devenir très imprécise si on ne prend pas certaines précautions, comme de déclarer la variable cumulante en double précision, voire de découper la boucle principale en deux boucles emboîtées.

2.2.3 Estimation d'une probabilité

Supposons que la quantité à estimer soit la probabilité p d'un évènement. On réalise une suite d'expériences indépendantes, en notant à chaque fois si l'évènement est réalisé (1) ou non (0). La variable aléatoire correspondant à la n-ième expérience est notée X_n. Les X_n suivent la loi de Bernoulli de paramètre p.

$$Prob[X_n = 0] = 1-p\ , \quad Prob[X_n = 1] = p\ .$$

$$\mu = I\!E[X_n] = p \qquad \text{et} \qquad \sigma^2 = \mathrm{Var}[X_n] = p(1-p)\ .$$

La somme $X_1 + \cdots + X_n$ (nombre de réalisations de l'évènement sur n expériences) suit la loi binomiale $\mathcal{B}(n,p)$. La moyenne empirique $\overline{X}_n$ de l'échantillon est ici la fréquence

expérimentale de l'évènement. La variance empirique S_n^2 est égale à $\overline{X}_n(1-\overline{X}_n)$. Il est à remarquer ici que la variance et la variance empirique sont majorées par $1/4$.

Exemple : l'aiguille de Buffon.

On lance au hasard une aiguille sur un parquet. On supposera pour simplifier que la longueur de l'aiguille est égale à la largeur d'une lame de parquet. Le problème consiste à calculer la probabilité pour que l'aiguille tombe à cheval sur 2 lames de parquet.

Une concrétisation de cette expérience se trouve au palais de la découverte : les "lames de parquet" sont métalliques, l'aiguille est retenue par un électro-aimant et tombe quand le visiteur appuie sur un interrupteur. Si elle tombe à cheval sur deux lames, il y a contact et un compteur est incrémenté. On peut donc calculer la fréquence expérimentale. Celle-ci est remarquablement proche de $2/\pi$ (des millions de visiteurs ont appuyé sur le bouton...). On a donc un moyen "expérimental" de calculer π. Notons que l'expérience du palais de la découverte est déjà une analogie, une idéalisation du problème initial : c'est un modèle physique.

Modèle mathématique.
Les hypothèses sont les suivantes.

- La position du milieu de l'aiguille est un réel au hasard entre 0 et $1/2$.
- L'angle de l'aiguille avec l'axe vertical est un réel au hasard entre 0 et $\pi/2$.
- Ces deux variables aléatoires sont indépendantes.

Comme conséquence du modèle mathématique, on peut démontrer que la probabilité cherchée vaut $2/\pi$.

Calcul par simulation

$n_A \leftarrow 0$
Répéter n fois
    $X \longleftarrow$ Random$/2$
    $\theta \longleftarrow$ Random $*\pi/2$
    Si $\cos(\theta) \geq 1 - 2X$ Alors $n_A \longleftarrow n_A + 1$
    finSi
finRépéter
$\overline{X}_n \leftarrow n_A/n$

A l'issue de n expériences, on obtient une valeur de la fréquence expérimentale n_A/n et donc un intervalle de confiance $[T_1, T_2]$. L'intervalle $[2/T_2, 2/T_1]$ est aussi un intervalle de confiance pour la valeur π. Voici par exemple des résultats obtenus sur $n = 10^6$ expériences.

$$\overline{X}_n = 0.636438\ ,\ \frac{2}{\overline{X}_n} = 3.14249\ .$$

Pour $1-\alpha = 0.99$ ($z_\alpha = 2.5758$) et $1-\alpha = 0.95$ ($z_\alpha = 1.96$), les intervalles de confiance pour la probabilité sont respectivement :

$$[0.6352\ ;\ 0.6377]\ \text{ et }\ [0.6355\ ;\ 0.6374]\ .$$

Ils correspondent aux encadrements suivants pour la valeur de π :

$$[3.1364\ ;\ 3.1486] \text{ et } [3.1378\ ;\ 3.14715]\ .$$

Dans les deux cas (calcul mathématique et simulation) on n'a fait que développer les conséquences des hypothèses de définition du modèle. La simulation n'a pas plus de rapport avec la réalité physique que le calcul mathématique. D'ailleurs, on est obligé d'introduire la valeur de π dans l'algorithme pour au bout du compte...en déduire une estimation de cette valeur ! Le miracle est que les conséquences calculées des hypothèses de modélisation puissent avoir un rapport avec une réalité physique, ou en d'autres termes que le modèle mathématique puisse être *validé* par confrontation avec l'expérience.

2.3 Inversion

2.3.1 Principe

La méthode d'inversion est la plus simple des méthodes générales de simulation. Elle consiste à composer un appel de Random avec l'inverse de la fonction de répartition de la loi à simuler. Soit F cette fonction de répartition. C'est une fonction de $\mathbb{R}$ dans $[0,1]$, croissante au sens large et continue à droite. Nous convenons de définir son inverse de la façon suivante.

$$\forall u \in [0,1]\ ,\ F^{-1}(u) \ =\ \inf\ \{x\ ;\ F(x) \geq u\}\ .$$

Proposition 2.10 *Soit F une fonction de répartition sur $\mathbb{R}$ et U une variable aléatoire de loi uniforme sur $[0,1]$. La variable aléatoire $X = F^{-1}(U)$ a pour fonction de répartition F.*

Démonstration :

$$\begin{aligned} \forall x \in \mathbb{R}\ , \quad Prob[X \leq x] = &\ Prob[\ \inf\ \{y\ ;\ F(y) \geq U\} \leq x\ ] \\ = &\ Prob[\ U \leq F(x)\ ] \\ = &\ F(x)\ . \end{aligned}$$

□

Exemple : Loi exponentielle de paramètre λ.

$$F(x) = (1 - e^{-\lambda x})\, \mathbb{1}_{\mathbb{R}^+}(x)\ .$$

$$\forall u \in]0,1], \quad F(x) = u \Longleftrightarrow x = -\frac{1}{\lambda}\log(1-u)\ .$$

D'où l'algorithme de simulation :

$X \longleftarrow -\log(\mathsf{Random})/\lambda\ .$

(Il est inutile de calculer $-\log(1-\mathsf{Random})/\lambda$ car Random et $1-$Random suivent la même loi).

La méthode d'inversion n'est exacte qu'à condition de connaître l'expression explicite de F^{-1}, comme pour la loi exponentielle. C'est rarement le cas. Si on veut appliquer la méthode à la loi normale par exemple, il faudra se donner une table de valeurs de F et procéder par interpolation linéaire. On simulera alors une loi dont la fonction de répartition, linéaire par morceaux, n'est qu'une approximation de la vraie fonction de répartition. En plus de l'imprécision, cette méthode présente deux autres inconvénients. L'un est l'encombrement de la place mémoire, l'autre est la lenteur due au nombre élevé de tests, même avec une recherche dichotomique. Même quand on connaît explicitement F^{-1}, la méthode d'inversion est rarement la plus efficace pour les variables à densité.

2.3.2 Lois discrètes

Un choix aléatoire dans un ensemble fini ou dénombrable peut toujours se ramener à la simulation d'une loi de probabilité sur $\mathbb{N}$ (il suffit de numéroter les éventualités). Nous supposons d'abord que les éventualités sont des réels rangés par ordre croissant.

$$\{x_i\,;\ i \in \{1,\ldots,n\} \text{ ou } i \in \mathbb{N},\ ,\ x_i < x_{i+1}\ ,\ \forall i\}\ .$$

Considérons la loi qui charge la valeur x_i avec probabilité p_i $(i \geq 1)$. La fonction de répartition correspondante est définie par :

$$F(x) = \begin{cases} 0 & \text{si } x < x_1 \\ p_1 + \cdots + p_i = F_i & \text{si } x_i \leq x < x_{i+1}\ . \end{cases}$$

L'algorithme de simulation par inversion est l'algorithme naturel de choix entre différentes éventualités.

$i \longleftarrow 1$
choix$\longleftarrow$Random
TantQue (choix$> F_i$) faire
    $i \longleftarrow i+1$
finTantQue
$X \longleftarrow x_i$

Il est inutile de recalculer les sommes $p_1 + \cdots + p_i$ à chaque passage dans la boucle. De même, le nombre de tests de l'algorithme ci-dessus valant i avec probabilité p_i, on aura intérêt à ranger les éventualités par ordre de probabilités décroissantes. Si le nombre de valeurs est important, il vaudra mieux utiliser un algorithme de recherche dichotomique, qui sera plus rapide que la recherche séquentielle.

Exemple : simulation de la loi de Poisson

Si X suit la loi de Poisson de paramètre λ on a :

$$Prob[X = n] = e^{-\lambda}\lambda^n/n! = \frac{\lambda}{n} Prob[X = n-1]\ .$$

Il n'y a pas d'expression simple pour la fonction de répartition et l'ensemble des valeurs possibles est infini. Il faut donc calculer les valeurs F_i au fur et à mesure. L'algorithme est le suivant.

$P \longleftarrow e^{-\lambda}$
$F \longleftarrow P$
$X \longleftarrow 0$
choix $\longleftarrow$ Random
TantQue (choix $> F$)
    $X \longleftarrow X + 1$
    $P \longleftarrow P * \lambda / X$
    $F \longleftarrow F + P$
finTantQue

Mais simuler une seule valeur n'a aucun sens. L'algorithme ci-dessus sera sans doute appelé un grand nombre de fois et recalculera à chaque fois les mêmes valeurs de la fonction de répartition. Voici par exemple les 6 premières valeurs de cette fonction pour $\lambda = 1$.

i	0	1	2	3	4	5
F_i	$0,3679$	$0,7358$	$0,9193$	$0,9810$	$0,9963$	$0,9994$

On a tout intérêt à mettre dans un tableau en début de programme ces 6 valeurs, quitte à calculer les suivantes le cas échéant. Dans environ 9994 cas sur 10000, les 6 valeurs précalculées suffiront.

En début de programme :

Tableau des $F[i]$, $i = 0 \cdots MAX$
$FMAX \longleftarrow F[MAX]$
$PMAX \longleftarrow F[MAX] - F[MAX - 1]$

Dans la boucle principale :

choix$\longleftarrow$Random
Si (choix $\leq FMAX$)
    alors
        $X \longleftarrow 0$
        TantQue (choix $> F[X]$)
            $X \longleftarrow X + 1$
        finTantQue
    Sinon
        $X \longleftarrow MAX$
        $P \longleftarrow PMAX$
        $F \longleftarrow FMAX$
        TantQue (choix $> F$)
            $X \longleftarrow X + 1$
            $P \longleftarrow P * \lambda / X$
            $F \longleftarrow F + P$

finTantQue

finSi

Cet exemple peut être étendu à n'importe quelle loi discrète. Si on l'utilise convenablement la méthode d'inversion est une excellente méthode de simulation des lois discrètes, en particulier quand un faible nombre de valeurs ont une probabilité cumulée proche de 1. L'utiliser convenablement signifie :

- éliminer au maximum les calculs répétés (comme ci-dessus).
- classer les valeurs de la fonction de répartition par ordre décroissant si on utilise une recherche séquentielle ou bien utiliser une recherche par dichotomie.

Histogrammes : Modifions légèrement l'algorithme de choix aléatoire entre k éventualités $\{x_1, \ldots, x_k\}$, en lui rajoutant une interpolation linéaire. Quand `choix` tombe dans l'intervalle $]F_{i-1}, F_i]$, au lieu de retourner x_i comme précédemment, nous retournons :

$$x_{i-1} + (x_i - x_{i-1}) * \frac{\texttt{choix} - F_{i-1}}{F_i - F_{i-1}} .$$

Ceci revient à remplacer la fonction de répartition en escalier par une fonction de répartition linéaire par morceaux, passant par les points (x_i, F_i). La distribution de probabilité correspondante admet pour densité une fonction en escalier (constante sur les intervalles $]x_{i-1}, x_i[$). C'est un histogramme. Simuler une loi de probabilité par inversion à partir d'une table de valeurs de la fonction de répartition, comme évoqué plus haut, revient à remplacer sa densité par une discrétisation en une fonction en escalier, qui est un histogramme.

2.4 Méthodes de rejet

2.4.1 Lois uniformes

Soit X une variable aléatoire produite par un algorithme $\mathcal{A}$ utilisant la fonction Random. La loi d'une variable aléatoire X est une mesure sur $\mathbb{R}$, image par l'algorithme $\mathcal{A}$ de la loi des appels de Random successifs. Conditionner par un évènement E de probabilité non nulle revient à remplacer cette loi $Prob[\,\cdot\,]$ par $Prob[\,\cdot \mid E]$. Il est alors naturel de modifier la loi de X pour la remplacer par sa loi conditionnelle sachant E. C'est la mesure sur $\mathbb{R}$ définie pour tout intervalle B de $\mathbb{R}$ par :

$$Prob[X \in B \mid E] = \frac{Prob[X^{-1}(B) \cap E]}{Prob[E]} .$$

En pratique, passer de la probabilité $Prob[\,\cdot\,]$ à la probabilité conditionnelle $Prob[\,\cdot \mid E]$ revient à remplacer l'algorithme $\mathcal{A}$ par le suivant.

Répéter $\mathcal{A}$ Jusqu'à (E réalisé).

Si X est une variable aléatoire calculée par $\mathcal{A}$, alors la loi de X en sortie de la boucle ci-dessus est la loi conditionnelle de X sachant E. Cette simple observation est à la base

de très nombreuses méthodes de simulation. Le coût d'une telle méthode est lui-même aléatoire, car il dépend du nombre de passages dans la boucle "Répéter ... Jusqu'à". Ce nombre suit la loi géométrique de paramètre $Prob[E]$. Son espérance vaut donc $1/Prob[E]$.

Exemple :

Soit $\alpha \in]0,1[$. Considérons l'algorithme

Répéter
    $X \longleftarrow$ Random
Jusqu'à $X \leq \alpha$

Désignons par R l'appel de Random. La loi de X en sortie de cet algorithme a pour fonction de répartition :

$$F_X^E(x) = Prob[R \leq x \mid R \leq \alpha\ ] = \begin{cases} 0 & \text{si } x \leq 0 \\ \frac{x}{\alpha} & \text{si } x \in [0,\alpha] \\ 1 & \text{si } x \geq \alpha\ . \end{cases}$$

La densité correspondante est :

$$f_X^E(x) = \frac{1}{\alpha} \mathbb{1}_{[0,\alpha]}(x)\ .$$

C'est celle de la loi uniforme sur $[0,\alpha]$. L'algorithme ci-dessus retourne donc un réel au hasard entre 0 et α. Il peut sembler plus économique de multiplier un appel de Random par α pour arriver au même résultat. Cela peut dépendre du compilateur et de la valeur de α. Voici quelques temps d'exécution observés, pour une boucle de taille 10000 en centièmes de secondes (TurboPascal 4.0 sur PC 386).

α	0.1	0.2	0.3	0.5	0.8	0.99
$X \longleftarrow \alpha *$ Random	94	93	94	44	93	93
Répéter $X \longleftarrow$ Random Jusqu'à $X < \alpha$	182	94	60	33	22	22

La technique ci-dessus vaut pour toutes les lois uniformes. Pour tirer au hasard (loi uniforme) dans un ensemble D on peut tirer au hasard dans un ensemble D' qui le contient et rejeter tous les tirages qui ne tombent pas dans D. Voici un énoncé plus précis pour les domaines de $\mathbb{R}^k$.

Proposition 2.11 *Soient D et D' deux domaines mesurables de $\mathbb{R}^k$ tels que*

$$D \subset D' \quad \text{et} \quad 0 < v(D) \leq v(D') < +\infty\ ,$$

où $v(D)$ et $v(D')$ désignent les volumes respectifs de D et D' dans $\mathbb{R}^k$.
Soit X un point aléatoire de loi uniforme sur D'. La loi conditionnelle de X sachant "$X \in D$" est la loi uniforme sur D.

Démonstration : L'évènement E par lequel on conditionne est "$X \in D$". Sa probabilité est

$$Prob[E] = \int_D \frac{1}{v(D')} \mathbb{1}_{D'}(x)\, dx = \frac{v(D)}{v(D')} \; .$$

Pour tout pavé B (produit cartésien d'intervalles) inclus dans $\mathbb{R}^k$, on a :

$$\begin{aligned} Prob[X \in B \,|\, E] = & \quad Prob[X \in B \cap D]/Prob[E] \\ = & \quad \frac{v(D')}{v(D)} \int_{B \cap D} \frac{1}{v(D')} \mathbb{1}_{D'}(x)\, dx \\ = & \quad \frac{1}{v(D)} \int_B \mathbb{1}_D(x)\, dx \; . \end{aligned}$$

□

La traduction algorithmique est la suivante.

Répéter
    tirer un point X au hasard dans D'
Jusqu'à $(X \in D)$

Le nombre moyen de passages dans la boucle suit la loi géométrique $\mathcal{G}\left(\frac{v(D)}{v(D')}\right)$. On a donc intérêt à choisir D' le plus proche possible de D, mais suffisamment simple pour ne pas ralentir la simulation. Ce sont les lois uniformes sur les pavés (produits d'intervalles) qui sont les plus rapides à simuler. On réalisera en général un compromis en construisant D' comme une réunion de pavés disjoints.

Exemple

Loi uniforme sur le disque unité

Répéter
    $X \longleftarrow 2 * \text{Random} - 1$
    $Y \longleftarrow 2 * \text{Random} - 1$
    $S \longleftarrow X * X + Y * Y$
Jusqu'à $(S < 1)$

$$\begin{array}{rclrcl} D & = & \{(x,y)\,;\; x^2 + y^2 \leq 1\} & v(D) & = & \pi \; , \\ D' & = & [-1,1]^2 & v(D') & = & 4 \; , \end{array}$$

le nombre moyen de passages dans la boucle est

$$\frac{v(D')}{v(D)} = \frac{4}{\pi} \simeq 1.27 \; .$$

La simulation des lois uniformes revient souvent dans les applications. Le résultat suivant montre que la simulation d'une loi de densité quelconque peut toujours se ramener à la simulation d'une loi uniforme.

Proposition 2.12 *Soit f une densité de probabilité par rapport à la mesure de Lebesgue sur $\mathbb{R}^k$, continue par morceaux, et*

$$D_f \;=\; \{(x,u) \in \mathbb{R}^k \times \mathbb{R}^+ \;;\; 0 < u < f(x)\} \;.$$

Soit X un vecteur aléatoire à valeurs dans $\mathbb{R}^k$ et U une variable aléatoire réelle. Le couple (X,U) suit la loi uniforme sur D_f si et seulement si

1. *X a pour densité f,*
2. *la loi conditionnelle de U sachant "$X = x$" est la loi uniforme sur $[0, f(x)]$.*

Démonstration : La densité de la loi uniforme sur D_f est :

$$f_{(X,U)}(x,u) = \frac{1}{v(D_f)}\, \mathbb{1}_{D_f}(x,u) \;.$$

Cette densité est bien le produit des deux densités :

$$f_{(X,U)}(x,u) = f(x)\, \Big(\frac{1}{f(x)} \mathbb{1}_{[0,f(x)]}(u)\Big) \;.$$

□

En d'autres termes, une variable aléatoire de densité f est l'abscisse d'un point au hasard sous le graphe de f. Ce qui vient d'être dit pour des densités par rapport à la mesure de Lebesgue s'étend directement à des mesures quelconques.

Une idée récurrente dans les méthodes de Monte-Carlo est qu'il est à peu près équivalent, sur le plan de la complexité algorithmique, d'évaluer la taille d'un domaine (qu'il s'agisse d'énumération ou d'un calcul d'intégrale) ou de tirer au hasard des éléments de ce domaine (voir [1, 2]). Nous venons d'en fournir une première illustration. L'algorithme qui tire des points au hasard dans D peut calculer en même temps le rapport $v(D)/v(D')$. Il permet donc aussi de calculer des intégrales.

Soit à intégrer la fonction f, supposée positive, continue et bornée sur le domaine Δ de $\mathbb{R}^d$, de volume fini. Cela revient à calculer le volume du domaine D de $\mathbb{R}^{d+1}$ défini par

$$D \;=\; \{(x,u) \in \mathbb{R}^d \times \mathbb{R} \,;\, x \in \Delta \,,\, 0 \le u \le f(x)\} \;.$$

$$I \;=\; \int_\Delta f(x)\, dx \;=\; v(D) \;=\; \int_{\mathbb{R}^{d+1}} \mathbb{1}_D(x)\, dx \;.$$

L'idée générale de la méthode de rejet consiste à tirer des points au hasard dans un domaine D' contenant D et à compter ceux d'entre eux qui tombent dans D. Leur proportion converge vers

$$\frac{1}{v(D')} \int_\Delta f(x) \quad .$$

On estimera donc I par le produit de $v(D')$ par la fréquence expérimentale des points qui tombent dans D. La variance σ^2 pour cet estimateur de I est

$$\sigma^2 \;=\; I(v(D') - I) \;.$$

Pour un nombre de tirages donné, la précision sera d'autant meilleure que D' sera plus proche de D. Mais bien sûr, approcher au mieux le domaine D comporte un coût algorithmique qu'il faudra mettre en balance avec le gain en précision.

2.4.2 Lois quelconques

Proposition 2.13 *Soit μ une mesure positive sur $\mathbb{R}^k$, f et g deux densités de probabilité sur $(\mathbb{R}^k, \mu)$ telles qu'il existe une constante c vérifiant :*

$$\forall x \in \mathbb{R}^k \ , \quad cg(x) \geq f(x) \ .$$

Soit X une variable aléatoire de densité g par rapport à μ et U une variable aléatoire de loi uniforme sur $[0,1]$, indépendante de X. Alors la loi conditionnelle de X sachant l'évènement E "$cUg(X) < f(X)$" a pour densité f par rapport à μ.

Démonstration : Si μ est la mesure de Lebesgue, on déduit le résultat des propositions 2.11 et 2.12. Voici une démonstration directe.

Deux remarques préliminaires :

- Si $g(x) = 0$ alors $f(x) = 0$ (le support de f est inclus dans celui de g).
- La constante c est plus grande que 1 car $c \int g(x)\,\mu(dx) \geq \int f(x)\,\mu(dx)$.

Calculons d'abord $Prob[E]$.

$$\begin{aligned}
Prob[E] = & \quad Prob[(X,U) \in \{(x,u) \ ; \ cug(x) < f(x)\}] \\
= & \quad \int_{\{(x,u) \ ; \ cug(x) < f(x)\}} g(x) \mathbb{1}_{[0,1]}(u)\,\mu(dx)du \\
= & \quad \int_{\{x \ ; \ g(x)>0\}} g(x) \left(\int_0^{\frac{f(x)}{cg(x)}} \mathbb{1}_{[0,1]}(u)du \right) \mu(dx) \\
= & \quad \int_{\{x \ ; \ g(x)>0\}} g(x) \frac{f(x)}{cg(x)}\,\mu(dx) \\
= & \quad \frac{1}{c} \int_{\mathbb{R}^k} f(x)\,\mu(dx) \\
= & \quad \frac{1}{c} \ .
\end{aligned}$$

On veut montrer que la densité conditionnelle de X sachant E est $f(x)$ soit, pour tout pavé B de $\mathbb{R}^k$:

$$Prob[X \in B \,|\, E] = \int_B f(x)\,\mu(dx) \ .$$

$$
\begin{aligned}
Prob[X \in B \,|\, E] = &\; c\, Prob[X \in B \text{ et } E] \\
= &\; c \int_{\{(x,u)\ ;\ x\in B\ ,\ cug(x)<f(x)\}} g(x) \mathbb{1}_{[0,1]}(u)\, \mu(dx)du \\
= &\; c \int_{x\in B} g(x) \left(\int_0^{\frac{f(x)}{cg(x)}} \mathbb{1}_{[0,1]}(u)du \right) \mu(dx) \\
= &\; c \int_B g(x) \frac{f(x)}{cg(x)}\, \mu(dx) \\
= &\; \int_B f(x)\, \mu(dx)\ ,
\end{aligned}
$$

en supposant (sans perte de généralité) que B est inclus dans le support de g. □

Cette proposition permet de partir d'une densité g facile à simuler pour simuler une loi de densité f quelconque. L'algorithme est le suivant.

Répéter
    Simuler X de densité g
    $U \longleftarrow$ Random
Jusqu'à $(cUg(X) < f(X))$

Le nombre de passages dans la boucle suit la loi géométrique $\mathcal{G}\left(\frac{1}{c}\right)$, d'espérance c. On aura donc intérêt à choisir g assez proche de f de sorte que la constante $c = \max\{f(x)/g(x)\}$ soit la plus faible possible.

Exemple

Soit à simuler la loi de densité :

$$f(x) = \frac{2}{\pi}\sqrt{1-x^2}\, \mathbb{1}_{[-1,1]}(x)\ .$$

(loi de l'abscisse d'un point au hasard dans le disque unité). Partons de la loi uniforme sur $[-1,1]$:

$$g(x) = \frac{1}{2}\mathbb{1}_{[-1,1]}(x)\ .$$

Prenons $c = \frac{4}{\pi}$ (n'importe quelle constante supérieure à $\frac{4}{\pi}$ conviendrait mais il faut choisir c minimale). Voici l'algorithme.

Répéter
    $X \longleftarrow 2 * \text{Random} - 1$
    $U \longleftarrow$ Random
Jusqu'à $\left(\frac{4}{\pi} U \frac{1}{2} < \frac{2}{\pi}\sqrt{1-X^2}\right)$

On l'écrira évidemment :

Répéter
    $X \longleftarrow 2 * \text{Random} - 1$

$U \longleftarrow$ Random
Jusqu'à $(U * U < 1 - X * X)$

Cet algorithme revient à tirer X comme l'abscisse d'un point au hasard sur le demi-disque supérieur :

$$\{(x, u)\ ;\ x^2 + u^2 < 1,\ u \geq 0\}\ .$$

Ceci n'est pas surprenant au vu de la proposition 2.12.

2.4.3 Lois discrètes

On peut appliquer la proposition 2.13 dans le cas où μ est la mesure de dénombrement sur un ensemble fini ou dénombrable.

Proposition 2.14 *Soient $p = (p(i))_{i \in I}$ et $q = (q(i))_{i \in I}$ deux lois de probabilités sur un même ensemble I, fini ou dénombrable. Supposons qu'il existe une constante c telle que :*

$$\forall i \in I\ , \quad p(i) \leq cq(i)\ .$$

Soit X une variable aléatoire de loi q et U une variable aléatoire de loi uniforme sur $[0, 1]$, indépendante de X. Alors la loi conditionnelle de X sachant l'évènement "$Ucq(X) < p(X)$" est la loi p.

La probabilité de l'évènement par lequel on conditionne est $1/c$. Le nombre moyen de tirages avant acceptation est donc c. Il faut choisir c le plus petit possible ($c = \max p(i)/q(i)$). Par rapport à la méthode d'inversion, cette proposition n'a d'intérêt que si d'une part q est très rapide à simuler et d'autre part p est proche de q (c proche de 1). En pratique, on ne l'utilise guère que si q est la loi uniforme sur un ensemble fini.

Répéter
$X \longleftarrow$ Random$(\{1, \ldots, n\})$
Jusqu'à (Random$< np(X)/c$)

Cette méthode sera meilleure qu'une méthode d'inversion typiquement pour un grand nombre d'éventualités quand leurs probabilités sont peu différentes.

Exemple.

Soit k un entier fixé. Considérons la loi $p = (p(k))$ sur $\{1, \ldots, n\}$ définie par

$$\begin{aligned} p(k) &= \frac{1}{2n-1} \quad \text{si } k = 1\ , \\ &= \frac{2}{2n-1} \quad \text{si } k = 2, \ldots, n\ . \end{aligned}$$

Un algorithme suivant la méthode d'inversion prendrait au mieux de l'ordre de $\log(n)$ opérations par variable engendrée. Dans la simulation par rejet à partir de la loi uniforme, la boucle principale sera exécutée en moyenne $c = 2n/(2n-1)$ fois. L'algorithme est le suivant.

Répéter
    $X \longleftarrow$ Random$[\{1, \ldots, n\}]$
    test $\longleftarrow$ vrai
    Si $(X = 1)$ alors
        Si (Random< 0.5) alors
            test $\longleftarrow$ faux
        finSi
    finSi
Jusqu'à (test = vrai)

2.5 Décomposition

2.5.1 Principe

Soit μ une mesure sur $\mathbb{R}^k$ et $(f_n), n \in \mathbb{N}$ une famille de densités de probabilité sur $(\mathbb{R}^k, \mu)$. Soit $(p_n), n \in \mathbb{N}$ une loi de probabilité sur $\mathbb{N}$. Notons f la densité de probabilité :

$$f = \sum_{n \in \mathbb{N}} p_n f_n \ .$$

Proposition 2.15 *Soit $(X_n), n \in \mathbb{N}$ une famille de variables aléatoires telles que X_n ait pour densité f_n par rapport à μ. Soit N une variable aléatoire de loi (p_n), indépendante de la suite (X_n). La variable aléatoire $X = X_N$ a pour densité f par rapport à μ.*

Démonstration : Si B est un pavé de $\mathbb{R}^k$, on a :

$$\begin{aligned} Prob[X \in B] &= \sum_{n \in \mathbb{N}} Prob[X \in B \,|\, N = n]\, Prob[N = n] \\ &= \sum_{n \in \mathbb{N}} Prob[X_n \in B]\, p_n \\ &= \sum_{n \in \mathbb{N}} p_n \int_B f_n(x)\, \mu(dx) \\ &= \int_B f(x)\, \mu(dx) \ . \end{aligned}$$

□

L'algorithme de simulation de la loi de densité f est donc

Choisir n avec probabilité p_n
Choisir X de densité f_n

La méthode de décomposition est particulièrement naturelle quand on souhaite simuler la loi uniforme sur une réunion de domaines disjoints. Soit D_i , $i = 1, \ldots, n$ une famille de domaines disjoints de $\mathbb{R}^k$, de volumes finis. Notons D leur réunion. La loi uniforme

sur D a pour densité

$$\frac{1}{v(D)}\mathbb{1}_D(x) \;=\; \sum_{i=1}^{n} \frac{v(D_i)}{v(D)}\,\frac{1}{v(D_i)}\mathbb{1}_{D_i}(x)\;.$$

C'est une combinaison convexe des densités des lois uniformes sur les domaines D_i. Pour simuler la loi uniforme sur D on peut donc procéder en deux temps.

Choisir i avec probabilité $v(D_i)/v(D)$
Tirer x au hasard sur D_i

L'algorithme tire au hasard un point dans l'un des domaines D_i, choisi avec une probabilité proportionnelle à son volume. Ceci est un cas particulier de la méthode générale de décomposition des lois à densité.

2.5.2 Lois à densité

Quand μ est la mesure de Lebesgue sur $\mathbb{R}^k$, on peut interpréter l'algorithme de décomposition par référence au cas des densités uniformes. Nous avons vu (proposition 2.12) que choisir un vecteur X de densité f revient à choisir un point au hasard dans le domaine

$$D_f \;=\; \{(x,u) \in \mathbb{R}^k \times \mathbb{R}^+ \;;\; 0 \le u \le f(x)\}\;.$$

L'algorithme de décomposition revient à découper le domaine D, de volume 1 en une réunion de domaines D_n, de volumes p_n. Dans les applications de la méthode de décomposition, on s'arrange pour choisir un découpage de sorte qu'avec une forte probabilité, on ait à simuler des lois uniformes.

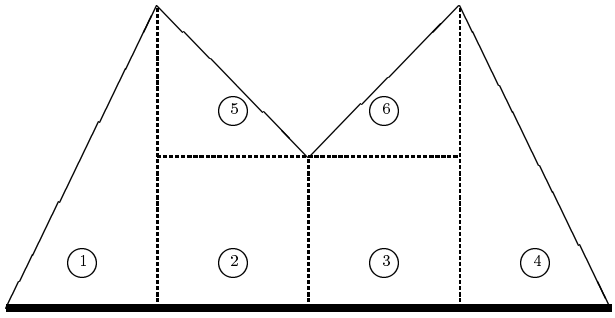
Exemple.

Considérons la densité suivante.

$$f(x) = \begin{cases} 0 & \text{si } x < 0 \\ \frac{2}{5}x & \text{si } x \in [0,1] \\ \frac{3}{5} - \frac{1}{5}x & \text{si } x \in [1,2] \\ -\frac{1}{5} + \frac{1}{5}x & \text{si } x \in [2,3] \\ \frac{8}{5} - \frac{2}{5}x & \text{si } x \in [3,4] \\ 0 & \text{si } x > 4 \end{cases}$$

Voici un algorithme de simulation par décomposition correspondant au découpage en 6 densités de la figure (remarquer que les probabilités p_i sont les aires respectives des morceaux du découpage.)

choix $\longleftarrow$ Random$(\{1,\ldots,10\})$
Selon choix



choix = 1 ou 2 faire
    $X \longleftarrow$ Max(Random, Random) *(densité 1 avec proba 1/5)*
choix = 3 ou 4 faire
    $X \longleftarrow$ Random +1 *(densité 2 avec proba 1/5)*
choix = 5 ou 6 faire
    $X \longleftarrow$ Random +2 *(densité 3 avec proba 1/5)*
choix = 7 ou 8 faire
    $X \longleftarrow$ Min(Random, Random)+2 *(densité 4 avec proba 1/5)*
choix = 9 faire
    $X \longleftarrow$ Min(Random, Random)+1 *(densité 5 avec proba 1/10)*
choix = 10 faire
    $X \longleftarrow$ Max(Random, Random)+2 *(densité 6 avec proba 1/10)*
finSelon

2.5.3 Lois discrètes

Soit $(X_n), n \in I\!N$ une famille de variables aléatoires à valeurs dans $I\!N$ telles que :

$$\forall n, m \in I\!N\,, \quad Prob[X_n = m] \;=\; p_{nm}\;.$$

Soit $(p_n), n \in I\!N$ une loi de probabilité sur $I\!N$. Pour tout $m \in I\!N$, posons :

$$q_m \;=\; \sum_{n \in I\!N} p_n\, p_{nm}\;.$$

On simule la loi de probabilité $q = (q_m)$ par l'algorithme :

Choisir n avec probabilité p_n
Choisir m avec probabilité p_{nm}

Ceci revient à simuler un pas d'une chaîne de Markov sur $I\!N$ (voir section 3.1).

Pour une loi discrète $q = (q_m)$, la simulation par inversion est bien adaptée au cas où un petit nombre parmi les probabilités q_m ont une somme déjà proche de 1. La simulation par rejet, au contraire, convient bien au cas où les différences entre probabilités sont faibles. La méthode de décomposition permet de réaliser des compromis.

Exemple.

Soit k un entier fixé. Considérons la loi $q = (q(k))$ sur $\{1, \ldots, 2n\}$ définie par :

$$\begin{aligned} q(k) &= \frac{1}{4(2n-1)} \quad \text{si } k = 1 \ , \\ &= \frac{2}{4(2n-1)} \quad \text{si } k = 2, \ldots, n \ , \\ &= \frac{6}{4(2n-1)} \quad \text{si } k = n+1, \ldots, 2n-1 \ , \\ &= \frac{3}{4(2n-1)} \quad \text{si } k = 2n \ . \end{aligned}$$

Un algorithme suivant la méthode d'inversion prendrait au mieux de l'ordre de $\log(n)$ opérations par variable engendrée. Dans la simulation par rejet à partir de la loi uniforme, la boucle principale sera exécutée en moyenne environ 1.5 fois. L'algorithme suivant commence par choisir entre les deux sous-ensembles $\{1, \ldots, n\}$ et $\{n+1, \ldots, 2n\}$ par inversion, puis simule par rejet à partir de la loi uniforme dans chacun des deux intervalles.

```
Répéter
    Si Random< 0.25 alors
        X ⟵ Random[{1, ..., n}]
        test ⟵ vrai
        Si (X = 1 ) alors
            Si (Random< 0.5) alors
                test ⟵ faux
            finSi
        finSi
    Sinon
        X ⟵ Random[{n + 1, ..., 2n}]
        test ⟵ vrai
        Si (X = 2n) alors
            Si (Random< 0.5) alors
                test ⟵ faux
            finSi
        finSi
    finSi
Jusqu'à (test = vrai)
```

2.6 Simulation des lois normales

Bien que nous ayons choisi de ne pas entrer dans les détails des méthodes de simulations des lois usuelles, nous ferons une exception pour la loi normale. Tout d'abord parce

qu'elle entre comme ingrédient essentiel dans les simulation des équations différentielles stochastiques, mais aussi pour donner des exemples de calculs de complexité sur des algorithmes de simulation. De nombreuses méthodes sont proposées dans les manuels. Toutes ne sont pas à conseiller. Il est bien sûr possible de trouver une méthode de décomposition, adaptée non seulement à la densité de la loi $\mathcal{N}(0,1)$ mais aussi aux qualités du générateur et du compilateur, qui soit plus rapide que celles qui suivent (voir Devroye [26] ou Morgan [67]). Les méthodes que nous donnons ici sont faciles à programmer.

2.6.1 Principe

Deux algorithmes de simulation usuels sont basés sur le résultat théorique suivant.

Proposition 2.16 *Soit (X,Y) un couple de variables aléatoires, de loi uniforme sur le disque unité*

$$D = \{(x,y)\ ;\ x^2+y^2<1\}\ .$$

Soit (R,Θ) le couple de coordonnées polaires correspondant :

$$X = R\cos\Theta\ ;\ Y = R\sin\Theta\ .$$

On pose :

$$R' = \sqrt{-4\log R}\ .$$

Alors :

$$U = R'\cos\Theta \text{ et } V = R'\sin\Theta$$

sont deux variables aléatoires indépendantes, de même loi $\mathcal{N}(0,1)$.

Démonstration :

Nous commençons par déterminer la densité du couple (R,Θ), par le changement de variables suivant.

$$\begin{array}{rccc} \Phi\ :\ & D\setminus(]0,1[\times\{0\}) & \longrightarrow &]0,1[\times]0,2\pi[\\ & (x,y) & \longrightarrow & (r,\theta) \end{array}$$

$$\begin{array}{rccc} \Phi^{-1}\ :\ &]0,1[\times]0,2\pi[& \longrightarrow & D\setminus(]0,1[\times\{0\}) \\ & (r,\theta) & \longrightarrow & (r\cos\theta, r\sin\theta) \end{array}$$

$$J_{\Phi^{-1}} = \left|\begin{array}{cc} \cos\theta & -r\sin\theta \\ \sin\theta & r\cos\theta \end{array}\right| = r\ .$$

$$f_{R,\Theta}(r,\theta) = \frac{1}{\pi}\mathbb{1}_D(r\cos\theta, r\sin\theta)r = \frac{r}{\pi}\mathbb{1}_{]0,1[\times]0,2\pi[}(r,\theta)\ ,$$

$$f_R(r) = \int_0^{2\pi}\frac{1}{\pi}r\mathbb{1}_{]0,1[}(r)d\theta = 2r\mathbb{1}_{]0,1[}(r)\ ,$$

$$f_\Theta(\theta) = \int_0^1\frac{1}{\pi}r\mathbb{1}_{]0,2\pi[}(\theta)dr = \frac{1}{2\pi}\mathbb{1}_{]0,2\pi[}(\theta)\ .$$

Les variables R et Θ sont indépendantes, R de loi triangulaire sur $]0,1[$, Θ de loi uniforme sur $]0,2\pi[$). Déterminons la loi de R'.

$$F_{R'}(r') = Prob[R' \leq r'] = Prob\left[\sqrt{-4\log R} < r'\right] = 0 \quad \text{si } r' \leq 0 \ ,$$

$$F_{R'}(r') = Prob\left[R > \exp\left(-\frac{1}{4}r'^2\right)\right] = 1 - F_R\left(\exp -\frac{1}{4}r'^2\right) \quad \text{si } r' > 0 \ .$$

D'où la densité

$$f_{R'}(r') = \frac{1}{2}r' \exp -\frac{1}{4}r'^2 f_R(\exp -\frac{1}{4}r'^2) \quad \text{si } r' > 0 \ ,$$

$$f_{R'}(r') = r' \exp(-\frac{1}{2}r'^2)\, \mathbb{1}_{I\!R^+}(r') \ . \tag{2.1}$$

Déterminons enfin la densité du couple (U,V), par un nouveau changement de variables.

$$\begin{array}{rccl} \Psi \ : & I\!R^+ \times]0,2\pi[& \longrightarrow & I\!R^2 \setminus (]0,+\infty[\times\{0\}) \\ & (r',\theta) & \longrightarrow & (u,v) = (r'\cos\theta, r'\sin\theta) \end{array}$$

Ψ restreint à $]0,1[\times]0,2\pi[$ coïncide avec Φ^{-1}, donc

$$J_{\Psi^{-1}} = \frac{1}{J_{\Phi^{-1}}(r'(u,v),\theta(u,v))} = \frac{1}{r'} = \frac{1}{\sqrt{u^2+v^2}} \ .$$

D'où

$$\begin{array}{rcl} f_{U,V}(u,v) = & f_{R',\Theta}(r'(u,v),\theta(u,v))\dfrac{1}{\sqrt{u^2+v^2}} \\ = & \dfrac{1}{2\pi}\sqrt{u^2+v^2}\exp -\dfrac{1}{2}(u^2+v^2)\dfrac{1}{\sqrt{u^2+v^2}} \\ = & \dfrac{1}{2\pi}\exp -\dfrac{1}{2}(u^2+v^2) \\ = & \left(\dfrac{1}{\sqrt{2\pi}}\exp -\dfrac{u^2}{2}\right)\left(\dfrac{1}{\sqrt{2\pi}}\exp -\dfrac{v^2}{2}\right) \ . \end{array}$$

□

2.6.2 Algorithme polaire

Nous avons vu précédemment comment simuler la loi uniforme sur le disque unité.

Répéter
  $X \longleftarrow 2 * \mathsf{Random} - 1$
  $Y \longleftarrow 2 * \mathsf{Random} - 1$
  $S \longleftarrow X * X + Y * Y$
Jusqu'à $(S < 1)$ *(* (X,Y) de loi uniforme sur le disque unité *)*
$Z \longleftarrow \mathsf{Sqrt}(-2 * \log(S)/S)$ *(* changement de norme *)*

$U \longleftarrow Z * X$
$V \longleftarrow Z * Y$

Justification :

$$Z = \sqrt{\frac{-2\log R^2}{R^2}} = \frac{1}{R}\sqrt{-4\log R} = \frac{R'}{R} ,$$

$$U = \frac{R'}{R}X = R'\cos\Theta$$

et de même

$$V = R'\sin\Theta .$$

D'après la proposition 2.16, U et V sont indépendantes et de même loi $\mathcal{N}(0,1)$. Cet algorithme engendre donc les variables aléatoires 2 par 2. Ce n'est pas un inconvénient dans la mesure où un grand nombre de simulations sont nécessaires et où les résultats U et V pourront être utilisés successivement.

Etudions maintenant le coût de cet algorithme. Rappelons que le nombre d'exécutions de la boucle "Répéter ... Jusqu'à" suit une loi géométrique de paramètre $\pi/4$, d'espérance $4/\pi \simeq 1.27$. Voici le tableau des nombres moyens d'opérations pour une variable aléatoire engendrée.

Affectations	Tests	Additions	Random	Multiplications	Fonctions chères
$\frac{3\times 1.27+3}{2}$	$\frac{1.27}{2}$	$\frac{5\times 1.27+1}{2}$	$\frac{2\times 1.27}{2}$	$\frac{2\times 1.27+3}{2}$	$\frac{2}{2}$

Temps d'exécution observés pour 20000 simulations en Pascal sur PC 386 : entre 24.27s. et 24.34s.

2.6.3 Algorithme de Box-Muller

Fréquemment proposé dans les manuels cet algorithme est basé sur la même méthode polaire, mais programmée différemment.

$R' \longleftarrow \mathsf{Sqrt}(-2 * \log(\mathsf{Random}))$
$\Theta \longleftarrow 2\mathsf{Pi} * \mathsf{Random}$
$U \longleftarrow R' * \cos\theta$
$V \longleftarrow R' * \sin\theta$

Justification : En se reportant à la démonstration de la proposition 2.16, il suffit de vérifier que $\sqrt{-2\log(\mathsf{Random})}$ a la même densité (2.1) que R'.

Pour $r' \geq 0$:

$$Prob\left[\sqrt{-2\log(\mathsf{Random})} < r'\right] = Prob\left[\mathsf{Random} > \exp -\frac{1}{2}r'^2\right] = 1 - \exp(-\frac{1}{2}r'^2) .$$

D'où la densité :

$$r'\exp(-\frac{1}{2}r'^2)\mathbb{1}_{I\!R^+}(r') .$$

Voici le tableau des nombres moyens d'opérations pour une variable engendrée.

Affectations	Tests	Additions	Random	Multiplications	Fonctions chères
$\frac{4}{2}$	0	$\frac{1}{2}$	$\frac{2}{2}$	$\frac{3}{2}$	$\frac{4}{2}$

Temps d'exécution observé pour 20000 simulations en Pascal sur PC 386 : 39.99s. (1.64 fois plus lent que l'algorithme polaire). Mais selon les compilateurs, il peut se faire que l'algorithme de Box-Muller soit plus rapide que l'algorithme polaire.

2.6.4 Conditionnement d'exponentielles

Voici une méthode différente proposée dans certains manuels.

Proposition 2.17 *Soit X et Y deux variables aléatoires indépendantes et de même loi, exponentielle de paramètre* 1. *La loi conditionnelle de X sachant $E =$ "$Y > \frac{1}{2}(1-X)^2$" a pour densité :*

$$f_X^E(x) = \frac{2}{\sqrt{2\pi}} e^{-x^2/2} \mathbb{1}_{I\!R^+}(x) \ .$$

Soit S une variable aléatoire indépendante de X et Y, prenant les valeurs ± 1 avec probabilité 1/2. *Alors SX suit la loi $\mathcal{N}(0,1)$.*

Démonstration :

$$\begin{aligned} Prob[E] = & \int_0^{+\infty} e^{-x} \int_{\frac{(1-x)^2}{2}}^{+\infty} e^{-y}\, dydx \\ = & \int_0^{\infty} e^{-1/2} e^{-x^2/2}\, dx \\ = & \frac{\sqrt{2\pi}}{2} e^{-1/2} \\ \simeq & \ 0,76 \ . \end{aligned}$$

$$Prob[X \leq x \text{ et } E] = \int_0^x e^{-u} \int_{\frac{(1-u^2)}{2}}^{+\infty} e^{-y} dydu = \int_0^x e^{-1/2} e^{-u^2/2} du \ .$$

Donc

$$F_X^E(x) = \frac{Prob[X \leq x \text{ et } E]}{Prob[E]} = \frac{2}{\sqrt{2\pi}} \int_0^x e^{-u^2/2} du$$

et

$$f_X^E(x) = \frac{2}{\sqrt{2\pi}} e^{-x^2/2} \mathbb{1}_{I\!R^+}(x) \ .$$

$$Prob[SX \leq x] = \frac{1}{2} + \frac{1}{2} \int_0^x f_X^E(u) du \ \text{ si } x \geq 0$$

$$Prob[SX \leq x] = \frac{1}{2} \int_{-x}^{+\infty} f_X^E(u) du \ \text{ si } x \leq 0$$

d'où la densité de SX :

$$\frac{1}{\sqrt{2\pi}} e^{-x^2/2} \ .$$

□

Voici l'algorithme correspondant.

Répéter
$X \longleftarrow -\log(\text{Random})$
$Y \longleftarrow -\log(\text{Random})$
Jusqu'à $(Y > (1-X)^2/2)$
Si (Random< 0.5) alors $X \longleftarrow -X$

Le nombre d'exécutions de la boucle suit une loi géométrique $\mathcal{G}(Prob[E])$, d'espérance $1/Prob[E] \simeq 1.32$. Voici le tableau des nombres moyens d'opérations pour une variable engendrée.

Affectations	Tests	Additions	Random	Multiplications	Fonctions chères
$2 \times 1.32 + \frac{1}{2}$	$1.32 + 1$	2×1.32	$2 \times 1.32 + 1$	1.32	2×1.32

Temps d'exécution observés pour 20000 simulations en Pascal sur PC 386 : entre 66.3 et 67.1 secondes (2,7 fois plus cher que l'algorithme polaire).

2.6.5 Lois normales multidimensionnelles

De la simulation de la loi $\mathcal{N}(0,1)$ on déduit celle de la loi normale d'espérance μ et de variable σ^2 quelconques par une transformation affine. Si X suit la loi $\mathcal{N}(0,1)$, alors $Y = \sigma X + \mu$ suit la loi $\mathcal{N}(\mu, \sigma^2)$. La situation est analogue en dimension k quelconque. Tout d'abord si $X_1, \ldots, X_k$ sont indépendantes et de même loi $\mathcal{N}(0,1)$, alors le vecteur (X_i) suit la loi normale dans $\mathbb{R}^k$, d'espérance nulle et de matrice de covariance identité : $\mathcal{N}_k(0, I)$. On en déduit la simulation d'une loi normale d-dimensionnelle quelconque par la proposition suivante.

Proposition 2.18 *Soit μ un vecteur de $\mathbb{R}^k$ et Σ une matrice de $\mathcal{M}_{k\times k}(\mathbb{R})$, symétrique positive. Soit $X = (X_i)$ un vecteur aléatoire de loi $\mathcal{N}_k(0, I)$ dans $\mathbb{R}^k$. Soit A une matrice carrée d'ordre k telle que $A\,{}^tA = \Sigma$. Alors le vecteur $Y = AX + \mu$ est un vecteur gaussien de moyenne μ et de matrice de covariance Σ.*

Démonstration : Toute combinaison linéaire des coordonnées de Y est combinaison affine des coordonnées de X, et suit une loi normale. Le vecteur Y est donc gaussien. On a :

$$\mathbb{E}[Y] \;=\; A\,\mathbb{E}[X] \;+\; \mu \;=\; \mu\;,$$

et

$$\mathbb{E}[(Y-\mu)\,{}^t(Y-\mu)] \;=\; A\,\mathbb{E}[X\,{}^tX]\,A \;=\; A\,{}^tA \;=\; \Sigma\;.$$

□

Il faut donc trouver une matrice A telle que $A\,{}^tA = \Sigma$. C'est un problème très classique. Une des réponses est implémentée dans la plupart des librairies d'algèbre linéaire. C'est

la décomposition de Cholesky. Dans le cas où Σ est définie positive, cette méthode calcule colonne par colonne une matrice A, triangulaire inférieure telle que $A\,{}^tA = \Sigma$. Voici par exemple le cas particulier $d = 2$. Soit (X_1, X_2) un couple de variables aléatoires indépendantes, de même loi $\mathcal{N}(0,1)$. Soient μ_1 et μ_2 deux réels quelconques, σ_1 et σ_2 deux réels positifs et ρ un réel compris entre -1 et 1. Posons :

$$\begin{aligned} Y_1 &= \mu_1 + \sigma_1 X_1 \ , \\ Y_2 &= \mu_2 + \sigma_2(\rho X_1 + \sqrt{1-\rho^2} X_2) \ . \end{aligned}$$

Les variables aléatoires Y_1 et Y_2 forment un couple gaussien, leurs espérances respectives sont μ_1 et μ_2, leurs variances sont σ_1^2 et σ_2^2 et leur coefficient de corrélation est ρ.

2.7 Calculs d'espérances

2.7.1 Principe

Il existe de nombreuses manières d'utiliser la loi des grands nombres pour calculer une intégrale. Soit à intégrer la fonction f, supposée positive, continue et bornée sur le domaine Δ de $\mathbb{R}^d$, de mesure finie.

$$I = \int_\Delta f(x)\,dx \ .$$

La méthode de rejet calcule le volume d'un domaine de $\mathbb{R}^{d+1}$. Soit f_X une densité de probabilité, strictement positive sur Δ, nulle en dehors. Pour tout $x \in \Delta$, posons :

$$g(x) = f(x)/f_X(x) \ .$$

Soit $(X_n)_{n\in\mathbb{N}}$ une suite de variables aléatoires indépendantes de densité f_X. L'espérance de $g(X_n)$ est :

$$\mathbb{E}[g(X_n)] = \int_\Delta g(x) f_X(x)\,dx = I \ .$$

On obtient donc une valeur approchée de l'intégrale par la moyenne empirique des $g(X_n)$.

$I \longleftarrow 0$
Répéter n fois
    tirer X dans Δ selon la densité f_X
    $I \longleftarrow I + g(X)$
finRépéter
$I \longleftarrow I/n$

La précision est contrôlée par la variance :

$$\begin{aligned} \sigma^2 &= \int_\Delta (g(x) - I)^2 f_X(x)\,dx \ , \\ &= \int_\Delta I\, f(x)\, \frac{f(x)}{I\, f_X(x)} \left(1 - \frac{I\, f_X(x)}{f(x)}\right)^2 dx \ . \end{aligned}$$

La précision est donc d'autant meilleure que la densité f_X est proche de $f(x)/I$. A la limite, elle vaut 0 si la densité est proportionnelle à la fonction. Mais ceci n'est d'aucune utilité pratique car pour savoir simuler suivant une densité proportionnelle à f, il faut pouvoir la calculer, c'est-à-dire connaître I.

L'algorithme le plus simple est obtenu pour la loi uniforme sur Δ.

$I \longleftarrow 0$
Répéter n fois
    tirer X au hasard dans Δ
    $I \longleftarrow I + f(X)$
finRépéter
$I \longleftarrow I * v(\Delta)/n$

La variance dans ce cas est

$$\sigma^2 \;=\; \int_\Delta (v(\Delta)f(x) - I)^2 \frac{1}{v(\Delta)}\, dx \;.$$

Cette variance est meilleure que celle que l'on obtient par la méthode de rejet appliquée au domaine

$$D' \;=\; \Delta \times [0, \max_{x\in\Delta} f(x)] \;,$$

qui vaut

$$\sigma'^2 \;=\; I\Big(v(\Delta) \max_{x\in\Delta} f(x) - I\Big) \;.$$

2.7.2 Variables négativement corrélées

L'idée des variables négativement corrélées (antithetic variates) est extrêmement séduisante quand on la présente en dimension 1. Soit f une fonction croissante définie sur $[0,1]$, et supposons que l'on veuille calculer son intégrale. On peut l'approcher par les moyennes empiriques des images par f d'appels de Random successifs, comme nous venons de le voir. On peut aussi remarquer que

$$\int_{[0,1]} f(x)\, dx \;=\; \int_{[0,1]} \frac{1}{2}(f(x) + f(1-x))\, dx \;,$$

et donc approcher l'intégrale par

$$\frac{1}{n}\sum_{i=1}^{n} \frac{1}{2}(f(u_i) + f(1-u_i)) \;,$$

où les u_i sont des appels successifs de Random. Si U désigne une variable de loi uniforme sur $[0,1]$ (appel de Random), $1-U$ a la même loi que U. La variance devient

$$Var[\frac{1}{2}(f(U) + f(1-U))] \;=\; \frac{1}{2}\Big(Var[f(U)] + Cov[f(U), f(1-U)]\Big) \;.$$

Comme f est croissante, $f(U)$ et $f(1-U)$ sont corrélées négativement

$$Cov(f(U), f(1-U)) \leq 0 \;.$$

Pour un même nombre d'appels de Random, on a certes alourdi l'algorithme en doublant le nombre d'évaluations de f, mais on a divisé par plus de deux la variance.

Le problème avec cette astuce est de l'appliquer valablement en dimension supérieure (rappelons qu'on ne calcule pas des intégrales simples par Monte-Carlo). De nombreuses heuristiques ont été avancées, sans pour autant aboutir à des algorithmes optimisés, valables pour tous types de fonctions (voir [52] p. 186–200, [35] p. 311–321).

2.7.3 Réduction de la variance

Nous ne ferons pas ici un recensement exhaustif des multiples astuces regroupées sous le nom de "méthodes de réduction de la variance". Deux principes doivent être gardés en mémoire.

1. Plus on a d'information sur la fonction à intégrer, mieux on pourra ajuster l'algorithme de manière à diminuer la variance. Il faut se souvenir qu'en pratique on n'utilise un algorithme de Monte-Carlo que dans des cas où la fonction est très difficile à étudier.
2. Une diminution de la variance s'accompagne en général d'une augmentation du coût. Or c'est la précision atteinte pour un coût donné qui compte.

Nous illustrerons les multiples manières de calculer une intégrale par un algorithme de Monte-Carlo, sur un exemple élémentaire en dimension 1 :

$$\int_0^2 x^2\,dx = \frac{8}{3}\ .$$

Pour chacune des 10 méthodes proposées, on pourra à titre d'exercice écrire l'algorithme, calculer le nombre moyen d'appels de Random, de multiplications, d'additions et de tests. On vérifiera que c'est bien la valeur $8/3$ qui est calculée, mais surtout que la variance annoncée est la bonne. On en déduira un classement des méthodes par ordre d'efficacité.

Méthode 1 : Rejet sur le domaine Δ avec :

$$\Delta = [0,2]\times[0,4]\ .$$

$$Variance = \frac{8}{3}\left(8 - \frac{8}{3}\right) = 14.22\ .$$

Méthode 2 : Rejet sur le domaine Δ avec :

$$\Delta = [0,1]\times[0,1] \bigcup [1,2]\times[0,4]\ .$$

$$Variance = \frac{8}{3}\left(5 - \frac{8}{3}\right) = 6.22\ .$$

Méthode 3 : Rejet sur le domaine Δ avec :

$$\Delta = [0,\frac{1}{2}]\times[0,\frac{1}{4}] \bigcup [\frac{1}{2},1]\times[0,1] \bigcup [1,\frac{3}{2}]\times[0,\frac{9}{4}] \bigcup [\frac{3}{2},2]\times[0,4] .$$

$$Variance = \frac{8}{3}\left(\frac{30}{8} - \frac{8}{3}\right) = 2.89 .$$

Méthode 4 : Espérance par rapport à la loi de densité f_X avec :

$$f_X(x) = \frac{1}{2}\mathbb{1}_{[0,2]}(x) .$$

$$Variance = \frac{256}{45} = 5.69 .$$

Méthode 5 : Espérance par rapport à la loi de densité f_X avec :

$$f_X(x) = \frac{1}{5}\mathbb{1}_{[0,1]}(x) + \frac{4}{5}\mathbb{1}_{[1,2]}(x) .$$

$$Variance = \frac{59}{36} = 1.64 .$$

Méthode 6 : Espérance par rapport à la loi de densité f_X avec :

$$f_X(x) = \frac{2}{30}\mathbb{1}_{[0,\frac{1}{2}]}(x) + \frac{8}{30}\mathbb{1}_{[\frac{1}{2},1]}(x) + \frac{18}{30}\mathbb{1}_{[1,\frac{3}{2}]}(x) + \frac{32}{30}\mathbb{1}_{[\frac{3}{2},2]}(x) .$$

$$Variance = \frac{2227}{4608} = 0.48 .$$

Méthode 7 : Espérance par rapport à la loi de densité f_X avec :

$$f_X(x) = \frac{1}{2}x\mathbb{1}_{[0,2]}(x) .$$

$$Variance = \frac{8}{9} = 0.89 .$$

Méthode 8 : Espérance par rapport à la loi de densité f_X avec :

$$f_X(x) = \frac{1}{3}x\mathbb{1}_{[0,1]}(x) + \frac{1}{3}(3x-2)\mathbb{1}_{[1,2]}(x) .$$

$$Variance = \frac{71}{36} + \frac{4}{81}(-45 + 4\log(4)) = 0.18 .$$

Méthode 9 : Variables négativement corrélées :

$$\int_0^2 x^2\,dx = \frac{1}{2}\int_0^2 (x^2 + (2-x)^2)\,dx\ .$$

Rejet sur le domaine Δ avec :

$$\Delta = [0,2]\times[0,4]\ .$$

$$Variance = \frac{8}{3}(4 - \frac{8}{3}) = 3.56\ .$$

Méthode 10 : Variables négativement corrélées :

$$\int_0^2 x^2\,dx = \frac{1}{2}\int_0^2 (x^2 + (2-x)^2)\,dx\ .$$

Espérance par rapport à la loi de densité f_X avec :

$$f_X(x) = \frac{1}{2}\mathbb{1}_{[0,2]}(x)\ .$$

$$Variance = \frac{16}{45} = 0.35\ .$$

2.8 Suites déterministes

On attend des k-uplets consécutifs d'appels de Random qu'ils se comportent comme des points au hasard dans $[0,1]^k$. Nous avons traduit ceci par la notion de k-uniformité. C'est cette propriété qui est utilisée dans la méthode de rejet (pour $k = d$+1) aussi bien que dans le paragraphe précédent. Une suite (u_n) de vecteurs de $[0,1]^k$ est uniforme si

$$\forall D =]a_1,b_1]\times\ldots\times]a_k,b_k] \subset [0,1]^k\ , \qquad \lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}\mathbb{1}_D(u_i)\ =\ v(D)\ .$$

Intuitivement, on demande à une suite uniforme dans $[0,1]^k$ de "visiter régulièrement" tout $[0,1]^k$. On exige beaucoup d'un générateur en demandant qu'il soit uniforme pour tout k, si on n'utilise en fait que la k-uniformité pour une valeur précise de k. Or dans les méthodes que nous avons vues jusqu'ici, la valeur de k est fixée par la nature du problème ($k = d$ ou $d+1$). Pour k fixé, il est possible de construire des ensembles de points qui visitent $[0,1]^k$ plus régulièrement et plus vite que des k-uplets d'appels de Random. Deux méthodes sont très répandues.

2.8.1 Points régulièrement répartis

Pour k fixé, considérons l'ensemble suivant de m^k points régulièrement répartis dans $[0,1]^k$.

$$\left\{(\frac{m_1}{m},\ldots,\frac{m_k}{m})\ ;\ (m_1,\ldots,m_k)\in\{1,\ldots,m\}^k\right\}\ .$$

Dans tous les algorithmes de calculs d'intégrales vus précédemment, on peut remplacer les moyennes sur n appels de Random par des moyennes sur les m^k points définis ci-dessus. Pour un nombre de points $n = m^k$ fixé, les calculs seront plus rapides qu'avec des appels de Random. La précision pourra être meilleure dans les cas où la fonction à intégrer est suffisamment régulière (voir [12], p. 226).

2.8.2 Suites de Van der Corput

L'inconvénient de la méthode précédente est que pour augmenter le nombre de points (passer de m à $m+1$) il faut recalculer tous les points. Il est donc souhaitable de disposer d'une suite déterministe de points qui visite $[0,1]^k$ plus régulièrement que la suite des appels de Random. De telles suites sont dites "à discrépance faible". Les plus courantes sont les suites de Van der Corput, définies comme suit. Pour k fixé, considérons les k premiers nombres premiers (2,3,5,7,11...). Soit π_i le i-ième nombre premier de la liste. Chaque entier n admet une écriture unique en base π_i.

$$n \;=\; a_0(n) + a_1(n)\pi_i + \cdots + a_\ell(n)\pi_i^\ell \;,$$

avec $0 \leq a_j(n) < \pi_i$. On lui associe

$$u_i(n) \;=\; \frac{a_0(n)}{\pi_i} + \cdots + \frac{a_\ell(n)}{\pi_i^{\ell+1}} \;.$$

On définit alors la suite $(u(n)), n \in I\!N$ d'éléments de $[0,1]^k$ par

$$u(n) \;=\; (u_1(n), \ldots, u_k(n)) \;.$$

On peut utiliser cette suite exactement comme une suite de k-uplets consécutifs d'appels de Random. Pour une fonction f définie sur $[0,1]^k$, on majore l'erreur d'approximation entre la moyenne des $f(u(n))$ et l'intégrale de f sur $[0,1]^k$ comme suit ([12] p. 229).

$$\left| \frac{1}{n} \sum_{j=1}^{n} f(u(j)) \;-\; \int_{[0,1]^k} f(x)\,dx \right| \;<\; V(f) \left(\prod_{i=1}^{k} \frac{\pi_i \log(\pi_i n)}{\log(\pi_i)} \right) \frac{1}{n} \;,$$

où $V(f)$ désigne la variation totale de la fonction f. L'erreur d'approximation est de l'ordre de $(\log(n))^k/n$, ce qui est meilleur que $1/\sqrt{n}$. Il est à noter cependant que cette erreur augmente fortement avec la dimension. De plus, le calcul des $u_i(n)$, s'il est peu coûteux pour de faibles valeurs de i, deviendra vite assez lourd en dimension plus grande (le 10-ième nombre premier est 29, le 100-ième est 541, le 1000-ième est 7919). Il est donc prévisible que sur les problèmes de très grande dimension, la suite de Van der Corput ne soit pas concurrentielle par rapport à l'utilisation de Random. Pour plus de détails sur les suites à discrépance faible, se reporter à Bouleau [14] p. 67-95.

2.9 Exercices

Exercice 2.1 Déterminer la loi de la variable aléatoire X, en sortie des algorithmes suivants :

1. $N \longleftarrow$ Int(Random*4)
   $X \longleftarrow$ Int(Random*N)
2. $N \longleftarrow$ Int(Random*3)
   $X \longleftarrow 0$
   Pour I de 0 à N faire
   $X \longleftarrow X + I$
   FinPour
3. $X \longleftarrow 0$; $Y \longleftarrow 1$
   Répéter
   $X \longleftarrow X + 1$; $Y \longleftarrow Y/2$
   Jusqu'à (Random $> Y$)
4. $N \longleftarrow 0$
   Répéter n fois
   Si (Random $< p$) alors $N \longleftarrow N + 1$
   finSi
   finRépéter
   $X \longleftarrow 0$
   Répéter N fois
   Si (Random $< q$) alors $X \longleftarrow X + 1$
   finSi
   finRépéter
5. $X \longleftarrow 0$
   Répéter n fois
   Si (Random $< p$) alors
   Si(Random $< q$) alors $X \longleftarrow X + 1$
   finSi
   finSi
   finRépéter
6. $P \longleftarrow p$; $F \longleftarrow P$; $X \longleftarrow 1$
   $C \longleftarrow$Random
   TantQue ($C > F$) faire
   $P \longleftarrow P * (1 - p)$
   $F \longleftarrow F + P$
   $X \longleftarrow X + 1$
   finTantQue
7. $N \longleftarrow 0$
   $X \longleftarrow 0$
   Répéter
   Si(Random $< p$) alors $N \longleftarrow N + 1$
   finSi
   $X \longleftarrow X + 1$
   Jusqu'à ($N = r$)

8. $X \longleftarrow 0$
   Répéter r fois
      Répéter
         $X \longleftarrow X + 1$
      Jusqu'à (Random $< p$)
   finRépéter

Exercice 2.2 On considère l'algorithme suivant, où Int désigne la partie entière.

$X \longleftarrow 0$
Répéter 3 fois
   $N \longleftarrow 0$
   Répéter
      $A \longleftarrow$ Int(Random $* 3$)
      $N \longleftarrow N + 1$
   Jusqu'à $(A < 2)$
   Si $(A = 0)$ alors $X \longleftarrow X + 1$
   finSi
finRépéter.

1. Quelle est la loi de la variable aléatoire X ?
2. Calculer la probabilité de l'évènement $X \leq 2$.
3. Quelle est la loi de la variable aléatoire N ?
4. Quelle est son espérance ?
5. Calculer la probabilité de l'évènement $N \leq 2$.
6. Sur 10000 répétitions indépendantes de cet algorithme, on a observé les résultats suivants pour X :

Valeurs	0	1	2	3
Effectifs	1264	3842	3681	1213

   (a) Donner un intervalle de confiance pour la probabilité de l'évènement $X \leq 2$, au niveau de confiance 0.95, puis 0.99.
   (b) Combien faudrait-il effectuer de répétitions indépendantes de l'algorithme pour que l'amplitude de l'intervalle de confiance au niveau 0.95 soit inférieure à 0.001 ?
   (c) Donner un intervalle de confiance pour l'espérance de X, au niveau de confiance 0.95, puis 0.99.
   (d) Combien faudrait-il effectuer de répétitions indépendantes de l'algorithme pour que l'amplitude de l'intervalle de confiance au niveau 0.95 soit inférieure à 0.001 ?
7. Sur 10000 répétitions indépendantes de cet algorithme, on a observé les résultats suivants pour N :

Valeurs	1	2	3	4	5	6	7	8
Effectifs	6613	2247	799	215	82	33	9	2

(a) Donner un intervalle de confiance pour la probabilité de l'évènement $N \leq 2$, au niveau de confiance 0.95, puis 0.99.

(b) Combien faudrait-il effectuer de répétitions indépendantes de l'algorithme pour que l'amplitude de l'intervalle de confiance au niveau 0.95 soit inférieure à 0.001 ?

(c) Donner un intervalle de confiance pour l'espérance de N, au niveau de confiance 0.95, puis 0.99.

(d) Combien faudrait-il effectuer de répétitions indépendantes de l'algorithme pour que l'amplitude de l'intervalle de confiance au niveau 0.95 soit inférieure à 0.001 ?

Exercice 2.3 Pour chacun des algorithmes suivants calculer l'amplitude des intervalles de confiance de niveau $0,95$ et $0,99$ sur la valeur finale de X, pour $n = 10^4$, puis calculer les valeurs de n pour lesquelles ces amplitudes sont inférieures à 10^{-3} et 10^{-5}.

1.
$X \longleftarrow 0$
Répéter n fois
    $X \longleftarrow X+$Random
finRépéter
$X \longleftarrow X/n$

2.
$X \longleftarrow 0$
Répéter n fois
    $X \longleftarrow X + 10*$Random
finRépéter
$X \longleftarrow X/n$

3.
$X \longleftarrow 0$
Répéter n fois
    Si (Random $< 0,5$) $X \longleftarrow X + 1$
    finSi
finRépéter
$X \longleftarrow X/n$

4.
$X \longleftarrow 0$
Répéter n fois
    Si (Random $< 0,05$) $X \longleftarrow X + 1$
    finSi
finRépéter
$X \longleftarrow X/n$

5.
$X \longleftarrow 0$
Répéter n fois
  Répéter
    $X \longleftarrow X + 1$
  Jusqu'à (Random $< 0,5$)
finRépéter
$X \longleftarrow X/n$

Exercice 2.4 Ecrire un algorithme de simulation par inversion pour les lois suivantes.

1. Loi binomiale $\mathcal{B}(5, 1/2)$.
2. Loi binomiale $\mathcal{B}(5, 9/10)$.
3. Loi binomiale $\mathcal{B}(5, 1/10)$.
4. Loi géométrique de paramètre $p \in]0, 1[$.
5. Loi binomiale négative de paramètres $r \in \mathbb{N}^*$ et $p \in]0, 1[$:

$$P[X = k] = \binom{k-1}{r-1} p^r (1-p)^{k-r} \ , \ \forall k \in \{r, r+1, \ldots\} \ .$$

6. Loi hypergéométrique de paramètres $N \in \mathbb{N}^*$ et $m, n \in \{1, \ldots, N\}$.

$$P[X = k] = \frac{\binom{m}{k}\binom{N-m}{n-k}}{\binom{N}{n}} \ , \ \forall k \in \{0, \ldots, \min(m, n)\} \ .$$

7. Loi sur $\mathbb{Z}$ définie par :

$$P[X = k] = \frac{1}{3} 2^{-|k|} \ , \ \forall k \in \mathbb{Z} \ .$$

8. Loi sur $\mathbb{N}^*$ définie par :

$$P[X = k] = \frac{1}{k^2 + k} \ , \ \forall k \in \mathbb{N}^* \ .$$

Exercice 2.5 Ecrire un algorithme de simulation par rejet à partir de la loi uniforme sur l'ensemble des valeurs prises, pour les lois suivantes (n est un entier > 1 fixé). Pour chacun des algorithmes, calculer le nombre moyen d'appels de Random.

1. Loi binomiale $\mathcal{B}(5, 1/2)$.
2. Loi binomiale $\mathcal{B}(5, 1/10)$.
3. Loi sur $\{1, \ldots, n\}$ définie par :

$$P[X = k] = \frac{n+1}{n(k^2 + k)} \ , \ \forall k \in \{1, \ldots, n\} \ .$$

4. Loi sur $\{-n, \ldots, n\}$ définie par :

$$P[X = k] = \frac{|k|}{n(n+1)} \ , \ \forall k \in \{-n, \ldots, n\} \ .$$

5. Loi sur $\{n, \ldots, 2n-1\}$ définie par :

$$P[X = k] = \frac{2n}{k^2 + k} \ , \ \forall k \in \{n, \ldots, 2n-1\} \ .$$

Exercice 2.6 Ecrire un algorithme de simulation pour les lois suivantes (n est un entier supérieur à 1 fixé). Pour chacun des algorithmes, calculer le nombre moyen d'appels de Random.

1. Loi sur $\{1, \ldots, 2n\}$ définie par :

$$\begin{aligned} P[X = k] &= \frac{1}{2n} && \text{si } k \text{ impair,} \\ &= \frac{k}{2n(n+1)} && \text{si } k \text{ pair.} \end{aligned}$$

2. Loi sur $\{1, \ldots, 3n\}$ définie par :

$$\begin{aligned} P[X = k] &= \frac{1}{6n} && \text{si } k \in \{1, \ldots, n\} \ , \\ &= \frac{1}{3n} && \text{si } k \in \{n+1, \ldots, 2n\} \ , \\ &= \frac{1}{2n} && \text{si } k \in \{2n+1, \ldots, 3n\} \ . \end{aligned}$$

3. Loi sur $\{1, \ldots, 3n\}$ définie par :

$$\begin{aligned} P[X = k] &= \frac{1}{12n} && \text{si } k \in \{1, n+1, 2n+1\} \ , \\ &= \frac{2n-1}{12n(n-1)} && \text{si } k \in \{2, \ldots, n\} \ , \\ &= \frac{4n-1}{12n(n-1)} && \text{si } k \in \{n+2, \ldots, 2n\} \ , \\ &= \frac{6n-1}{12n(n-1)} && \text{si } k \in \{2n+2, \ldots, 3n\} \ . \end{aligned}$$

Exercice 2.7 Soit n un entier fixé. On définit les lois de probabilité p_1 et p_2 sur l'ensemble $\{1, \ldots, n\}$ par :

$$\begin{aligned} p_1(1) &= 1/(2n-1) \ , \\ p_1(k) &= 2/(2n-1) \ , && \forall k = 2, \ldots, n \ . \\ p_2(k) &= 3/(3n-2) \ , && \forall k = 1, \ldots, n-1 \ , \\ p_2(n) &= 1/(3n-2) \ . \end{aligned}$$

On définit la loi de probabilité p sur $\{1, \ldots, 2n\}$ par :

$$\begin{array}{rcll} p(k) & = & (1/3)\, p_1((k+1)/2) & \text{si } k \text{ est impair,} \\ & = & (2/3)\, p_2(k/2) & \text{si } k \text{ est pair.} \end{array}$$

1. Ecrire un algorithme de simulation par rejet pour la loi p_1, à partir de la loi uniforme sur $\{1, \ldots, n\}$. Quel est le nombre moyen d'appels de Random dans cet algorithme ?
2. Même question pour la loi p_2.
3. Ecrire un algorithme de simulation par rejet pour la loi p, à partir de la loi uniforme sur $\{1, \ldots, 2n\}$. Quel est le nombre moyen d'appels de Random dans cet algorithme ?
4. En utilisant les algorithmes des questions 1. et 2., écrire un algorithme de simulation par décomposition pour la loi p. Quel est le nombre moyen d'appels de Random dans cet algorithme ?

Exercice 2.8 Ecrire un algorithme de simulation pour les lois suivantes.

1. Loi sur $\mathbb{N}^2$ définie par :

$$P[X = (k,h)] = \frac{e^{-1}}{k!\, 2^{h+1}} \ , \ \forall (k,h) \in \mathbb{N}^2 \ .$$

2. Loi sur $\mathbb{N}^2$ définie par :

$$P[X = (k,h)] = \frac{e^{-h} h^k}{k!\, 2^{h+1}} \ , \ \forall (k,h) \in \mathbb{N}^2 \ .$$

Exercice 2.9 On dispose d'un écran de $N_x \times N_y$ pixels que l'on souhaite colorier au hasard. Les couleurs sont numérotées de 1 à K, et on souhaite que le nombre de pixels affecté à la couleur i soit exactement n_i ($0 < n_i < N_x N_y$), pour tout $i = 1, \ldots, K$. On propose trois méthodes.

- *Echantillonnage aléatoire séquentiel :*
  Pour chacune des couleurs, puis pour chaque pixel, on décide ou non de le colorier, avec une probabilité dépendant du nombre de pixels déjà coloriés.
- *Echantillonnage aléatoire par rejet :*
  Pour chacune des couleurs, on tire au hasard un pixel de l'écran jusqu'à en trouver un non colorié, et on le colorie, jusqu'à avoir le bon nombre de pixels de chaque couleur.
- *Permutation aléatoire des pixels :*
  On commence par ordonner l'ensemble des pixels dans un tableau de taille $N_x N_y$. On colorie les n_1 premiers pixels de la couleur 1, les n_2 suivants de la couleur 2, etc... On permute ensuite aléatoirement le tableau de pixels.

1. Ecrire l'algorithme correspondant à chacune des trois méthodes.
2. Pour chacun des trois algorithmes, démontrer que toutes les manières possibles de colorier l'écran peuvent être obtenues avec la même probabilité.

3. Comparer les durées d'exécution des trois algorithmes, en fonction des paramètres du problème.
4. Pour $N_x = 640$, $N_y = 480$, $K = 16$ et $n_i = 19200$, $i = 1, \ldots, 16$, quel algorithme choisiriez-vous ?

Exercice 2.10 Soit f une application de $\mathbb{N}^*$ dans l'intervalle $]0, 1[$. On considère la variable aléatoire X en sortie de l'algorithme $\mathcal{A}_f$ suivant.

$X \longleftarrow 0$
Répéter
    $X \longleftarrow X + 1$
Jusqu'à (Random $< f(X)$)

On conviendra que X prend la valeur ∞ si l'algorithme ne se termine pas.

1. Lorsque f est constante, quelle est la loi de la variable X ? Quelle est son espérance ?
2. Pour tout $n \in \mathbb{N}^*$, calculer $Prob[X > n]$ et $Prob[X = n]$.
3. Démontrer que l'algorithme $\mathcal{A}_f$ se termine si et seulement si :

$$\sum_{k=1}^{\infty} f(k) = \infty \, .$$

4. On pose, pour tout entier $k \geq 1$,

$$f(k) = 1 - 2^{-(1/2)^k} \, .$$

   Montrer que $Prob[X = \infty] = 1/2$.
5. On pose, pour tout entier $k \geq 1$,

$$f(k) = 1 - \frac{1}{k+1} \, .$$

   (a) Déterminer la loi de la variable X.
   (b) Calculer la fonction génératrice de X.
   (c) Calculer $\mathbb{E}[X]$ et $Var[X]$.
6. On pose, pour tout entier $k \geq 1$,

$$f(k) = \frac{1}{k+1} \, .$$

   (a) Déterminer la loi de la variable aléatoire X.
   (b) Calculer la fonction génératrice de X.
   (c) Montrer que X n'admet pas d'espérance. Qu'en concluez-vous pour le temps d'exécution de l'algorithme ?

7. On pose, pour tout entier $k \geq 1$,

$$f(k) = \frac{2k+1}{k^2+2k+1} .$$

   (a) Montrer que l'algorithme $\mathcal{A}_f$ se termine.

   (b) Quelle est la loi de la variable X ?

   (c) Calculer $I\!E[X]$ et montrer que $Var[X]$ n'existe pas.

Exercice 2.11 Ecrire un algorithme de simulation par inversion pour les lois dont les densités suivent.

1.
$$f(x) = \alpha\lambda x^{\alpha-1} e^{-\lambda x^\alpha} \mathbb{1}_{I\!R^+}(x) .$$
(Densité de la loi de Weibull de paramètres $\alpha > 0$ et $\lambda > 0$).

2.
$$f(x) = \alpha x^{\alpha-1} \mathbb{1}_{[0,1]}(x) .$$
(Densité de la loi Béta $\mathcal{B}(\alpha, 1)$, le paramètre α est > 0.)

3.
$$f(x) = \alpha\lambda^\alpha x^{-\alpha-1} \mathbb{1}_{[\lambda,+\infty[}(x) .$$
(Densité de la loi de Pareto de paramètres $\alpha > 0$ et $\lambda > 0$).

Exercice 2.12 Déterminer la loi de la variable aléatoire X, en sortie des algorithmes suivants :

1. $X \longleftarrow 1.6*\mathsf{Random}$
2. Répéter
   $X \longleftarrow 2*\mathsf{Random}$
   Jusqu'à $(X < 1.6)$

Quelle est la loi du nombre de passages dans la boucle pour le deuxième algorithme ? Lequel des deux algorithmes vous semble le plus rapide ?

Exercice 2.13 Déterminer la fonction de répartition puis la densité s'il y a lieu, de la variable aléatoire X en sortie des algorithmes suivants :

1. $X \longleftarrow \mathsf{Sqrt}(\mathsf{Random})$
2. $X \longleftarrow 1/\mathsf{Random}$
3. $X \longleftarrow (2*\mathsf{Random}-1)^2$
4. $N \longleftarrow \mathsf{Int}(\mathsf{Random}*2)$
   $X \longleftarrow N*\mathsf{Random}$
5. $N \longleftarrow \mathsf{Int}(\mathsf{Random}*2+1)$
   $X \longleftarrow N*\mathsf{Random}$
6. Choix$\longleftarrow$Random
   Si (Choix < 0.4)
   Alors $X \longleftarrow$ Random
   Sinon $X \longleftarrow$ Random $+1$
   finSi

Exercice 2.14 Soit X une variable aléatoire dont la loi est symétrique par rapport à l'origine (X et $-X$ suivent la même loi). Soit Y une variable aléatoire de même loi que $|X|$ et S une autre variable aléatoire indépendante de Y prenant les valeurs -1 et $+1$ avec probabilité $1/2$. Montrer que $Z = S.Y$ suit la même loi que X.
Application : donner un algorithme de simulation pour la loi de densité f définie par :

$$f(x) = \frac{1}{2}\exp(-|x|) \ , \ \forall x \in I\!R \ .$$

Exercice 2.15 On considère la densité de probabilité f suivante.

$$f(x) = (1 - |x|)\mathbb{1}_{[-1,1]}(x) \ .$$

1. Utiliser la méthode d'inversion pour donner un algorithme de simulation de la loi de densité f.
2. Ecrire un algorithme de simulation par rejet à partir de la loi uniforme sur $[-1, 1]$.
3. Montrer que X a pour densité f en sortie de l'algorithme suivant :

   $X \longleftarrow$ Random
   $Y \longleftarrow$ Random
   Si $(Y < X)$ alors $X \longleftarrow Y$ finSi
   Si (Random < 0.5) alors $X \longleftarrow -X$ finSi

4. Lequel des trois algorithmes est le plus rapide ?

Exercice 2.16 On considère la densité de probabilité f suivante.

$$f(x) = \frac{2}{3}(x\mathbb{1}_{[0,1]}(x) + \mathbb{1}_{[1,2]}(x)) \ .$$

1. Utiliser la méthode d'inversion pour donner un algorithme de simulation de la loi de densité f.
2. Ecrire un algorithme de simulation par rejet à partir de la loi uniforme sur $[0, 2]$.
3. Soient Y et Z deux V.A. indépendantes telles que Y suit la loi uniforme sur $[0, 1]$ et Z suit la loi uniforme sur $[0, 2]$. Quelle est la fonction de répartition de la variable aléatoire $S = \max\{Y, Z\}$?
4. Montrer que la variable aléatoire X a pour densité f en sortie de l'algorithme suivant :

```
U ⟵ Random
Si (U < 1/3 )
      alors X ⟵ 3 * U + 1
      sinon
            Y ⟵ 3 * (U − 1/3)/2
            Z ⟵Random *2
            Si (Y > Z)
                  alors X ⟵ Y
```

sinon $X \longleftarrow Z$
finSi
finSi

5. Lequel des algorithmes proposés est le plus rapide ?

Exercice 2.17 1. Donner un algorithme de simulation par inversion pour la loi exponentielle de paramètre $\lambda > 0$, de densité :

$$f_\lambda(x) = \lambda e^{-\lambda x} \mathbb{1}_{\mathbb{R}^+}(x) \ .$$

2. Donner un algorithme de simulation par inversion pour la loi géométrique de paramètre $1/2$, définie pour tout $k \in \mathbb{N}^*$ par :

$$p(k) = \frac{1}{2^k} \ .$$

3. On considère la loi de probabilité sur $\mathbb{R}^+$ définie par la densité f suivante.

$$f(x) = \frac{e^{-x}/2}{(1-e^{-x}/2)^2} \mathbb{1}_{\mathbb{R}^+}(x) \ .$$

Calculer la fonction de répartition de cette loi. En déduire un algorithme de simulation par inversion pour la loi de densité f.

4. Montrer que pour tout $x \in \mathbb{R}^+$, $f(x) \leq 2e^{-x}$. En déduire un algorithme de simulation par rejet pour la loi de densité f, à partir de la loi exponentielle de paramètre 1. Quel est le nombre moyen d'appels de Random dans cet algorithme ?
5. Montrer que pour tout $x \in \mathbb{R}^+$, on a :

$$f(x) = \sum_{k=1}^{+\infty} \frac{1}{2^k} \ ke^{-kx} \mathbb{1}_{\mathbb{R}^+}(x) \ .$$

En déduire un algorithme de simulation par décomposition pour la loi de densité f, utilisant les algorithmes des questions 1 et 2.

Exercice 2.18 L'algorithme suivant calcule une valeur approchée de

$$I \ = \ \int_{[0,1]^2} xy \, dxdy \ .$$

$I \longleftarrow 0$
Répéter n fois
  $X \longleftarrow$Random
  $Y \longleftarrow$Random
  $U \longleftarrow$Random
  Si $(U < X * Y)$ Alors $I \longleftarrow I + 1$
  finSi
finRépéter
$I \longleftarrow I/n$

1. Calculer le nombre d'appels de Random nécessaires pour que l'amplitude de l'intervalle de confiance de niveau $0,95$ soit inférieure à 10^{-3}.
2. On considère le découpage de $\Delta = [0,1]^2$ en 4 carrés

$$\Delta_1 = [0,0.5]\times[0,0.5] \ , \ \ldots \ , \ \Delta_4 = [0.5,1]\times[0.5,1] \ .$$

   Pour $i = 1,\ldots,4$, soit μ_i le maximum de la fonction xy sur Δ_i. On définit le domaine D' par

$$D' \ = \ \bigcup_{i=1}^{4} \Delta_i \times [0,\mu_i] \ .$$

   Ecrire l'algorithme de simulation par rejet pour le calcul de I à partir du domaine D'.
3. Calculer le nombre d'appels de Random nécessaires pour que l'amplitude de l'intervalle de confiance de niveau $0,95$ soit inférieure à 10^{-3}.
4. Mêmes questions pour un découpage irrégulier en 4 carrés

$$\Delta_1 = [0,u]\times[0,u] \ , \ \ldots \ , \ \Delta_4 = [u,1]\times[u,1] \ .$$

5. Quelle valeur de u est optimale ?
6. Reprendre l'exercice pour un découpage de Δ en 9 carrés.

3 Méthodes markoviennes à temps fini

3.1 Simulation des chaînes de Markov

3.1.1 Définition algorithmique

Une chaîne de Markov est classiquement définie comme une suite de variables aléatoires pour laquelle la meilleure prédiction que l'on puisse faire pour l'étape $n+1$ si on connaît toutes les valeurs antérieures est la même que si on ne connaît que la valeur à l'étape n (le futur et le passé sont indépendants conditionnellement au présent). Nous partons ici d'une définition moins classique, mais plus proche des applications.

Définition 3.1 *Soit E un espace mesurable. Une chaîne de Markov sur E est une suite de variables aléatoires (X_n), $n \in \mathbb{N}$ à valeurs dans E telle qu'il existe :*

1. *une suite (U_n), $n \in \mathbb{N}$ de variables aléatoires indépendantes et de même loi, à valeurs dans un espace probabilisé $\mathcal{U}$,*
2. *une application mesurable Φ de $\mathbb{N} \times E \times \mathcal{U}$ dans E telle que :*

$$\forall n \in \mathbb{N}\,, \qquad X_{n+1} \;=\; \Phi(n, X_n, U_n)\;.$$

On distingue plusieurs cas particuliers.

- Si l'application Φ ne dépend pas de n, la chaîne est dite *homogène.*
- Si l'application Φ ne dépend pas de x, la chaîne est une suite de variables indépendantes. Si Φ ne dépend ni de n ni de x, c'est une suite de VAIID.
- Si l'application Φ ne dépend pas de u, Φ définit un système itératif. La chaîne est une suite récurrente (déterministe si sa valeur initiale est déterministe).

Toutes les chaînes de Markov que nous considérons ici sont homogènes. On peut toujours passer du cas non homogène au cas homogène en remplaçant X_n par le couple (n, X_n).

C'est évidemment aux appels d'un générateur pseudo-aléatoire qu'il faut penser pour la suite (U_n) de la définition 3.1. En pratique une chaîne de Markov est simulée de manière itérative comme le dit la définition 3.1. Une initialisation dans E est d'abord choisie (aléatoire ou non). Puis chaque nouveau pas est simulé selon une loi de probabilité dépendant du point atteint précédemment. Cette simulation utilise un ou plusieurs appels de Random successifs, qui constituent la variable U_n.

En toute rigueur, les chaînes de Markov au sens de la définition 3.1 devraient s'appeler "chaînes de Markov simulables". Elles vérifient la propriété suivante, dite "*propriété de Markov*".

Proposition 3.2 *Soit $(X_n), n \in \mathbb{N}$ une chaîne de Markov. Pour tout $n \geq 0$ et pour toute suite d'états $i_0, \ldots, i_n \in E$, la loi conditionnelle de X_{n+1} sachant "$X_0 = i_0, \ldots, X_n = i_n$" est égale à la loi conditionnelle de X_{n+1} sachant "$X_n = i_n$".*

Démonstration : Notons $\mathbb{P}$ la loi de probabilité conjointe de X_0 et de la suite (U_n). D'après la définition 3.1, U_n est indépendante de $X_0, \ldots, X_n$. Pour tout sous ensemble

mesurable B de E, on a :

$$\begin{aligned} I\!P[X_{n+1} \in B \mid X_0 = i_0, \ldots, X_n = i_n] &= I\!P[\Phi(X_n, U_n) \in B \mid X_0 = i_0, \ldots, X_n = i_n] \\ &= I\!P[\Phi(i_n, U_n) \in B] \\ &= I\!P[X_{n+1} \in B \mid X_n = i_n] \, . \end{aligned}$$

□

C'est cette propriété d'"oubli du passé" qui constitue la définition classique des chaînes de Markov. Il est naturel de se demander s'il existe des chaînes de Markov, au sens de la proposition 3.2, qui ne soient pas simulables. Il n'en existe pas si E est dénombrable, ou si $E = I\!R^d$, muni de sa tribu de boréliens. On n'en rencontrera donc jamais en pratique.

Exemple : Marches aléatoires.

Soit $(U_n), n \in I\!N$ une suite de variables aléatoires indépendantes et de même loi sur $I\!R^d$. La suite de variables aléatoires $(X_n), n \in I\!N$ définie par $X_0 \in I\!R^d$ et

$$\forall n \, , \qquad X_{n+1} \; = \; X_n + U_n \, ,$$

est une chaîne de Markov. Comme cas particulier, si $X_0 \in Z\!\!Z$ et

$$I\!P[U_n = -1] = I\!P[U_n = 1] = 1/2 \, ,$$

on obtient la marche aléatoire symétrique sur $Z\!\!Z$ (jeu de Pile ou Face).

Si U_n suit la loi normale $\mathcal{N}(0, h)$, on obtient une discrétisation du mouvement brownien standard sur $I\!R$.

Ces deux exemples s'étendent facilement en dimension quelconque (marche aléatoire symétrique sur $Z\!\!Z^d$ et mouvement brownien standard dans $I\!R^d$). Plus généralement, soit $(G, *)$ un groupe topologique quelconque, muni de sa tribu des boréliens. Soit P une loi de probabilité sur G, et (U_n) une suite de variables aléatoires de même loi π sur G. La suite de variables aléatoires définie par $X_0 \in G$ et pour tout $n \geq 0$:

$$X_{n+1} = X_n * U_n \, ,$$

est une chaîne de Markov sur G, dite "*marche aléatoire de pas* π". Les marches aléatoires sur les groupes constituent un cas particulier important de chaîne de Markov.

3.1.2 Espace d'états fini ou dénombrable

Lorsque $E = \{i, j, \ldots\}$ est un ensemble fini ou dénombrable, la famille de lois de probabilité des variables aléatoires $\Phi(n, i, U_n)$ (*cf.* définition 3.1) avec laquelle on tire le pas $n+1$ à partir du pas n, est habituellement notée sous forme matricielle. Si la chaîne est homogène, cette famille ne dépend pas de n. Dans ce cas, on note p_{ij} la probabilité de choisir l'état j à partir de l'état i :

$$p_{ij} \; = \; I\!P[\Phi(i, U_n) = j] \; = \; I\!P[X_{n+1} = j \mid X_n = i] \, , \quad \forall i, j \in E \, .$$

Dans la relation ci-dessus, $\mathbb{P}$ désigne encore la loi de probabilité conjointe de X_0 et de la suite (U_n).
La probabilité p_{ij} porte le nom de "*probabilité de transition de i à j*". La matrice

$$P \;=\; (p_{ij})_{i,j\in E} \;,$$

est la *matrice de transition* de la chaîne. C'est une matrice à coefficients positifs ou nuls telle que la somme des éléments d'une même ligne vaut 1. Comme nous le verrons dans les exemples des paragraphes suivants, il arrive fréquemment dans les applications que pour un état i donné, le nombre d'états j directement accessibles depuis i (tels que $p_{ij} > 0$) soit faible. La matrice de transition est alors très creuse (elle contient beaucoup de zéros). Il est plus économique de résumer les probabilités de transitions par le *diagramme de transition*. C'est un graphe orienté et pondéré, dont l'ensemble des sommets est E. Une arête de poids p_{ij} va de i à j si $p_{ij} > 0$.

L'algorithme de simulation d'une chaîne de Markov homogène de matrice de transition P est le suivant.

$n \longleftarrow 0$
Initialiser X
Répéter
    $i \longleftarrow X$ {*état présent*}
    choisir j avec probabilité p_{ij}
    $X \longleftarrow j$ {*état suivant*}
    $n \longleftarrow n+1$
Jusqu'à (arrêt de la simulation)

L'algorithme ci-dessus correspond bien à la définition 3.1, dans la mesure où les choix successifs sont effectués à l'aide d'appels de Random renouvelés à chaque itération (considérés comme indépendants des précédents). Supposons par exemple que la loi $(p_{ij})_{j\in E}$ soit simulée par inversion. Notons

- U_n le n-ième appel de Random.
- Φ l'application de $E \times [0,1]$ dans E qui au couple (i,u) associe l'inverse de la fonction de répartition de la loi $(p_{ij})_{j\in E}$, évalué en u.

L'algorithme calcule bien

$$X_{n+1} = \Phi(X_n, U_n) \;.$$

Ceci a une portée plutôt théorique. Il ne faudrait pas en déduire que c'est forcément par inversion que l'on doit simuler la loi $(p_{ij})_{j\in E}$. Dans certains cas un autre type de simulation (par exemple par rejet ou décomposition) pourra s'avérer plus efficace.

Exemple : Voici une matrice de transition sur $E = \{a, b, c, d, e\}$.

$$P = \begin{array}{c|ccccc} & a & b & c & d & e \\ \hline a & 0.2 & 0.2 & 0.2 & 0.2 & 0.2 \\ b & 0 & 0.2 & 0.3 & 0 & 0.5 \\ c & 0.3 & 0.3 & 0 & 0.4 & 0 \\ d & 0 & 0.3 & 0.3 & 0.3 & 0.1 \\ e & 0 & 1 & 0 & 0 & 0 \end{array}$$

L'algorithme ci-après simule une chaîne de Markov de matrice de transition P. Il n'est pas optimal mais il illustre quelques méthodes standard.

```
Tableau E = [a, b, c, d, e]
n ⟵ 0
Initialiser X
Répéter
    i ⟵ X                {état présent}
    Selon i
        i = a : j ⟵ E[Int(Random * 5) + 1]

        i = b : Choix ⟵ Random
                Si (Choix < 0.5) alors j ⟵ e
                    sinon Si (Choix < 0.8) alors j ⟵ c
                                sinon j ⟵ b
                        finSi
                finSi

        i = c : Choix ⟵ Random
                Si (Choix > 0.6) alors j ⟵ d
                    sinon j ⟵ E[Int(Random * 2) + 1]
                finSi

        i = d : Répéter
                Test ⟵ Vrai
                j ⟵ E[Int(Random * 4) + 2]
                Si j = e alors
                    Si (Random > 1/3) alors   Test ⟵ Faux   finSi
                finSi
            Jusqu'à (Test=Vrai)

        i = e : j ⟵ b

    finSelon
    X ⟵ j                {état suivant}
    n ⟵ n+1
Jusqu'à (arrêt de la simulation)
```

3.1.3 Relations algébriques

La loi d'une chaîne de Markov (X_n) est entièrement déterminée par la donnée de la loi de X_0 et de la matrice de transition P, au sens où pour tout n, la loi conjointe de $(X_0, \ldots, X_n)$ s'exprime en fonction de la loi de X_0 et de P. La matrice de transition P contient toute l'information sur l'évolution de la chaîne, qui est commune à toutes les conditions initiales. Cette information se lit sur les puissances successives de P.

Proposition 3.3 *Soit (X_n) une chaîne de Markov homogène de matrice de transition P. Alors*

1. *Pour toute suite d'états $i_0, i_1, \ldots, i_n$ de E,*

$$\mathbb{P}[X_0 = i_0 \text{ et } \ldots \text{ et } X_n = i_n] \;=\; \mathbb{P}[X_0 = i_0]\, p_{i_0 i_1} \ldots p_{i_{n-1} i_n} \;.$$

2. *Pour tout entier n, notons*

$$p_{ij}^{(n)} \;=\; \mathbb{P}[X_n = j \,|\, X_0 = i] \;=\; \mathbb{P}[X_{m+n} = j \,|\, X_m = i] \;, \quad \forall m \;.$$

C'est la probabilité de transition de i à j en n pas. On a

$$\left(p_{ij}^{(n)}\right)_{i,j \in E} \;=\; P^n \;.$$

3. *Notons $p(n)$ la loi de X_n :*

$$p(n) = \left(\mathbb{P}[X_n = i]\right)_{i \in E} \;.$$

On a

$$p(n) \;=\; {}^t\!P^n p(0) \;.$$

Démonstration : Nous montrons uniquement le premier point. Les suivants s'en déduisent de manière immédiate. La formule est vraie pour $n = 0$. Supposons-la vraie pour n. Si

$$\mathbb{P}[X_0 = i_0 \text{ et } \ldots \text{ et } X_n = i_n] = 0 \;,$$

alors pour tout i_{n+1},

$$\mathbb{P}[X_0 = i_0 \text{ et } \ldots \text{ et } X_{n+1} = i_{n+1}] = 0 \;.$$

Sinon :

$$\begin{aligned}
& \mathbb{P}[X_0 = i_0 \text{ et } \ldots \text{ et } X_{n+1} = i_{n+1}] \\
= \;& \mathbb{P}[X_{n+1} = i_{n+1} \,|\, X_0 = i_0 \text{ et } \ldots \text{ et } X_n = i_n] \mathbb{P}[X_0 = i_0 \text{ et } \ldots \text{ et } X_n = i_n] \\
= \;& \mathbb{P}[X_{n+1} = i_{n+1} \,|\, X_n = i_n] \mathbb{P}[X_0 = i_0 \text{ et } \ldots \text{ et } X_n = i_n] \\
= \;& p_{i_n i_{n+1}} \mathbb{P}[X_0 = i_0 \text{ et } \ldots \text{ et } X_n = i_n] \;.
\end{aligned}$$

Le résultat est donc vrai à l'ordre $n+1$. □

Dans la définition des probabilités de transition en n pas, on peut comprendre la notation $I\!P[X_n = j \,|\, X_0 = i]$ soit comme une probabilité conditionnelle, soit comme une probabilité relative à la loi des U_n, quand l'initialisation est fixée à $X_0 = i$.

On peut voir l'évolution en loi de la suite (X_n) comme un système dynamique linéaire dont ${}^tP$ est la matrice d'évolution.

$$p(n+1) \;=\; {}^tP p(n)\ .$$

Il est assez naturel au vu des relations ci-dessus que des techniques numériques utilisent l'évolution des chaînes de Markov pour la résolution de systèmes linéaires. Ces techniques font l'objet du paragraphe suivant.

3.2 Résolution de systèmes linéaires

3.2.1 Puissances de matrices

Considérons un système d'équations de taille d, mis sous la forme :

$$(I - A)z \;=\; b\ ,$$

où $A = (a_{ij}) \in \mathcal{M}_{d\times d}(I\!R)\ ,\ b \in I\!R^d$. On suppose que la matrice $I - A$ est inversible et que :

$$\lim_{k\to+\infty} I + A + \cdots + A^k \;=\; (I-A)^{-1}\ .$$

C'est le cas en particulier sous la condition suffisante suivante :

$$\max_i \sum_{j=1}^{d} |a_{ij}| \;<\; 1\ .$$

On peut donc approcher la solution z du système par la suite de vecteurs $(z^{(k)})$ définie par :

$$z^{(k)} \;=\; (I + A + \cdots + A^k)b\ .$$

Nous allons définir, à partir des trajectoires d'une chaîne de Markov, une variable aléatoire dont l'espérance est égale au produit scalaire ${}^t\varphi z^{(k)}$, où $\varphi = (\varphi_i)$ est un vecteur de $I\!R^d$. Par exemple si $\varphi = 1\!\!1_{\{i\}}$, le produit scalaire est la i-ième coordonnée de $z^{(k)}$.

Considérons une chaîne de Markov $(X_m), m \in I\!N$ à valeurs dans $\{1,\ldots,d\}$ dont la loi est définie par

1. La loi de X_0 : $(p_i)_{i\in E}$,
2. La matrice de transition $P = (p_{ij})_{i,j\in E}$.

On définit la suite de variables aléatoires (W_m) à partir de la chaîne (X_m) par

$$X_0 = i_0, \ldots, X_m = i_m \quad\Longrightarrow\quad W_m = \frac{\varphi_{i_0}}{p_{i_0}} \frac{a_{i_0 i_1}\ldots a_{i_{m-1}i_m}}{p_{i_0 i_1}\ldots p_{i_{m-1}i_m}} b_{i_m}\ .$$

Remarquons qu'en pratique la trajectoire $i_0,\ldots,i_m$ ne peut être suivie par la chaîne que si sa probabilité est strictement positive, à savoir $p_{i_0}\, p_{i_0 i_1}\ldots p_{i_{m-1}i_m} > 0$.

Proposition 3.4 *Supposons que :*

1. $\forall i \quad , \quad \varphi_i \neq 0 \implies p_i > 0$,
2. $\forall i, j \quad , \quad a_{ij} \neq 0 \implies p_{ij} > 0$.

Alors :

$$I\!E[W_m] \;=\; {}^t\varphi A^m b \;.$$

Démonstration : L'idée de la définition de W_m est de réaliser une moyenne des facteurs $\varphi_{i_0} a_{i_0 i_1} \ldots a_{i_{m-1} i_m}$ qui interviennent dans le calcul de ${}^t\varphi A^m$, au travers des trajectoires de la chaîne X_m. Il faut pour cela que tous les facteurs non nuls puissent effectivement être atteints, et c'est la signification des hypothèses de cette proposition. L'espérance de W_m vau :t

$$\begin{aligned} I\!E[W_m] &= \sum \frac{\varphi_{i_0}}{p_{i_0}} \frac{a_{i_0 i_1} \ldots a_{i_{m-1} i_m}}{p_{i_0 i_1} \ldots p_{i_{m-1} i_m}} b_{i_m} \; p_{i_0} p_{i_0 i_1} \ldots p_{i_{m-1} i_m} \\ &= \sum \varphi_{i_0} a_{i_0 i_1} \ldots a_{i_{m-1} i_m} b_{i_m} \;, \end{aligned}$$

la somme portant sur les trajectoires $i_0, \ldots, i_m$ de probabilité non nulle, c'est-à-dire telles que :

$$p_{i_0} p_{i_0 i_1} \ldots p_{i_{m-1} i_m} > 0 \;.$$

Or

$${}^t\varphi A^m b \;=\; \sum \varphi_{i_0} a_{i_0 i_1} \ldots a_{i_{m-1} i_m} b_{i_m} \;,$$

où la somme porte sur les $i_0, \ldots, i_m$ tels que le produit correspondant soit non nul. D'où le résultat. □

Pour calculer une valeur approchée de ${}^t\varphi z^{(k)}$, il ne reste plus qu'à définir :

$$Z_k \;=\; \sum_{m=0}^{k} W_m \;,$$

dont l'espérance est :

$$I\!E[Z_k] \;=\; {}^t\varphi z^{(k)} \;.$$

Cette espérance sera approchée par la moyenne empirique des valeurs prises par Z_k sur un grand nombre de trajectoires indépendantes de longueur k de la chaîne (X_m). Voici l'algorithme correspondant.

$S \longleftarrow 0$
Répéter n fois
    $m \longleftarrow 0$
    Choisir i_0 dans $\{1, \ldots, d\}$ avec probabilité p_{i_0}
    $X \longleftarrow i_0$
    $W \longleftarrow \varphi_{i_0} / p_{i_0}$
    $Z \longleftarrow \varphi_{i_0} / p_{i_0} * b_{i_0}$
    Répéter

$i \longleftarrow X$
Choisir j avec probabilité p_{ij}
$X \longleftarrow j$
$W \longleftarrow W * a_{ij}/p_{ij}$
$Z \longleftarrow Z + W * b_j$
$m \longleftarrow m + 1$
Jusqu'à $(m = k)$
$S \longleftarrow S + Z$
finRépéter
$S \longleftarrow S/n$

Un caractère intéressant de cet algorithme est qu'on peut utiliser les mêmes trajectoires pour calculer simultanément plusieurs produits scalaires, par exemple toutes les coordonnées de z. Il est même possible de calculer toute la matrice $(I - A)^{-1}$. Des variantes de cet algorithme ont été proposées qui n'utilisent qu'une seule trajectoire, suivie suffisamment longtemps (voir Rubinstein [81] p. 158-169).

Comme on l'aura constaté, on dispose d'une grande latitude pour choisir la loi initiale ainsi que les probabilités de transition. La seule contrainte que nous ayons donnée est que les p_i doivent s'annuler au plus aussi souvent que les φ_i, et les p_{ij} que les a_{ij}. La question se pose donc d'optimiser la méthode par un choix judicieux des p_i et p_{ij}. Le problème est que la variance de Z_k est impossible à calculer en général. Tout au plus peut-on donner quelques conseils de bon sens. La chaîne de Markov la plus rapide à simuler est la suite de variables indépendantes de loi uniforme sur $\{1, \ldots, d\}$. Elle correspond au choix

$$p_i \;=\; p_{ij} \;=\; \frac{1}{d}\,, \quad \forall i, j\,.$$

Ce n'est pas nécessairement elle qu'il faut utiliser. En ce qui concerne la loi initiale (p_i), le choix dépend de l'objectif. Si on souhaite calculer toutes les coordonnées de $z^{(k)}$, la loi uniforme sera le meilleur choix. Si c'est une seule coordonnée, c'est de l'indice correspondant qu'il faudra faire partir toutes les trajectoires. Pour la matrice de transition, il est difficile de donner des indications précises. Quand la matrice A est pleine (tous ses coefficients sont non nuls), le choix $p_{ij} = 1/d$ sera probablement un bon choix. C'est rarement le cas en pratique. Les gros systèmes linéaires proviennent de la discrétisation de problèmes intégro-différentiels pour lesquels les matrices sont assez creuses. Dans ce cas, c'est un choix algorithmiquement raisonnable que d'adapter la matrice de transition à la matrice du système, en annulant p_{ij} quand $a_{ij} = 0$. Nous en verrons un exemple dans le paragraphe suivant avec le problème (3.1). Signalons enfin que l'algorithme, pour être implémenté efficacement devra être quelque peu modifié. En particulier les quotients a_{ij}/p_{ij} seront calculés en début de programme, et non pas à chaque passage dans la boucle principale. De plus il sera judicieux de faire en sorte qu'un maximum d'entre eux soient égaux à ± 1 de manière à éviter des multiplications. Dans le cas où tous les p_{ij} ne s'annuleraient pas en même temps que les a_{ij}, il faut éviter de continuer à simuler des trajectoires pour lesquelles le facteur W_m est nul (modifier le test d'arrêt de chaque trajectoire).

3.2.2 Utilisation d'un état absorbant

Une variante de la méthode précédente permet d'atteindre en moyenne ${}^t\varphi z$, et non pas seulement ${}^t\varphi z^{(k)}$. L'astuce consiste à introduire un état absorbant pour la chaîne (X_m) que l'on simule. Soit $P = (p_{ij})$ une matrice de transition sur :

$$E \;=\; \{1, \ldots, d, a\} \;,$$

telle que :

$$\forall i = 1, \ldots, d \;, \quad p_{ai} = 0 \quad \text{et} \quad \exists m > 0 \;, \quad p_{ia}^{(m)} > 0 \;.$$

Une chaîne de Markov (X_m) de matrice P admet a comme état absorbant : toute trajectoire, quel que soit son point de départ, atteindra a au bout d'un nombre fini de pas, et n'en partira plus.

Le point de départ de la chaîne est choisi comme précédemment avec la loi $(p_i)_{1 \le i \le d}$. Toutes les trajectoires de la chaîne sont donc de la forme $(i_0, i_1, \ldots, i_m, a, a \ldots)$, avec $i_\ell \in \{1, \ldots, d\}$ pour tout ℓ entre 0 et m.

A la chaîne (X_m) on associe la variable aléatoire Z' définie par :

$$X_0 = i_0, \ldots, X_m = i_m, \, X_{m+1} = a \quad \Longrightarrow \quad Z' = \frac{\varphi_{i_0}}{p_{i_0}} \, \frac{a_{i_0 i_1} \ldots a_{i_{m-1} i_m}}{p_{i_0 i_1} \cdots p_{i_{m-1} i_m}} \, \frac{b_{i_m}}{p_{i_m a}} \;.$$

Proposition 3.5 *Supposons que :*

1. $\forall i \quad , \quad \varphi_i \neq 0 \implies p_i > 0 \;,$
2. $\forall i, j \quad , \quad a_{ij} \neq 0 \implies p_{ij} > 0 \;,$
3. $\forall i \quad , \quad b_i \neq 0 \implies p_{ia} > 0 \;.$

Alors

$$I\!E[Z'] \;=\; {}^t\varphi z \;.$$

Démonstration : L'espérance de Z' vaut :

$$\begin{aligned} I\!E[Z'] \;&=\; \sum_{m=0}^{+\infty} \sum \frac{\varphi_{i_0}}{p_{i_0}} \, \frac{a_{i_0 i_1} \ldots a_{i_{m-1} i_m}}{p_{i_0 i_1} \cdots p_{i_{m-1} i_m}} \, \frac{b_{i_m}}{p_{i_m a}} \; p_{i_0} p_{i_0 i_1} \cdots p_{i_{m-1} i_m} p_{i_m a} \;, \\ &=\; \sum_{m=0}^{+\infty} \sum \varphi_{i_0} a_{i_0 i_1} \ldots a_{i_{m-1} i_m} b_{i_m} \;, \end{aligned}$$

la seconde somme portant sur les trajectoires $i_0, \ldots, i_m$ de probabilité non nulle, c'est-à-dire telles que :

$$p_{i_0} p_{i_0 i_1} \cdots p_{i_{m-1} i_m} > 0 \;.$$

Or :

$$\begin{aligned} {}^t\varphi z \;&=\; {}^t\varphi \sum_{m=0}^{+\infty} A^m b \\ &=\; \sum_{m=0}^{+\infty} \sum \varphi_{i_0} \, a_{i_0 i_1} \ldots a_{i_{m-1} i_m} \, b_{i_m} \;, \end{aligned}$$

où la seconde somme porte sur les $i_0, \ldots, i_m$ tels que le produit correspondant soit non nul. D'où le résultat. □

L'espérance sera comme d'habitude approchée par la moyenne empirique des valeurs prises par Z' sur un grand nombre de trajectoires indépendantes de la chaîne (X_m). L'écriture de l'algorithme correspondant est laissée au lecteur en exercice. Les remarques déjà faites sur l'optimisation de l'algorithme précédent restent vraies pour celui-ci.

3.3 Problèmes différentiels

Les problèmes différentiels issus de la physique sont à l'origine de l'introduction des méthodes de Monte-Carlo. L'idée consiste encore à exprimer la solution du problème à résoudre comme l'espérance d'une certaine variable aléatoire, fonction d'une trajectoire d'une chaîne de Markov. On obtient alors une approximation de la solution, comme dans les paragraphes précédents, en calculant la moyenne empirique des valeurs prises par cette variable aléatoire sur un grand nombre de trajectoires indépendantes.

Nous allons voir que l'utilisation du hasard est ici beaucoup plus qu'une astuce de programmation. Il y a un lien mathématique profond entre certains systèmes d'équations aux dérivées partielles, les problèmes de physique dont ils sont issus, et les processus de diffusion. Nous ne développerons ce lien que pour certaines de ses conséquences numériques.

Les quelques exemples que nous donnons ici ne constituent qu'une introduction à un sujet en plein développement. Nous commençons par l'exemple basique du problème de la chaleur.

3.3.1 Le problème de la chaleur

Appliquons la méthode du paragraphe 3.2.2 à la discrétisation d'un problème de Dirichlet simple. Soit D le carré unité ouvert, ∂D son bord. Le problème consiste à trouver une fonction harmonique sur le carré, de valeur fixée sur le bord.

$$\left\{ \begin{array}{rcll} \Delta z(x) & = & 0 & , \quad \forall x \in D \\ z(x) & = & b(x) & , \quad \forall x \in \partial D \ . \end{array} \right. \tag{3.1}$$

Choisissons un pas de discrétisation h (inverse d'un entier), et examinons la version discrétisée du problème sur une grille de pas h.

$$G \ = \ \{x \in D \ : \ x = (hi, hj) \, , \ i, j = 1, \ldots, d\} \ ,$$

où $d = (1/h - 1)$. Cette version discrétisée est un système linéaire de d^2 équations à d^2 inconnues. A chaque point x de la discrétisation correspond une équation, qui fait intervenir les quatre voisins y_1, y_2, y_3, y_4 de x sur la grille de discrétisation. Les équations des points dont un des voisins est sur le bord étant particulières, on obtient des équations de trois types.

$$\left\{\begin{array}{lll} 1) \quad z(x) - \frac{1}{4}(z(y_1) + z(y_2) + z(y_3) + z(y_4)) & = & 0 \\ 2) \quad z(x) - \frac{1}{4}(z(y_1) + z(y_2) + z(y_3)) & = & \frac{1}{4} b(y_4) \\ 3) \quad z(x) - \frac{1}{4}(z(y_1) + z(y_2)) & = & \frac{1}{4}(b(y_3) + b(y_4)) \; . \end{array}\right.$$

Les équations du type 1 sont celles correspondant aux points qui ont leurs 4 voisins à l'intérieur du carré, celles du type 2 aux points qui ont un voisin (y_4) sur le bord, celles du type 3 aux 4 points qui ont deux voisins (y_3 et y_4) sur le bord. Ce système est bien mis sous la forme $(I - A)z = b$. La technique d'approximation par chaîne de Markov avec état absorbant (paragraphe 3.2.2) s'applique ici de manière particulièrement naturelle, puisqu'on peut prendre pour probabilités de transition p_{xy} les coefficients a_{xy} du système.

$$\begin{array}{rcll} p_{xy} & = & \frac{1}{4} & \forall x, y \in G \; , \; \|x - y\| = h \; , \\ & = & 0 & \text{dans tous les autres cas .} \end{array}$$

(La norme $\|.\|$ est la norme euclidienne). Supposons que l'on souhaite obtenir la valeur de la solution au point x_0 de la discrétisation. Le point de départ de la chaîne simulée sera x_0. L'évolution de la chaîne est la suivante. Pour chaque point de la discrétisation ayant ses 4 voisins à l'intérieur du carré, l'un de ces 4 voisins est choisi au hasard comme point suivant et la simulation continue. Si le point de départ a un voisin sur le bord, alors avec probabilité 1/4 la trajectoire s'arrête et la valeur retournée pour Z' est la valeur de b sur ce voisin. Si le point de départ a deux voisins sur le bord, alors avec probabilité 1/2 la trajectoire s'arrête et la valeur retournée est la moyenne des valeurs de b sur ces deux points. En résumé on laisse évoluer une marche aléatoire symétrique dans $h\mathbb{Z}^2$, jusqu'à ce qu'elle atteigne le bord de D. L'algorithme estime $z(x_0)$ par la moyenne empirique des valeurs prises par la condition de bord b sur les valeurs finales d'un grand nombre de trajectoires indépendantes partant de x_0.

Quel est au juste le rôle de la géométrie du réseau et de la discrétisation de pas h ? Il n'est pas très important. La marche aléatoire symétrique sur $h\mathbb{Z}^2$ découlait de la discrétisation du laplacien. Sa propriété importante est d'estimer localement le laplacien, en un sens probabiliste que nous explicitons ci-dessous.

Proposition 3.6 *Soit $\{e_1, \ldots, e_d\}$ une base orthonormée de $\mathbb{R}^d$. Soit $(X_n)_{n \in \mathbb{N}}$ la marche aléatoire symétrique sur $\mathbb{Z}^d$, de pas $X_{n+1} - X_n = G_n$ où les G_n sont indépendants et*

$$\mathbb{P}[G_n = \pm e_i] \; = \; \frac{1}{2d} \; .$$

Soit φ une fonction deux fois continûment différentiable sur $\mathbb{R}^d$. Alors :

$$\begin{array}{rcl} \lim\limits_{h \to 0^+} \frac{2d}{h^2} \mathbb{E}[\varphi(hX_{n+1}) - \varphi(x) \,|\, hX_n = x] & = & \\ \lim\limits_{h \to 0^+} \frac{1}{h^2} \sum\limits_{i=1}^{d} (\varphi(x + he_i) - \varphi(x)) + (\varphi(x - he_i) - \varphi(x)) & = & \Delta\varphi(x) \; . \end{array}$$

Cette proposition n'a bien sûr rien de profond. Elle explicite pourtant un lien important entre l'analyse numérique et les probabilités. Ce lien est le suivant. Pour calculer le laplacien d'une fonction en analyse numérique, on fait la somme de différences finies de la fonction, calculées dans les $2d$ directions de l'espace. En probabilité on tire au hasard une de ces différences finies pour se déplacer dans sa direction. En espérance, cela revient au même à une constante près. La traduction probabiliste du fait que la moyenne des différences finies, (pondérée par $1/h^2$) converge vers le laplacien, est le fait que la marche aléatoire symétrique de pas h, convenablement accélérée (en $1/h^2$) converge vers le mouvement brownien standard dans $\mathbb{R}^2$. Le rapport entre le mouvement brownien standard et le laplacien est fondamental.

Proposition 3.7 *Soit φ une fonction deux fois continûment différentiable sur $\mathbb{R}^d$. Soit $\{W(t)\,;\,t\geq 0\}$ le mouvement brownien standard dans $\mathbb{R}^d$. Alors pour tout $t\geq 0$:*

$$\lim_{\delta t\to 0^+}\frac{1}{\delta t}\Big(\,\mathbb{E}[\varphi(W(t+\delta t))\,|\,W(t)=x]-\varphi(x)\,\Big)\;=\;\frac{1}{2}\Delta\,\varphi(x)\;.$$

On comprendra donc qu'il n'est pas vraiment utile de discrétiser par un réseau carré pour résoudre le problème de Dirichlet (3.1). On obtiendra un résultat analogue en remplaçant la marche aléatoire symétrique sur $h\mathbb{Z}^2$ par une discrétisation de pas $\delta t = h^2$ du mouvement brownien. Le principe de l'algorithme reste le même : pour évaluer la solution $z(x)$ au point x du domaine, on calculera la moyenne, sur un grand nombre de trajectoires partant de x, des valeurs prises par b au point où la trajectoire atteint pour la première fois le bord.

Après avoir en quelque sorte court-circuité l'étape de discrétisation, nous pouvons nous poser la question du rapport entre la résolution du problème de Dirichlet par le mouvement brownien et le problème physique initial. En termes physiques le problème (3.1) s'énonce de la façon suivante. Supposons que D soit une plaque métallique (bonne conductrice de chaleur). Fixons la température sur le bord de D à la fonction b, et laissons s'établir l'équilibre thermique. Quand cet équilibre sera atteint, la température au point x de la plaque sera $z(x)$. Mais qu'est-ce qui transmet l'énergie des bords de la plaque vers le point x pour équilibrer sa température ? C'est l'agitation aléatoire des particules à l'intérieur de la plaque. Ces particules ne se déplacent pas selon un mouvement brownien d'un point du bord vers un point quelconque du domaine car elles ne sont mobiles que localement. Mais la transmission d'énergie entre particules se fait par des cumuls de petites interactions aléatoires locales et il n'est pas irréaliste de modéliser cette transmission par un mouvement brownien. En d'autres termes, la méthode de Monte-Carlo que nous venons de voir, au-delà de ses aspects numériques, peut être vue comme une modélisation aléatoire du phénomène physique, et constitue une alternative à la modélisation déterministe par le laplacien. Ce n'est bien sûr pas une coïncidence que le même terme "diffusion" apparaisse en physique, en analyse et en probabilités (même si les sens qui lui sont donnés dans ces trois disciplines peuvent sembler a priori n'avoir que peu de rapport entre eux). Sur cette cohérence entre modélisation déterministe et aléatoire, problèmes différentiels et simulation de processus de diffusion, on pourra se reporter à la troisième partie de Kloeden et Platen [54] ainsi qu'à Talay [91] et aux autres articles du même ouvrage.

Le rapport entre le mouvement brownien et le laplacien se généralise aux processus de diffusion. Cela fournit le principe de méthodes de Monte-Carlo pour la résolution de nombreux problèmes différentiels. Nous en donnerons plusieurs exemples dans les paragraphes suivants. Auparavant, nous présentons l'algorithme de simulation le plus simple des processus de diffusion.

3.3.2 Simulation des processus de diffusion

La référence générale sur le sujet est le livre de Kloeden et Platen [54]. Nous ne présentons ici que la méthode la plus simple, qui est la généralisation aux processus de diffusion de la méthode d'Euler pour les solutions d'équations différentielles ordinaires.

Les processus de diffusion sont solutions de problèmes de Cauchy stochastiques.

$$\left\{ \begin{array}{lcl} dX(t) & = & \mu(t, X(t))dt + \sigma(t, X(t))dW_t \\ X(0) & = & X_0 \ , \end{array} \right. \tag{3.2}$$

où

- $\mu(t,x)$ est une fonction de $\mathbb{R}^+ \times \mathbb{R}^d$ dans $\mathbb{R}^d$,
- $\sigma(t,x)$ est une fonction de $\mathbb{R}^+ \times \mathbb{R}^d$ dans $\mathcal{M}_{d\times d'}(\mathbb{R})$,
- $\{W_t \, ; \, t \geq 0\}$ désigne le mouvement brownien standard dans $\mathbb{R}^{d'}$.

Sans perte de généralité, on peut supposer $d' \leq d$. Par définition, une solution du problème (3.2) sur l'intervalle $[0,T]$ est un processus stochastique $\{X(t) \, ; \, t \in [0,T]\}$, à trajectoires continues, vérifiant pour tout $t \in [0,T]$,

$$X(t) = X_0 \ + \ \int_0^t \mu(s, X(s))ds \ + \ \int_0^t \sigma(s, X(s))dW_s \ , \tag{3.3}$$

où la seconde intégrale est une intégrale stochastique au sens de Itô (voir [13, 45, 54]). De plus, le processus $\{X(t) \, ; \, t \in [0,T]\}$ doit être adapté au sens où sa valeur à l'instant t est connue exactement si la trajectoire du mouvement brownien entre 0 et t est connue : dans l'évolution de $X(t)$, le hasard ne provient que du mouvement brownien. Dans le cas où les fonctions μ et σ ne dépendent pas de t, le processus de diffusion correspondant est dit *homogène*. Le résultat ci-dessous est le plus classique et le plus simple des résultats d'existence et d'unicité. Dans ce théorème, $\|.\|$ désigne n'importe quelle norme de vecteur ou de matrice, selon le cas.

Théorème 3.8 *Supposons les hypothèses suivantes vérifiées.*

1. *Les fonctions $\mu(t,x)$ et $\sigma(t,x)$ sont mesurables en t, pour tout $x \in \mathbb{R}^d$.*
2. *Il existe une constante positive K telle que pour tout $t \in [0,T]$ et tout $x, y \in \mathbb{R}^d$,*

$$\|\mu(t,x) - \mu(t,y)\| + \|\sigma(t,x) - \sigma(t,y)\| \ \leq \ K\|x - y\| \ .$$

   (Les fonctions μ et σ sont uniformément lipschitziennes en x.)
3. *Il existe une constante positive K' telle que pour tout $t \in [0,T]$ et tout $x \in \mathbb{R}^d$,*

$$\|\mu(t,x)\|^2 + \|\sigma(t,x)\|^2 \ \leq \ K'(1 + \|x\|^2) \ .$$

   (Les fonctions μ et σ sont à croissance au plus linéaire en x.)

4. *La variable aléatoire X_0 admet une variance et est indépendante du mouvement brownien $\{W_t\,;\,t\geq 0\}$.*

Alors il existe une solution $\{X(t)\,;\,t\in[0,T]\}$ du problème (3.2) adaptée à trajectoires continues, telle que

$$\sup\{I\!E[X^2(t)]\,;\,t\in[0,T]\}<+\infty\ .$$

De plus, il y a unicité trajectorielle : si $\{X(t)\,;\,t\in[0,T]\}$ et $\{Y(t)\,;\,t\in[0,T]\}$ sont deux solutions,

$$I\!P[\,\sup\{\|X(t)-Y(t)\|\,;\,t\in[0,T]\}=0\,]\ =\ 1\ .$$

Pour résoudre numériquement un problème de Cauchy déterministe ($\sigma(t,x)=0$), le moyen le plus simple est la méthode d'Euler. Cette méthode s'étend naturellement au cas des diffusions sous le nom de méthode d'Euler-Maruyama. Elle consiste à calculer une approximation de $X(t)$ sur une discrétisation de l'intervalle $[0,T]$, par une chaîne de Markov.

Soit $\{X(t)\,;\,t\in[0,T]\}$ le processus de diffusion solution du problème de Cauchy (3.2). Fixons un pas de temps $\delta t>0$ et notons t_n la suite des instants de discrétisation.

$$t_n\ =\ n\,\delta t\ ,\ n\geq 0\ .$$

Soit (δW_n) la suite des incréments de la discrétisation correspondante du mouvement brownien $\{W_t\,;\,t\geq 0\}$.

$$\delta W_n\ =\ W_{t_{n+1}}-W_{t_n}\ ,\ n\geq 0\ .$$

Par définition du mouvement brownien, la suite (δW_n) est une suite de vecteurs aléatoires dans $I\!R^{d'}$, indépendants et de même loi. Chacune des coordonnées de δW_n suit la loi normale $\mathcal{N}(0,\delta t)$, de moyenne 0 et de variance δt (d'écart-type $\sqrt{\delta t}$).

Définissons la chaîne de Markov $(\tilde{X}_n)$ par $\tilde{X}_0=X_0$ et pour $n\geq 0$

$$\tilde{X}_{n+1}\ =\ \tilde{X}_n+\mu(t_n,\tilde{X}_n)\,\delta t+\sigma(t_n,\tilde{X}_n)\,\delta W_n\ .$$

Dans le cas où les fonctions μ et σ ne dépendent pas de t, $(\tilde{X}_n)$ est une chaîne de Markov homogène.

Sous certaines hypothèses de régularité, les variables aléatoires $\tilde{X}_n$ approchent les variables aléatoires $X(t_n)$ de la solution exacte, au sens où l'erreur quadratique moyenne entre solution exacte et solution approchée tend vers 0 quand le pas δt tend vers 0.

Théorème 3.9 *Supposons que les hypothèses du théorème 3.8 soient vérifiées et que de plus il existe une constante $K''>0$ telle que pour tout $s,t\in[0,T]$ et tout $x\in I\!R^d$,*

$$\|\mu(t,x)-\mu(s,x)\|+\|\sigma(t,x)-\sigma(s,x)\|\ \leq\ K''|t-s|^{1/2}\ .$$

Définissons l'erreur globale $EG(\delta t)$ par

$$EG(\delta t)\ =\ \max\Big\{\,I\!E[(\tilde{X}_n-X(t_n))^2\,|\,\tilde{X}_0=X(0)=x_0]^{1/2}\ ;\ t_n\in]0,T]\,\Big\}\ .$$

Alors quand δt tend vers 0, $EG(\delta t)=O(\sqrt{\delta t})$.

L'erreur globale EG est une erreur quadratique moyenne maximale. En pratique, le point de départ x_0 est connu "exactement". C'est donc EG qui mesure la précision de l'approximation trajectorielle.

L'algorithme de simulation de la chaîne de Markov $(\tilde{X}_n)$ est très facile à implémenter.

Hors de la boucle principale :

Définir les constantes δt et $\sqrt{\delta t}$
Définir la fonction "Normale" qui retourne un vecteur
de d' variables indépendantes de loi $\mathcal{N}(0,1)$.
Initialiser X_0.

Boucle principale :

$t \longleftarrow 0$
$\tilde{X} \longleftarrow X_0$
TantQue $(t < T)$
$\tilde{X} \longleftarrow \tilde{X} + \delta t * \mu(t, \tilde{X}) + \sigma(t, \tilde{X}).(\sqrt{\delta t} * \texttt{Normale})$
$t \longleftarrow t + \delta t$
finTantQue

Dans l'algorithme ci-dessus, le produit "$\sigma(t, \tilde{X}).(\sqrt{\delta t} * \text{Normale})$" est le produit d'une matrice par un vecteur. S'il y a lieu, il faudra évidemment le programmer à part. L'évaluation des fonctions $\mu(t, \tilde{X})$ et $\sigma(t, \tilde{X})$ pourra également se faire à l'extérieur de la boucle principale. En ce qui concerne la fonction Normale, qui retourne d' variables aléatoires indépendantes de loi $\mathcal{N}(0,1)$, elle peut être programmée en utilisant l'algorithme polaire du paragraphe 2.6.2. Dans ce cas, il faudra prendre garde à utiliser successivement les deux variables aléatoires retournées par cet algorithme. Ceci ne pose pas de problème si d' est pair, mais peut conduire à doubler les instructions dans la boucle principale (simuler deux pas consécutifs) si d' est impair.

Il existe de nombreuses autres méthodes pour simuler les processus de diffusion (voir Kloeden et Platen [54]). La méthode d'Euler-Maruyama, si elle est la moins précise de toutes, présente l'avantage d'être la plus naturelle, la plus facile à programmer, et la plus rapide à l'exécution. Dans le cas déterministe, la méthode d'Euler est connue pour "cumuler les erreurs" (au sens ou l'écart entre la solution exacte et son approximation numérique augmente avec le temps). C'est aussi le cas pour la version stochastique. Le problème de la stabilité numérique est abordé dans [54] p. 331-337. Des techniques de réduction de variance appropriées aux méthodes de Monte-Carlo utilisant ce type de simulation ont été proposées. Nous ne les aborderons pas (voir [54] p. 511-527).

3.3.3 Problèmes de Dirichlet

C'est la généralisation du problème (3.1) que nous considérons ici. La présentation que nous en faisons est celle de [45] p. 364-365 (voir aussi [14] p. 237). Nous nous plaçons dans le cas de diffusions *homogènes* (les coefficients μ et σ du problème (3.2) ne dépendent pas de t).

L'application μ va de $\mathbb{R}^d$ dans $\mathbb{R}^d$, σ de $\mathbb{R}^d$ dans $\mathcal{M}_{d\times d'}(\mathbb{R})$. On note $S = (s_{ij})$ l'application qui à $x \in \mathbb{R}^d$ associe

$$S(x) \;=\; (s_{ij}(x)) \;=\; \sigma(x)\,{}^t\sigma(x)\;.$$

L'opérateur différentiel du second ordre associé à la diffusion $\{X(t)\,;\,t \geq 0\}$ est noté $\mathcal{A}$. C'est son *générateur* en tant que processus de Markov homogène : voir paragraphe suivant. A une application φ, de $\mathbb{R}^d$ dans $\mathbb{R}$, deux fois différentiable, il associe :

$$\mathcal{A}\varphi\,(x) \;=\; \frac{1}{2}\sum_{i,j=1}^{d}\; s_{ij}(x)\,\frac{\partial^2\varphi}{\partial x_i \partial x_j}(x) \;+\; \sum_{i=1}^{d}\;\mu_i(x)\frac{\partial\varphi}{\partial x_i}(x)\;.$$

L'opérateur $\mathcal{A}$ est dit elliptique au point x si la forme quadratique de matrice $S(x)$ est strictement positive. Remarquons que $\mathcal{A}$ est elliptique au point x si et seulement si la matrice $\sigma(x)$ est de rang d. On supposera donc désormais que $d' = d$.

Le problème de Dirichlet généralisé est le suivant. Soit D un domaine ouvert borné connexe de $\mathbb{R}^d$ et ∂D sa frontière, supposée suffisamment régulière (lipschitzienne par morceaux). Soient a, b, c trois applications continues

$$a : \overline{D} \longrightarrow [0, +\infty[\quad , \quad b : \partial D \longrightarrow \mathbb{R} \quad , \quad c : \overline{D} \longrightarrow \mathbb{R}\;.$$

Le problème consiste à trouver une application z continue de $\overline{D}$ dans $\mathbb{R}$, deux fois continûment différentiable sur D, telle que :

$$\left\{\begin{array}{rcll} \mathcal{A}z(x) - a(x)z(x) & = & c(x) & ,\quad \forall x \in D \\ z(x) & = & b(x) & ,\quad \forall x \in \partial D\;. \end{array}\right. \tag{3.4}$$

Le théorème suivant montre que la solution du problème (3.4) peut sous certaines hypothèses s'écrire comme l'espérance d'une fonction de la trajectoire du processus de diffusion solution du problème de Cauchy stochastique (3.2), suivie jusqu'à ce qu'elle atteigne le bord de D.

Théorème 3.10 *Supposons que l'opérateur $\mathcal{A}$ soit elliptique en tous les points de D. Supposons que les applications μ et σ vérifient les conditions du théorème 3.8.*

Notons $\{X_x(t)\,;\,t \geq 0\}$ la solution du problème de Cauchy (3.2), partant de $X_0 = x$. Le temps d'atteinte du bord de D est la variable aléatoire τ_x définie par :

$$\tau_x = \inf\{t \geq 0\,;\, X_x(t) \notin D\}\;.$$

Soit Z_x la variable aléatoire définie par :

$$Z_x \;=\; b(X_x(\tau_x))\exp\left(-\int_0^{\tau_x} a(X_x(t))\,dt\right) \;-\; \int_0^{\tau_x} c(X_x(t))\exp\left(-\int_0^{t} a(X_x(s))\,ds\right)\,dt\;.$$

Alors pour tout $x \in D$ on a :

$$\mathbb{E}[Z_x] \;=\; z(x)\;,$$

où $z(x)$ est la valeur en x de la solution du problème de Dirichlet (3.4).

D'après ce théorème, on peut donc calculer une valeur approchée de la solution du problème (3.4) au point x en simulant des trajectoires partant de x, jusqu'à ce qu'elles atteignent le bord de D, et en calculant les valeurs prises par la variable aléatoire Z_x dont l'espérance vaut $z(x)$. La moyenne empirique de ces valeurs est l'approximation cherchée. L'implémentation pose quelques problèmes pratiques, en particulier de précision numérique sur le calcul des intégrales et des exponentielles. Un autre problème est celui du temps d'atteinte du bord par la diffusion. L'hypothèse que $I\!E[\tau_x]$ est finie n'est pas particulièrement restrictive et sera vraie en dehors des cas pathologiques. Cette valeur $I\!E[\tau_x]$ contrôle le coût de l'algorithme puisque pour un pas de discrétisation δt, et un nombre de trajectoires simulées N, l'espérance du nombre total de pas simulés sera $N\,I\!E[\tau_x]/\delta t$. Un pas de discrétisation trop faible par rapport à $I\!E[\tau_x]$ conduira à un temps de calcul beaucoup trop long.

3.3.4 Equations de Fokker-Planck et Feynman-Kac

Nous avons déjà employé le terme de "générateur" à propos de l'opérateur elliptique $\mathcal{A}$ dans le paragraphe précédent. Sans rentrer dans des détails qui sortiraient du cadre de ce cours, comprendre la signification de ce terme nous permettra de comprendre pourquoi suivre au cours du temps les trajectoires d'un processus de diffusion, permet de résoudre certains problèmes différentiels. Comme dans le paragraphe précédent, les applications μ et σ ne dépendent pas de t pour l'instant. Elles sont supposées vérifier les hypothèses du théorème 3.8. Le processus de diffusion solution du problème (3.2) pour $X_0 = x$, est encore noté $\{X_x(t)\,;\, t \geq 0\}$.

Proposition 3.11 *Soit φ une application continue bornée de $I\!R^d$ dans $I\!R$. Pour tout $t > 0$, on note $\mathcal{S}_t\,\varphi$ l'application qui à $x \in I\!R^d$ associe*

$$\mathcal{S}_t\,\varphi(x) = I\!E[\varphi(X_x(t))]\ .$$

Alors

1. *L'application $\mathcal{S}_t\,\varphi$ est continue et bornée.*
2. *Quand t tend vers 0^+, $\mathcal{S}_t\,\varphi$ converge uniformément vers φ.*
3. *Pour tous $s, t \geq 0$,*
$$\mathcal{S}_{s+t}\,\varphi \;=\; \mathcal{S}_t\,\mathcal{S}_s\,\varphi \;=\; \mathcal{S}_s\,\mathcal{S}_t\,\varphi\ .$$

Les propriétés énoncées ci-dessus font de la famille d'opérateurs linéaires $\{\mathcal{S}_t\,;\, t \geq 0\}$ un *semi-groupe*. C'est ce semi-groupe qui décrit la dynamique d'évolution inhérente à l'équation

$$dX \;=\; \mu(X)dt + \sigma(X)dW\ .$$

La propriété essentielle est la troisième. Elle découle du caractère markovien et de l'homogénéité (voir par exemple [13]).

Sous les hypothèses du théorème de Hille-Yosida, tout semi-groupe admet un générateur qui, formellement, est sa dérivée logarithmique. Dans le cas du semi-groupe associé à un processus de diffusion, ce générateur est l'opérateur $\mathcal{A}$ du paragraphe

précédent. Pour toute fonction φ de $\mathbb{R}^d$ dans $\mathbb{R}$ deux fois continûment différentiable et bornée, on a

$$\frac{d}{dt}\mathcal{S}_t\,\varphi \;=\; \mathcal{A}\mathcal{S}_t\,\varphi \;=\; \mathcal{S}_t\mathcal{A}\varphi\;, \tag{3.5}$$

avec

$$\mathcal{A}\varphi\,(x) \;=\; \frac{1}{2}\sum_{i,j=1}^{d}\; s_{ij}(x)\,\frac{\partial^2\varphi}{\partial x_i\partial x_j}(x) \;+\; \sum_{i=1}^{d}\;\mu_i(x)\frac{\partial\varphi}{\partial x_i}(x)\;.$$

On peut alors traduire la relation (3.5) entre le semi-groupe et son générateur de différentes manières.

Notons $u(t,x) \;=\; \mathcal{S}_t\,\varphi(x) \;=\; \mathbb{E}[\varphi(X_x(t))]$. Alors $u(t,x)$ est solution du problème différentiel suivant.

$$\left\{\begin{array}{lcl} \dfrac{\partial u(t,x)}{\partial t} & = & \mathcal{A}u(t,x) \\ u(0,x) & = & \varphi(x)\;. \end{array}\right. \tag{3.6}$$

Mais pour tout $T > 0$ fixé et tout $t \in [0,T]$, on peut aussi remonter le temps, et considérer $v(t,x) = u(T-t,x)$. Cette fonction est solution du problème différentiel suivant.

$$\left\{\begin{array}{lcl} -\dfrac{\partial v(t,x)}{\partial t} & = & \mathcal{A}v(t,x) \\ v(T,x) & = & \varphi(x)\;. \end{array}\right. \tag{3.7}$$

L'opérateur $\mathcal{A}$ étant elliptique, sous des hypothèses de régularité suffisante des coefficients μ et σ, on démontre que la variable aléatoire $X_x(t)$ admet une densité par rapport à la mesure de Lebesgue sur $\mathbb{R}^d$, pour tout $x \in \mathbb{R}^d$ et tout $t > 0$. Supposons que ce soit le cas et notons $p(t,x,y)$ cette densité. Pour toute fonction φ continue et bornée, elle vérifie

$$\mathbb{E}[\varphi(X_x(t))] \;=\; \int_{\mathbb{R}^d}\;\varphi(y)\,p(t,x,y)\,dy\;.$$

De plus quand t tend vers 0, la mesure de densité $p(t,x,y)$ converge vers la masse de Dirac $\delta(x)$ au point x :

$$\lim_{t\to 0^+}\;p(t,x,y)\,dy \;=\; \delta(x)\;.$$

Les deux équations (3.5) entraînent les suivantes.

$$\frac{\partial p(t,x,y)}{\partial t} \;=\; \frac{1}{2}\sum_{i,j=1}^{d}\; s_{ij}(x)\,\frac{\partial^2}{\partial x_i\partial x_j}p(t,x,y) \;+\; \sum_{i=1}^{d}\;\mu_i(x)\frac{\partial}{\partial x_i}p(t,x,y)\;. \tag{3.8}$$

$$\frac{\partial p(t,x,y)}{\partial t} \;=\; \frac{1}{2}\sum_{i,j=1}^{d}\;\frac{\partial^2}{\partial y_i\partial y_j}(s_{ij}(y)\,p(t,x,y)) \;-\; \sum_{i=1}^{d}\;\frac{\partial}{\partial y_i}(\mu_i(y)\,p(t,x,y))\;. \tag{3.9}$$

La première équation porte le nom d'équation de Fokker-Planck, ou équation de Kolmogorov rétrograde (les dérivées portent sur x qui est la variable de départ). La seconde

équation porte le nom d'équation de Feynman-Kac, ou équation de Kolmogorov directe (elle porte sur la variable d'arrivée). La famille de densités $\{p(t,x,y)\}$ constitue donc un système fondamental de solutions, ou noyau fondamental. On obtient la solution générale des équations (3.8) et (3.9) en prenant le produit de convolution de $p(t,x,y)$ par une fonction quelconque.

Exemple : Noyau de la chaleur

Le mouvement brownien standard est un processus de diffusion particulier, correspondant à $\mu = 0\,,\ \sigma = I_d$. Ses composantes sont indépendantes. S'il part du point $x = (x_i)$ à l'instant s, sa densité à l'instant $s+t$ sera le produit des densités des composantes, à savoir le produit des densités de lois normales de moyenne x_i et de variance t.

$$p(t,x,y) \;=\; \prod_{i=1}^{d} \frac{1}{\sqrt{2\pi t}} e^{-\frac{(y_i-x_i)^2}{2t}} \;=\; \frac{1}{(2\pi t)^{d/2}} e^{-\frac{\|y-x\|^2}{2t}}\ ,$$

où $\|.\|$ désigne la norme euclidienne de $\mathbb{R}^d$. C'est le noyau de la chaleur, solution des deux "équations de la chaleur" suivantes, à comparer avec la proposition 3.7.

$$\frac{\partial p(t,x,y)}{\partial t} \;=\; \frac{1}{2}\,\Delta_x\, p(t,x,y)\ ,$$

et

$$\frac{\partial p(t,x,y)}{\partial t} \;=\; \frac{1}{2}\,\Delta_y\, p(t,x,y)\ .$$

Ce qui vient d'être dit pour le cas des diffusions homogènes reste valable dans le cas non homogène, même si on perd alors l'interprétation de l'opérateur différentiel $\mathcal{A}$ comme générateur d'un semi-groupe. Ce qui suit est tiré de Karatzas et Shreve [45] p. 366-369.

Soient μ et σ deux fonctions de $\mathbb{R}^+ \times \mathbb{R}^d$ dans $\mathbb{R}^d$ et $\mathcal{M}_{d\times d}(\mathbb{R})$ respectivement. On note $S = (s_{ij})$ l'application qui à $(t,x) \in \mathbb{R}^+ \times \mathbb{R}^d$ associe

$$S(t,x) \;=\; (s_{ij}(t,x)) \;=\; \sigma(t,x)\,{}^t\sigma(t,x)\ .$$

L'opérateur différentiel $\mathcal{A}_t$ est par définition celui qui associe à une application φ, de $\mathbb{R}^d$ dans $\mathbb{R}$, deux fois différentiable,

$$\mathcal{A}_t\varphi\,(x) \;=\; \frac{1}{2}\sum_{i,j=1}^{d} s_{ij}(t,x)\,\frac{\partial^2\varphi}{\partial x_i \partial x_j}(x) \;+\; \sum_{i=1}^{d} \mu_i(t,x)\frac{\partial\varphi}{\partial x_i}(x)\ .$$

Les conditions sous lesquelles les affirmations qui vont suivre sont vraies portent sur

- la régularité des fonctions μ et σ, ainsi que des fonctions b et c du problème (3.12) ci-dessous,
- la croissance linéaire ou polynômiale de ces mêmes fonctions,

– l'ellipticité uniforme des opérateurs $\mathcal{A}_t$.

(Se reporter à [45] pour les énoncés précis). On note $\{X(t)\,;\,t \geq 0\}$ le processus de diffusion, solution du problème de Cauchy (3.2). Les opérateurs $\mathcal{A}_t$ étant elliptiques, si μ et σ sont suffisamment régulières, alors le processus $\{X(t)\,;\,t \geq 0\}$ admet un noyau de transition, noté $p(s,x,t,y)$. Pour toute fonction φ continue et bornée, ce noyau vérifie, pour tout couple d'instants $0 \leq s < t$,

$$\mathbb{E}[\varphi(X(t))|X(s)=x] \;=\; \int_{\mathbb{R}^d} \varphi(y)\, p(s,x,t,y)\, dy\ .$$

De plus, on a :

$$\lim_{t-s\to 0^+} p(s,x,t,y)dy = \delta(x)\ ,$$

Ce noyau de transition est solution des équations de Fokker-Planck (3.10) et de Feynman-Kac (3.11) ci-dessous.

$$\frac{\partial p(s,x,t,y)}{\partial s} \;=\; \frac{1}{2}\sum_{i,j=1}^{d} s_{ij}(s,x)\,\frac{\partial^2}{\partial x_i \partial x_j} p(s,x,t,y) \;+\; \sum_{i=1}^{d} \mu_i(s,x)\frac{\partial}{\partial x_i} p(s,x,t,y)\ . \tag{3.10}$$

$$\frac{\partial p(s,x,t,y)}{\partial t} \;=\; \frac{1}{2}\sum_{i,j=1}^{d} \frac{\partial^2}{\partial y_i \partial y_j}(s_{ij}(t,y)\, p(s,x,t,y)) \;-\; \sum_{i=1}^{d} \frac{\partial}{\partial y_i}(\mu_i(t,y)\, p(s,x,t,y))\ . \tag{3.11}$$

Considérons le problème différentiel suivant.

$$\left\{\begin{array}{lcl} -\dfrac{\partial u(t,x)}{\partial t} & = & \mathcal{A}_t u(t,x) + c(t,x) \\ u(T,x) & = & b(x)\ . \end{array}\right. \tag{3.12}$$

La notation $\mathcal{A}_t u(t,x)$ désigne l'image par l'opérateur $\mathcal{A}_t$ de l'application qui à $x \in \mathbb{R}^d$ associe $u(t,x)$. La solution de (3.12) (sur $[0,T]\times\mathbb{R}^d$) s'écrit :

$$u(t,x) \;=\; \int_{\mathbb{R}^d} p(t,x,T,y) b(y)\, dy \;+\; \int_t^T \int_{\mathbb{R}^d} p(t,x,\tau,y)\, c(\tau,y)\, dy d\tau\ .$$

Considérons le problème plus général suivant.

$$\left\{\begin{array}{lcl} -\dfrac{\partial u(t,x)}{\partial t} & = & \mathcal{A}_t u(t,x) + a(t,x)u(t,x) + c(t,x) \\ u(T,x) & = & b(x)\ . \end{array}\right. \tag{3.13}$$

La solution de ce problème s'écrit comme l'espérance d'une variable aléatoire, fonction des trajectoires du processus de diffusion issu de x à l'instant t $\{X_x(s)\,;\,s\in[t,T]\}$, de la façon suivante.

$$\begin{array}{rcl} u(t,x) & = & \mathbb{E}\left[b(X_x(T)) \exp\left(\displaystyle\int_t^T a(s,X_x(s))\, ds\right)\right. \\ & + & \left.\displaystyle\int_t^T c(\tau, X_x(\tau)\, \exp\left(\int_t^\tau a(s,X_x(s))\, ds\right)\, d\tau\right]\ . \end{array}$$

Tous les résultats théoriques de ce paragraphe se traduisent de manière algorithmique en des méthodes de résolution approchée de problèmes différentiels paraboliques. Pour ce qui est des solutions fondamentales sous forme de noyau, elles peuvent être approchées par des histogrammes : l'histogramme à l'instant $t > s$ des valeurs prises par N trajectoires de la diffusion, partant toutes de x à l'instant s, est une approximation de la fonction (de y) $p(s, x, t, y)$.

3.4 Méthodes particulaires

L'intérêt des problèmes différentiels des paragraphes 3.3.3 et 3.3.4 est plus de faire ressortir une cohérence dans la modélisation que de proposer des méthodes numériques efficaces. Au moins en basse dimension, pour la résolution des problèmes linéaires (de type Dirichlet ou Fokker-Planck), les méthodes de Monte-Carlo sont beaucoup moins efficaces que les méthodes d'éléments finis classiques. Nous allons montrer dans cette dernière partie que les processus de diffusion peuvent servir à résoudre certains problèmes d'EDP non linéaires. Notre référence générale sera ici l'ouvrage collectif [40], et en particulier les contributions de Méléard p. 42–95 et Pulvirenti p. 96–126 (voir aussi [59]). Nous nous contenterons de dégager quelques idées générales sur la propagation du chaos et la résolution de certaines équations aux dérivées partielles non linéaires par des méthodes stochastiques. Nous renvoyons à [40] et aux nombreuses références de cet ouvrage pour le traitement mathématique, qui est d'un niveau souvent très supérieur à celui de ces notes. Notre objectif est de faire apparaître une fois de plus la cohérence entre le point de vue déterministe et le point de vue stochastique, tout en restant le plus proche possible des considérations algorithmiques.

3.4.1 Propagation du chaos

Dans les méthodes des paragraphes précédents, la solution était écrite comme l'espérance d'une certaine fonction d'un processus de diffusion. Il suffisait de calculer cette espérance de manière approchée en appliquant la loi des grands nombres à N trajectoires indépendantes. L'idée ici sera encore d'approcher une solution par des moyennes empiriques en utilisant la loi des grands nombres. Mais on l'appliquera à une famille de processus qui ne sont indépendants qu'asymptotiquement. Par analogie avec les modèles physiques à propos desquels ces méthodes sont apparues, les processus qui interagissent sont appelés particules, d'où le nom de méthodes particulaires.

Il s'agit en général d'un ensemble de N processus de diffusion, $(\{X_i^N(t)\,;\, t \geq 0\})_{i=1,\ldots,N}$ dont chacun interagit de manière symétrique avec tous les autres, les interactions individuelles restant faibles. Nous précisons ce cadre à l'aide de l'exemple introductif de Pulvirenti, [40] p. 97.

Exemple :

Soit K une application de $\mathbb{R}^d$ dans $\mathbb{R}^d$, de classe $\mathcal{C}^\infty$. Pour $N \in \mathbb{N}^*$, considérons le

système d'équations différentielles ordinaires suivant.

$$\frac{dX_i^N(t)}{dt} \;=\; \frac{1}{N}\sum_{j=1}^{N} K(X_i^N(t) - X_j^N(t)) \;, \quad i = 1, \ldots, N \;. \tag{3.1}$$

Supposons que la condition initiale de ce système soit formée de vecteurs aléatoires indépendants $(X_i^N(0))_{i\leq N}$. Les vecteurs solution $(X_i^N(t))_{i\leq N}$ ne sont pas indépendants. Il est cependant possible de montrer que pour tout $k \in I\!N$ et pour tout $t > 0$, le vecteur $(X_i^N(t))_{i\leq k}$ converge en loi, lorsque N tend vers l'infini, vers une mesure produit. En d'autres termes, les $X_i^N(t)$ sont asymptotiquement indépendants pour tout t. Considérons la mesure empirique $\pi_N(t)$ associée au vecteur $(X_i^N(t))_{i\leq N}$:

$$\pi_N(t) \;=\; \frac{1}{N}\sum_{i=1}^{N} \delta(X_i^N(t)) \;,$$

où $\delta(x)$ désigne la masse de Dirac au point $x \in I\!R^d$. Soit φ une fonction test (de classe $\mathcal{C}^\infty$, à support compact sur $I\!R^d$). On a :

$$\begin{aligned}
\frac{d}{dt}\langle\, \pi_N(t)\,,\, \varphi\,\rangle \;&=\; \frac{d}{dt}\,\frac{1}{N}\sum_{i=1}^{N} \varphi(X_i^N(t)) \\
&=\; \frac{1}{N}\sum_{i=1}^{N} \Big(\frac{d}{dt}X_i^N(t)\Big).\nabla\varphi(X_i^N(t)) \\
&=\; \frac{1}{N}\sum_{i=1}^{N} \Big(\frac{1}{N}\sum_{j=1}^{N} K(X_i^N(t) - X_j^N(t))\Big).\nabla\varphi(X_i^N(t)) \\
&=\; \langle\, \pi_N(t)\,,\, K * \pi_N(t).\nabla\varphi\,\rangle \\
&=\; -\langle\, div[(K * \pi_N(t))\pi_N(t)]\,,\, \varphi\,\rangle \;,
\end{aligned}$$

où $K * \pi_N(t)(x) = \int K(x-y)\pi_N(t)(dy)$. En d'autres termes, $\pi_N(t)$ est une solution faible de l'équation non linéaire suivante.

$$\frac{\partial\pi}{\partial t} + div[(K * \pi)\pi] \;=\; 0 \;. \tag{3.2}$$

Supposons qu'au temps $t = 0$, la mesure $\pi_N(0)$ converge vers la mesure $f(0,x)dx$. C'est le cas si les conditions initiales sont des variables aléatoires indépendantes de densité $f(0,x)$, par la loi des grands nombres. On peut alors démontrer, les variables $X_i^N(t)$ étant asymptotiquement indépendantes, que leur mesure empirique $\pi_N(t)$ converge pour tout t vers la mesure $f(t,x)dx$, où $f(t,x)$ est solution forte de (3.2), pour la condition initiale $f(0,x)$. On a donc mis en correspondance l'équation non linéaire (3.2), avec le système (3.1). En pratique, on pourra approcher la solution (forte) de (3.2) par un histogramme des coordonnées de la solution de (3.1), pour N assez grand.

Dans cet exemple, deux propriétés ont joué un rôle essentiel.

1. Si les $X_i^N(0)$ sont indépendantes, alors les $X_i^N(t)$ sont asymptotiquement indépendantes quand N tend vers l'infini. Cette propriété porte le nom de *propagation du chaos*.
2. La mesure empirique $\pi_N(t)$ converge vers une mesure déterministe, par la loi des grands nombres.

De ces deux propriétés, on conçoit que la première entraîne la seconde. Elles sont en fait équivalentes pour des lois échangeables, au sens du théorème suivant (voir par exemple Méléard [40] p. 66).

Théorème 3.12 *Pour tout $N \in \mathbb{N}^*$, soit $(X_i^N)_{i\leq N}$ un vecteur de variables aléatoires à valeurs dans $\mathbb{R}^d$, de loi échangeable, c'est-à-dire que (X_i^N) a même loi que $(X_{\rho(i)}^N)$ pour toute permutation ρ des coordonnées. Soit π une loi de probabilité sur $\mathbb{R}^d$. Les deux propriétés suivantes sont équivalentes.*

1. *Pour tout $k \geq 0$, le vecteur $(X_i^N)_{i\leq k}$ converge en loi vers la mesure produit $\pi^{\otimes k}$.*
2. *La mesure empirique*
$$\pi_N \;=\; \frac{1}{N}\sum_{i=1}^{N} \delta(X_i^N)$$
converge vers la mesure π.

Les paragraphes qui suivent sont consacrés à d'autres exemples d'applications de la même démarche.

3.4.2 Equations de McKean-Vlasov

L'équation de Vlasov modélise le comportement d'un gaz à forte densité de particules, pour lesquelles les collisions sont si fréquentes que leur effet peut être décrit par une diffusion dont le terme de dérive interactif traduit l'attraction ou la répulsion des particules. C'est McKean qui le premier a donné la traduction stochastique de ce modèle. L'équation de McKean-Vlasov dans $\mathbb{R}^d$ s'écrit sous la forme générale suivante, que l'on peut voir comme une généralisation de l'équation de Fokker-Planck (3.8).

$$\frac{\partial \pi(t)}{\partial t} = \frac{1}{2}\sum_{i,j=1}^{d} \frac{\partial^2}{\partial x_i \partial x_j}(s_{ij}[x,\pi(t)])\pi(t) - \sum_{i=1}^{d}\frac{\partial}{\partial x_i}(\mu_i[x,\pi(t)])\pi(t)\;, \tag{3.3}$$

où $\pi(t)$ est une mesure de probabilité sur $\mathbb{R}^d$ et

$$\begin{aligned}
\mu[x,\pi] &= \int_{\mathbb{R}^d} \mu(x,y)\pi(dy)\;, \quad \mu(x,y) \in \mathbb{R}^d\;,\\
S[x,\pi] &= \sigma[x,\pi]\,{}^t\sigma[x,\pi]\;,\\
\sigma[x,\pi] &= \int_{\mathbb{R}^d} \sigma(x,y)\pi(dy)\;, \quad \sigma(x,y) \in \mathcal{M}_{d\times d'}(\mathbb{R})\;,
\end{aligned}$$

pour toute mesure de probabilité π sur $\mathbb{R}^d$. L'équation (3.3) doit être comprise au sens faible : pour toute fonction test φ, on doit avoir

$$\frac{\partial}{\partial t}\langle \pi(t), \varphi \rangle = \langle\, \pi(t)\,,\, \frac{1}{2}\sum_{i,j=1}^{d} s_{ij}[x, \pi(t)]\frac{\partial^2 \varphi}{\partial x_i \partial x_j} + \sum_{i=1}^{d} \mu_i[x, \pi(t)]\frac{\partial \varphi}{\partial x_i}\,\rangle\ . \tag{3.4}$$

L'idée, comme en 3.3.4, est d'associer à (3.3) un générateur, et une équation différentielle stochastique. Le générateur est l'opérateur différentiel $\mathcal{A}(\pi)$, qui à une fonction test φ associe :

$$\mathcal{A}(\pi)\varphi(x) = \frac{1}{2}\sum_{i,j=1}^{d} s_{ij}[x, \pi]\frac{\partial^2 \varphi}{\partial x_i \partial x_j} + \sum_{i=1}^{d} \mu_i[x, \pi]\frac{\partial \varphi}{\partial x_i}\ . \tag{3.5}$$

L'équation différentielle stochastique correspondante est :

$$dX = \mu[X(t), \pi(t)]dt + \sigma[X(t), \pi(t)]dW\ , \tag{3.6}$$

où $\{W(t)\,;\, t \geq 0\}$ est le mouvement brownien standard dans $\mathbb{R}^{d'}$. Dans (3.6), $\pi(t)$ désigne la loi à l'instant t de $X(t)$, d'où la non linéarité. On peut comprendre physiquement cette équation en disant que la trajectoire d'une particule entre les instants t et $t + dt$ est celle d'une diffusion dont les coefficients dépendent non seulement de la position de la particule à l'instant t, mais aussi de la densité globale $\pi(t)$ des particules. Nous ne donnerons pas les conditions d'existence et d'unicité de la solution de (3.6) (voir [40] p. 46).

L'algorithme de simulation approchée de cette solution est tout à fait naturel. Il suffit en effet de simuler une famille de processus de diffusion, par exemple par la méthode d'Euler-Maruyama vue en 3.3.2, en remplaçant dans les coefficients de dérive μ et de diffusion σ, la loi $\pi(t)$ par son approximation à l'aide de la mesure empirique :

$$\pi_N(t) \ = \ \frac{1}{N}\sum_{i=1}^{N} \delta(X_i(t))\ .$$

Pour $N \in \mathbb{N}^*$, on est alors conduit à définir l'ensemble de diffusions $(X_i^N)_{i \leq N}$ comme solution du système d'équations différentielles stochastiques "ordinaires" :

$$dX_i^N(t) = \mu[X_i^N(t), \pi_N(t)]dt + \sigma[X_i^N(t), \pi_N(t)]dW_i(t)\ , \tag{3.7}$$

où pour $i \leq N$ les $\{W_i(t)\,;\, t \geq 0\}$ sont des mouvements browniens indépendants.

On montre alors que les "particules" solutions de (3.7) vérifient la propriété de propagation du chaos. Si à l'instant 0 elles sont indépendantes, alors quand N tend vers l'infini elles sont asymptotiquement indépendantes pour tout t et leur mesure empirique converge vers la solution de (3.3).

3.4.3 Equations de Boltzmann

Les équations de Boltzmann modélisent le comportement d'un gaz dont les molécules, en plus de leur mouvement diffusif, subissent des chocs qui modifient instantanément leur vitesse. Tenir compte de ces chocs impose des discontinuités qui rajoutent une difficulté supplémentaire par rapport au cas précédent. Considérons en effet la variable d'état position–vitesse $(z, v) \in I\!R^3 \times I\!R^3$, pour une molécule de gaz. Un choc est une transition instantanée de l'état (z, v) vers l'état (z, v'). Un processus modélisant ces chocs est donc un cas particulier de processus de saut pur dans $I\!R^d$. De tels processus peuvent être définis par leur générateur, avec le même sens mathématique donné à celui-ci qu'en 3.3.4. Si φ désigne une fonction test sur $I\!R^d$, le générateur $\mathcal{B}(\pi)$ du processus de saut est défini, pour toute mesure de probabilité π sur $I\!R^d$ par :

$$\mathcal{B}(\pi)\varphi(x) \;=\; \int (\varphi(x+h) - \varphi(x))\, \Gamma(x, \pi, dh)\ , \tag{3.8}$$

où $\Gamma(x, \pi, dh)$ désigne une famille de mesures positives bornées sur $I\!R^d$. Nous noterons $|\Gamma(x, \pi)|$ la masse totale de la mesure $\Gamma(x, \pi, dh)$.

$$|\Gamma(x, \pi)| \;=\; \int_{h \in I\!R^d} \Gamma(x, \pi, dh)\ .$$

Il faut comprendre la définition du générateur $\mathcal{B}(\pi)$ comme suit. Si x est l'état courant de la particule, alors x sautera avec taux $|\Gamma(x, \pi)|$, c'est-à-dire au bout d'un temps suivant la loi exponentielle de moyenne $1/|\Gamma(x, \pi)|$. Au bout de ce temps, x sautera vers $x+h$, où h sera choisi selon la loi de probabilité $\Gamma(x, \pi, dh)/|\Gamma(x, \pi)|$. La mesure de probabilité π dont dépend le générateur $\mathcal{B}(\pi)$ est pour l'instant un paramètre. Comme dans (3.5), π sera interprétée comme la loi de probabilité du processus courant.

Le modèle le plus général sera obtenu en superposant un processus de diffusion et un processus de saut indépendants. Cela se fait en ajoutant au générateur $\mathcal{A}(\pi)$ de (3.5), le générateur $\mathcal{B}(\pi)$ qui vient d'être défini. Cette extension conduit fort naturellement à résoudre l'équation suivante, qui généralise (3.3), et que nous interprèterons au sens faible, comme (3.4). La solution est une mesure de probabilité $\pi(t)$ sur $I\!R^d$, telle que pour toute fonction test φ, on ait :

$$\frac{\partial}{\partial t}\langle \pi(t), \varphi \rangle = \langle\, \pi(t)\,,\, \mathcal{A}(\pi(t))\varphi + \mathcal{B}(\pi(t))\varphi\, \rangle\ . \tag{3.9}$$

L'idée de la méthode particulaire consiste à simuler un ensemble de N processus de diffusion avec sauts, $(\{X_i(t)\})_{i \le N}$, qui interagissent par l'intermédiaire de leur mesure empirique, comme le système défini par (3.7). Ce système vérifie la propriété de propagation du chaos, et pour des conditions initiales indépendantes, sa mesure empirique $\pi_N(t)$ converge vers la solution de (3.9). Plutôt que de détailler le résultat théorique de convergence, pour lequel nous renvoyons à Méléard [40], p. 54–63, nous donnerons un algorithme de simulation. L'algorithme que nous proposons n'est évidemment pas le seul possible (voir Méléard [40] p. 63). Il consiste à simuler le vecteur des N diffusions, en lui superposant un processus de sauts, simulé par une technique de rejet classique.

Pour appliquer cette technique au processus de saut de générateur $\mathcal{B}(\pi)$, nous devons supposer que le taux de saut est uniformément borné.

$$\sup_{x,\pi} |\Gamma(x,\pi)| = C < +\infty \ .$$

La constante C est en quelque sorte l'échelle de temps selon laquelle se produisent les sauts. Supposons tout d'abord que la partie diffusion n'existe pas. On a alors à simuler N processus de saut pur, interagissant seulement par l'intermédiaire de leur mesure empirique. L'algorithme de simulation du vecteur $(\{X_i(t)\})_{i\leq N}$ de ces N processus est le suivant.

$t \longleftarrow 0$
Initialiser $(X_1, \ldots, X_N)$
Répéter
    choisir i avec probabilité $1/N$
    calculer $|\Gamma(X_i, \pi_N)|$
    Si (Random $< |\Gamma(X_i, \pi_N)|/C$) alors
        choisir h selon la loi $\Gamma(X_i, \pi_N, dh)/|\Gamma(X_i, \pi_N)|$
        $X_i \longleftarrow X_i + h$
    FinSi
    $t \longleftarrow t + 1/(NC)$
Jusqu'à (arrêt de la simulation)

Nous devons maintenant superposer cet algorithme de saut avec un processus de diffusion. La seule difficulté consiste à harmoniser les deux échelles de temps,

1. celle de la chaîne associée aux N processus de diffusion par la méthode d'Euler-Maruyama, qui dépend du pas de discrétisation δt (voir 3.3.2),
2. celle des processus de saut, pour laquelle l'intervalle de temps entre deux tentatives de saut consécutives vaut en moyenne $1/(NC)$.

L'algorithme que nous proposons consiste à accélérer l'échelle de temps du processus de saut de $1/(NC)$ jusqu'à δt, quitte à augmenter le nombre de sauts fictifs. Ceci suppose évidemment que δt soit inférieur à $1/(NC)$. Il semble raisonnable, pour éviter trop d'appels inutiles de Random, de faire en sorte que $1/(NC)$ et δt soient relativement proches, ce qui impose que $N\,\delta t$ soit de l'ordre de $1/C$. Il n'est pas évident que C soit effectivement calculable, auquel cas on le remplacera par un majorant, au prix d'une nouvelle perte de temps en sauts fictifs.

L'algorithme de simulation est le suivant.

$t \longleftarrow 0$
Initialiser $(X_1, \ldots, X_N)$
Répéter
    Pour i de 1 à N *(* diffusion *)*
        $X_i \longleftarrow X_i + \delta t * \mu(X_i, \pi_N) + \sigma(X_i, \pi_N).(\sqrt{\delta t} * \texttt{Normale})$
    FinPour
    choisir i avec probabilité $1/N$ *(* sauts *)*

calculer $|\Gamma(X_i, \pi_N)|$
Si (Random $< \dfrac{|\Gamma(X_i, \pi_N)|}{C} \dfrac{\delta t}{1/(NC)}$) alors
choisir h selon la loi $\Gamma(X_i, \pi_N, dh)/|\Gamma(X_i, \pi_N)|$
$X_i \longleftarrow X_i + h$
FinSi
$t \longleftarrow t + \delta t$ *(* échelle de temps *)*
Jusqu'à (arrêt de la simulation)

3.5 Exercices

Exercice 3.1 Des objets, nommés $x, y_1, y_2, \ldots, y_{N-1}$, sont rangés dans un tableau de taille N dans lequel on accède de manière séquentielle. A chaque accès au tableau, on recherche l'un des N objets, soit x avec probabilité a, soit l'un des $N-1$ autres, avec probabilité b pour chacun d'eux ($a + (N-1)b = 1$). Le choix à chaque accès est indépendant des recherches précédentes.

Les probabilités d'accès a et b sont a priori inconnues, mais on soupçonne que l'objet x est plus fréquemment appelé que les autres. Dans toute la suite on supposera donc $a > b$. A chaque accès, on décide de déplacer l'objet choisi, de manière à ce qu'il soit placé plus près de la tête du tableau s'il est fréquemment appelé. Deux stratégies sont envisagées.

1. *Move ahead :* Si l'objet choisi est le premier, il n'est pas déplacé. Sinon, il est échangé avec l'objet qui le précédait. On note $X_n \in \{1, \ldots, N\}$ le rang de l'objet x dans le tableau à l'issue du n-ième accès.
   (a) Montrer que $\{X_n\ ,\ n \in I\!N\}$ est une chaîne de Markov homogène.
   (b) Ecrire le diagramme de transition et la matrice de transition de la chaîne $\{X_n\ ,\ n \in I\!N\}$.
   (c) Soit $p = (p_i)\ ,\ i = 1, \ldots, N$ la mesure stationnaire de la chaîne $\{X_n\ ,\ n \in I\!N\}$. Montrer que pour tout $i = 2, \ldots, N$,
   $$\frac{p_i}{p_{i-1}} = \frac{b}{a}\ .$$
   (d) En déduire que la suite des p_i est décroissante (on dit que la stratégie est *auto-arrangeante*).
2. *Move to front :* Si l'objet choisi est le premier, il n'est pas déplacé. Sinon, il est placé en tête, et les objets qui le précédaient sont décalés vers la droite. On note $Y_n \in \{1, \ldots, N\}$ le rang de l'objet x dans le tableau à l'issue du n-ième accès.
   (a) Montrer que $\{Y_n\ ,\ n \in I\!N\}$ est une chaîne de Markov homogène.
   (b) Ecrire le diagramme de transition et la matrice de transition de la chaîne $\{Y_n\ ,\ n \in I\!N\}$.

(c) Soit $q = (q_i)\ ,\ i = 1, \ldots, N$ la mesure stationnaire de la chaîne $\{Y_n\ ,\ n \in I\!N\}$. Montrer que pour tout $i = 2, \ldots, N$,

$$\frac{q_i}{q_{i-1}} = \frac{(N-i+1)b}{a+(N-i)b}\ .$$

(d) En déduire que la suite des q_i est décroissante.

3. *Comparaison :*

(a) Montrer que pour tout $i = 2, \ldots, N$,

$$\frac{p_i}{p_{i-1}} < \frac{q_i}{q_{i-1}}\ .$$

(b) En déduire que $p_1 > q_1$.

(c) Laquelle des deux stratégies choisiriez-vous ?

Exercice 3.2 Soit d un entier pair et $\alpha \in]0, 1/d[$. Soit A une matrice de dimension d telle que sur chaque ligne, la moitié des coefficients valent $+\alpha$, l'autre moitié $-\alpha$. Soit b le vecteur dont toutes les coordonnées valent 1.

1. Ecrire la solution du système :

$$(I - A)z \ = \ b\ .$$

2. Soit $z^{(k)}$ la solution approchée à l'ordre k.

$$z^{(k)} \ = \ (I + A + \cdots + A^k)b\ .$$

Comment choisir k pour être sûr que $|z - z^{(k)}|$ soit inférieur à 10^{-3} ?

3. Ecrire l'algorithme de calcul de la i-ième coordonnée de $z^{(k)}$, en appliquant la première des deux méthodes précédentes (paragraphe 3.2.1), pour

$$p_{ij} = \frac{1}{d}\ , \quad \forall i, j = 1, \ldots, d\ .$$

4. Calculer la variance de Z_k.
5. En déduire le nombre de trajectoires nécessaires pour que l'amplitude de l'intervalle de confiance soit inférieure à 10^{-3}.
6. Soit $\beta \in]0, 1/d[$. Ecrire l'algorithme de calcul de la i-ième coordonnée de z par la seconde des deux méthodes précédentes (paragraphe 3.2.2), pour

$$p_{ij} = \beta\ , \quad \forall i, j = 1, \ldots, d\ .$$

7. Quelle est la loi du nombre moyen de pas de la chaîne avant absorption ?
8. Calculer la variance de Z'.
9. En déduire le nombre de trajectoires nécessaires pour que l'amplitude de l'intervalle de confiance soit inférieure à 10^{-3}.

10. Discuter de l'intérêt éventuel de choisir $\beta \neq \alpha$.
11. Comparer les deux méthodes.

Exercice 3.3 1. Ecrire un algorithme de simulation approchée pour le processus de diffusion dans $\mathbb{R}$, solution de l'équation différentielle stochastique :

$$\left\{ \begin{array}{rcl} dX(t) & = & \sin(X(t))dt + \cos(X(t))dW_t \\ X(0) & = & X_0 \ , \end{array} \right. \tag{3.10}$$

où $\{W_t \, ; \, t \geq 0\}$ désigne le mouvement brownien standard dans $\mathbb{R}$ et X_0 une variable aléatoire de carré intégrable indépendante de $\{W_t \, ; \, t \geq 0\}$.

2. En déduire un algorithme de calcul approché de la fonction z, de $[-1,1]$ dans $\mathbb{R}$, solution du problème différentiel suivant.

$$\left\{ \begin{array}{rcl} \frac{1}{2}\cos^2(x)\frac{d^2z}{dx^2}(x) + \sin(x)\frac{dz}{dx}(x) - z(x) & = & 0 \quad , \quad \forall x \in]-1,1[\\ z(-1) \ = \ z(1) & = & 1 \quad . \end{array} \right. \tag{3.11}$$

Exercice 3.4 Soit $\mathcal{A}$ l'opérateur différentiel qui à une application φ, de $\mathbb{R}^2$ dans $\mathbb{R}$, deux fois différentiable, associe :

$$\mathcal{A}\varphi\,(x_1,x_2) \ =$$

$$(1+\cos^2 x_1)\frac{\partial^2\varphi}{\partial x_1^2}(x_1,x_2) \ + \ 2\sin(x_1+x_2)\frac{\partial^2\varphi}{\partial x_1\partial x_2}(x_1,x_2) \ + \ (1+\cos^2 x_2)\frac{\partial^2\varphi}{\partial x_2^2}(x_1,x_2)$$

$$+ \ x_1\frac{\partial\varphi}{\partial x_1}(x_1,x_2) \ + \ x_2\frac{\partial\varphi}{\partial x_2}(x_1,x_2) \ .$$

1. Montrer que l'opérateur $\mathcal{A}$ est elliptique en tout point (x_1,x_2) de $\mathbb{R}^2$.
2. Soit D le disque unité ouvert de $\mathbb{R}^2$. On considère le problème de Dirichlet suivant.

$$\left\{ \begin{array}{rcll} \mathcal{A}z(x_1,x_2) - (x_1+x_2)^2 z(x_1,x_2) & = & e^{-(x_1+x_2)^2} & , \quad \forall (x_1,x_2) \in D \ , \\ z(x_1,x_2) & = & 1 & , \quad \forall (x_1,x_2) \in \partial D \ . \end{array} \right. \tag{3.12}$$

Ecrire un algorithme de Monte-Carlo pour le calcul approché de la solution en un point (x_1,x_2) quelconque de D.

3. On considère l'équation parabolique :

$$\frac{\partial u}{\partial t}(t,x_1,x_2) \ = \ \mathcal{A}u(t,x_1,x_2) \ .$$

Ecrire un algorithme de calcul approché du noyau fondamental $p(t,x_1,x_2,y_1,y_2)$ de cette équation.

4. En déduire un algorithme de résolution approchée du problème de Cauchy suivant :

$$\left\{ \begin{array}{rcll} \dfrac{\partial u}{\partial t}(t,x_1,x_2) & = & \mathcal{A}u(t,x_1,x_2) & , \quad \forall (t,x_1,x_2) \in [0,T] \times I\!R \times I\!R \, , \\ u(0,x_1,x_2) & = & e^{-(x_1+x_2)^2} & , \quad \forall (x_1,x_2) \in I\!R \times I\!R \, . \end{array} \right. \tag{3.13}$$

5. On note $\mathcal{A}_t$ l'opérateur différentiel défini par :

$$\mathcal{A}_t \varphi \; = \; e^t \mathcal{A} \varphi \, .$$

6. Modifier les algorithmes des questions précédentes pour la résolution du problème suivant.

$$\left\{ \begin{array}{rcll} \dfrac{\partial u}{\partial t}(t,x_1,x_2) & = & \mathcal{A}_t u(t,x_1,x_2) & , \quad \forall (t,x_1,x_2) \in [0,T] \times I\!R \times I\!R \, , \\ u(0,x_1,x_2) & = & \cos(x_1 x_2) & , \quad \forall (x_1,x_2) \in I\!R \times I\!R \, . \end{array} \right. \tag{3.14}$$

7. Ecrire un algorithme de résolution approchée du problème suivant.

$$\left\{ \begin{array}{rcl} \dfrac{\partial u}{\partial t}(t,x_1,x_2) & = & \mathcal{A}_t u(t,x_1,x_2) + \cos(t x_1 x_2)\, u(t,x_1,x_2) + \sin(t x_1 x_2) \, , \\ u(T,x_1,x_2) & = & 1 \, . \end{array} \right. \tag{3.15}$$

Exercice 3.5 Soit $\mathcal{A}$ l'opérateur différentiel qui à une application φ, de $I\!R^2$ dans $I\!R$, deux fois différentiable, associe :

$$\mathcal{A}\varphi\,(x_1,x_2) \; = \; \frac{1}{2} e^{-x_1^2} \frac{\partial^2 \varphi}{\partial x_1^2}(x_1,x_2) + \frac{1}{\sqrt{2}} e^{-(x_1^2+x_2^2)/2} \frac{\partial^2 \varphi}{\partial x_1 \partial x_2}(x_1,x_2) + \frac{1}{2} e^{-x_2^2} \frac{\partial^2 \varphi}{\partial x_2^2}(x_1,x_2) \, .$$

1. Ecrire un algorithme de simulation pour un processus de diffusion homogène $\{(X_1(t), X_2(t))\, ,\; t \geq 0\}$, admettant l'opérateur $\mathcal{A}$ pour générateur.
2. Soit D le disque unité ouvert de $I\!R^2$. On considère le problème de Dirichlet suivant.

$$\left\{ \begin{array}{rcll} \mathcal{A}z(x_1,x_2) - z(x_1,x_2) & = & 1 & , \quad \forall (x_1,x_2) \in D \, , \\ z(x_1,x_2) & = & 2 & , \quad \forall (x_1,x_2) \in \partial D \, . \end{array} \right. \tag{3.16}$$

   Donner l'expression de la solution du problème (3.16) au point $x = (x_1,x_2) \in D$, en fonction du temps d'atteinte du bord de D, τ_x défini par :

$$\tau_x = \inf\{t \geq 0 \, ; \, X^x(t) \notin D\} \, ,$$

   où $\{X^x(t)\} = \{(X_1^x(t), X_2^x(t))\, ,\; t \geq 0\}$ est un processus de diffusion de générateur $\mathcal{A}$, tel que :

$$(X_1^x(0), X_2^x(0)) = (x_1,x_2) \, .$$

3. En déduire un algorithme de calcul approché de la solution z du problème (3.16).

4 Exploration markovienne

Les méthodes d'exploration markoviennes, ou méthodes de Monte-Carlo par chaîne de Markov (MCMC), ont été considérablement développées ces 10 dernières années (voir Robert [78] ou Fishman [35]). Si p est une loi de probabilité à simuler, l'idée est de l'exprimer comme la mesure stationnaire d'une chaîne de Markov. Deux alternatives s'offrent alors. La première est la méthode de simulation exacte de Propp et Wilson (section 4.1.3) qui consiste à coupler dans le passé plusieurs trajectoires de la chaîne. L'autre alternative consiste à simuler la chaîne dans le futur pendant suffisamment longtemps, jusqu'à ce qu'elle approche sa mesure stationnaire. Là deux alternatives sont encore possibles, la méthode séquentielle et la méthode parallèle. Dans la première on obtient un échantillon de taille n en extrayant n valeurs régulièrement espacées d'une seule trajectoire. Dans la seconde on simule n trajectoires indépendantes jusqu'à un certain temps d'arrêt. C'est surtout cette dernière que nous étudierons (section 4.1.5).

Quelques algorithmes d'optimisation stochastique seront introduits ensuite, principalement le recuit simulé, les algorithmes génétiques et l'algorithme MOSES. Bien que ces algorithmes simulent des chaînes de Markov non homogènes, on peut néanmoins utiliser les résultats sur la vitesse de convergence des chaînes homogènes pour étudier leur comportement, comme nous le ferons pour le recuit simulé en 4.2.2 et 4.2.3.

4.1 Comportement asymptotique

Les résultats décrivant la classification des états d'une chaîne de Markov sur un ensemble fini, ses mesures stationnaires, la convergence vers ces mesures, sont extrêmement classiques et se retrouvent dans de nombreux manuels [7, 15, 23, 33, 47, 50]. Pour l'essentiel, nous nous limiterons aux chaînes de Markov à temps discret *réversibles*, puisque ce sont elles que l'on rencontre dans la plupart des méthodes markoviennes de Monte-Carlo. Ce choix entraîne une grande simplification du traitement mathématique.

4.1.1 Mesures stationnaires et mesures réversibles

Soit $E = \{i, j, \ldots\}$ un ensemble fini. Soit $P = (p_{ij})$ la matrice de transition d'une chaîne de Markov homogène sur l'ensemble fini E (voir 3.1.2). La somme des coefficients de chaque ligne de P vaut 1. Autrement dit, le vecteur $\mathbb{1}_E$ dont toutes les coordonnées valent 1 est vecteur propre de P, associé à la valeur propre 1. Donc 1 est également valeur propre de ${}^tP$. Les mesures de probabilité, vecteurs propres de ${}^tP$ associés à la valeur propre 1 sont les états d'équilibre de la chaîne.

Définition 4.1 *Soit $p = (p_i)_{i \in E}$ une mesure de probabilité sur E. On dit que p est une* mesure stationnaire *de la chaîne de Markov de matrice de transition P, ou que la matrice P est p-*stationnaire*, si*

$$ {}^tP\, p \;=\; p\,. $$

Il faut comprendre cette définition par référence à la proposition 3.3. Soit $(X_n), n \in \mathbb{N}$ une chaîne de Markov de matrice de transition P. Supposons que la loi de X_0 soit une mesure stationnaire p, alors pour tout $n \geq 1$, la loi de X_n est encore p (d'où le nom

de mesure stationnaire). On démontre que toute matrice de transition admet au moins une mesure stationnaire. La réversibilité est un cas très particulier de stationnarité.

Définition 4.2 *Soit $p = (p_i)_{i\in E}$ une mesure de probabilité sur E. On dit que p est une* mesure réversible *pour la chaîne de Markov de matrice de transition P, ou que la matrice P est* p-réversible, *si*

$$p_i\, p_{ij} \;=\; p_j\, p_{ji}\;, \quad \forall i,j \in E\;. \tag{4.1}$$

Le livre de Kelly [49] est une bonne référence générale sur la réversibilité et ses applications. La relation (4.1) s'appelle *condition de bilan détaillé*. Observons tout d'abord qu'une mesure réversible est nécessairement stationnaire. En effet si on somme par rapport à j l'équation (4.1), on obtient

$$p_i \;=\; \sum_{j\in E} p_j\, p_{ji}\;, \quad \forall i \in E\;,$$

qui est la condition de stationnarité.

Si p est une mesure réversible et si la loi de X_m est p, alors non seulement la loi de X_{m+1} est encore p (stationnarité), mais on a :

$$I\!P[X_m = i \text{ et } X_{m+1} = j] \;=\; I\!P[X_m = j \text{ et } X_{m+1} = i]\;.$$

C'est la raison pour laquelle on parle de mesure réversible. Soit P une matrice de transition p-réversible. Soient i et j deux états tels que $p_i > 0$ et $p_j = 0$. Alors $p_{ij} = 0$. Donc la restriction de P à l'ensemble des états i tels que $p_i > 0$ est encore une matrice de transition, qui est réversible par rapport à la restriction de p à son support. Quitte à réduire l'espace d'états, on peut donc se ramener au cas où la mesure réversible p est strictement positive ($p_i > 0, \forall i \in E$). C'est ce que nous supposerons désormais.

La condition de bilan détaillé (4.1) s'écrit sous une forme matricielle qui nous sera très utile par la suite. Soit $p = (p_i)_{i\in E}$ une mesure de probabilité strictement positive sur E. Notons D la matrice diagonale dont le coefficient d'ordre (i,i) est $\sqrt{p_i}$.

$$D \;=\; Diag(\,(\sqrt{p_i}\,)\,,\; i \in E\,)\;.$$

On vérifie immédiatement que la matrice de transition P est p-réversible si et seulement si la matrice DPD^{-1} est symétrique.

Pour donner des exemples de chaînes admettant une mesure réversible, nous commençons par observer qu'on peut toujours "symétriser" une matrice de transition admettant p comme mesure stationnaire, pour la rendre p-réversible.

Proposition 4.3 *Soit P une matrice de transition et $p = (p_i)_{i\in E}$ une mesure stationnaire pour P, telle que pour tout i, $p_i > 0$. Soit P^* la matrice $D^{-2}\,{}^tPD^2$, et $Q = (P + P^*)/2$. Alors les matrices P^* et Q sont des matrices de transition, P^* est p-stationnaire et Q est p-réversible.*

La matrice P^* est celle de l'adjoint de l'opérateur de matrice P dans $L_2(p)$. Une matrice de transition P est p-réversible si l'opérateur est auto-adjoint dans $L_2(p)$. Par souci de

simplicité, nous avons choisi de nous en tenir à une présentation matricielle élémentaire, même si beaucoup de résultats s'écrivent de manière beaucoup plus élégante avec le formalisme de l'analyse hilbertienne.

L'observation suivante est immédiate, mais elle contient bon nombre d'applications.

Proposition 4.4 *Supposons que P soit une matrice de transition symétrique, alors P admet la loi uniforme sur E comme mesure réversible.*

Exemple : Marches aléatoires symétriques.

Supposons E muni d'une structure de graphe non orienté $G = (E, A)$, où A désigne l'ensemble des arêtes :

$$A \subset \Big\{ \{i,j\}\ ,\ i,j \in E \Big\}\ .$$

Soit r le degré maximal d'un sommet du graphe.

$$r \ = \ \max_{i \in E} \Big\{ \Big| \{ j \in E \ :\ \{i,j\} \in A \} \Big| \Big\}\ ,$$

où $|.|$ désigne le cardinal d'un ensemble fini. Définissons la matrice de transition $P = (p_{ij})$ par

$$\begin{aligned} p_{ij} \ &= \ \frac{1}{r} \quad \text{si } \{i,j\} \in A\ , \\ &= \ 0 \quad \text{si } \{i,j\} \notin A\ , \end{aligned}$$

les coefficients diagonaux étant tels que la somme des éléments d'une même ligne vaut 1. La chaîne de Markov de matrice de transition P s'appelle *marche aléatoire symétrique* sur le graphe G. La matrice P est, à une transformation près, ce que les combinatoriciens nomment le laplacien du graphe G (voir [11]). Cet exemple est à rapprocher de la proposition 3.6.

Il existe une analogie étroite entre les chaînes de Markov symétriques et les réseaux électriques (voir [28]). Les états de E sont vus comme les sommets d'un réseau, reliés par des lignes électriques. L'analogue de la probabilité de transition p_{ij} est la *conductance* (inverse de la résistance) de la ligne reliant i à j.

Des critères pour vérifier si une matrice de transition donnée admet ou non une mesure réversible ont été donnés par Kolmogorov (voir [49]). Nous nous intéresserons plutôt ici à la construction d'une matrice de transition p-réversible, quand p est une mesure donnée. Voici une méthode générale.

Proposition 4.5 *Soit $Q = (q_{ij})$ une matrice de transition sur E, vérifiant*

$$q_{ij} > 0 \implies q_{ji} > 0\ , \quad \forall i,j \in E\ .$$

Soit $p = (p_i)_{i \in E}$ une loi de probabilité strictement positive sur E. Définissons la matrice de transition $P = (p_{ij})$ de la façon suivante : pour $i \neq j$

$$\begin{aligned} p_{ij} \ &= \ q_{ij} \ \min \left\{ \frac{p_j\, q_{ji}}{p_i\, q_{ij}}\ ,\ 1 \right\} \quad \textit{si } q_{ij} \neq 0\ , \\ &= \ 0 \qquad\qquad\qquad\qquad\qquad\quad \textit{sinon}\ . \end{aligned} \tag{4.2}$$

Les coefficients diagonaux sont tels que la somme des éléments d'une même ligne vaut 1.
La matrice de transition P est p-réversible.

On peut voir la proposition ci-dessus comme une extension de la méthode de rejet qui permet de simuler une loi de probabilité quelconque à partir d'une autre. Dans la proposition 4.5, la matrice Q s'appelle *matrice de sélection*. L'algorithme correspondant porte le nom d'*algorithme de Metropolis*.

Initialiser X
$t \longleftarrow 0$
Répéter
    $i \longleftarrow X$
    choisir j avec probabilité q_{ij}
    $\rho \longleftarrow (p_j * q_{ji})/(p_i * q_{ij})$
    Si ($\rho \geq 1$) alors
        $X \longleftarrow j$
    Sinon
        Si (Random $< \rho$) alors
            $X \longleftarrow j$
        finSi
    finSi
    $t \longleftarrow t+1$
Jusqu'à (arrêt de la simulation)

Tel qu'il est écrit, cet algorithme n'est évidemment pas optimisé. Dans la plupart des applications, la matrice de transition Q est symétrique, ce qui simplifie le calcul du coefficient de rejet ρ (remarquer qu'il vaut mieux dans ce cas tester si $p_j < p_i$ avant de faire le calcul de ρ). Très souvent, l'espace des états est naturellement muni d'une structure de graphe déduite du contexte d'application, et on choisit alors pour Q la matrice de transition de la marche aléatoire symétrique sur ce graphe.

Exemple : Ensemble des stables d'un graphe.

Soit $G = (S, B)$ un graphe fini non orienté, dont S est l'ensemble des sommets et $B \subset S \times S$ l'ensemble des arêtes. On considère l'ensemble E des *stables* de ce graphe. Un sous-ensemble R de S est dit *stable* si

$$\forall x, y \in R\ , \quad \{x, y\} \notin B\ .$$

L'algorithme suivant simule une chaîne de Markov irréductible et apériodique qui admet la loi uniforme sur E pour mesure réversible.

$R \longleftarrow \emptyset$
$t \longleftarrow 0$
Répéter
    choisir x au hasard dans S
    Si ($x \in R$)

alors $R \longleftarrow R \setminus \{x\}$
sinon
Si $(\forall y \in R\,,\ \{x,y\} \notin B)$
alors $R \longleftarrow R \cup \{x\}$
finSi
finSi
$t \longleftarrow t+1$
Jusqu'à (arrêt de la simulation)

L'ensemble des stables E est un sous-ensemble de l'ensemble E' de tous les sous-ensembles de S. E' est naturellement muni d'une structure de graphe (hypercube), pour laquelle deux sous-ensembles sont voisins s'ils diffèrent en un seul élément. L'algorithme ci-dessus simule la marche aléatoire sur cet hypercube (à chaque pas on choisit un élément de S au hasard, on le rajoute à l'ensemble courant s'il n'y était pas, on le retranche sinon). Pour obtenir une marche aléatoire symétrique sur l'ensemble des stables, il suffit d'imposer la contrainte que l'on ne peut rajouter un élément x à R que si $R \cup \{x\}$ est encore stable.

Supposons maintenant que l'on veuille simuler la loi de probabilité sur E telle que la probabilité de tout stable R est donnée par

$$p_R \;=\; \frac{1}{Z}\,\lambda^{|R|}\,,$$

où λ est un réel strictement positif, et $Z = \sum_{R\in E} \lambda^{|R|}$. Il est inutile de calculer la constante de normalisation Z pour appliquer l'algorithme de Metropolis (proposition 4.5). Pour $\lambda > 1$, l'algorithme est le suivant (on le modifierait de manière évidente pour $\lambda < 1$).

$R \longleftarrow \emptyset$
$t \longleftarrow 0$
Répéter
choisir x au hasard dans S
Si $(x \in R)$
alors
Si (Random $< 1/\lambda$) alors $R \longleftarrow R - \{x\}$
finSi
sinon
Si $(\forall y \in R\,,\ \{x,y\} \notin B)$
alors $R \longleftarrow R \cup \{x\}$
finSi
finSi
$t \longleftarrow t+1$
Jusqu'à (arrêt de la simulation)

Dans la section suivante, nous décrivons une première application des méthodes MCMC au dénombrement de grands ensembles.

4.1.2 Dénombrement par chaîne de Markov

Ce qui suit s'inspire de Aldous [1, 2] (voir aussi [68, 88]).

Etant donné un domaine (ensemble fini ou bien ouvert connexe dans $\mathbb{R}^d$), les deux problèmes consistant d'une part à évaluer sa taille (cardinal ou bien volume d-dimensionnel) et d'autre part à simuler la loi uniforme sur ce domaine sont liés, et même équivalents au moins au sens de la complexité algorithmique. Une méthode générale pour la simulation de la loi uniforme sur un domaine D quelconque est la méthode de rejet. L'algorithme consiste à tirer au hasard dans un ensemble D' qui le contient et rejeter tous les tirages qui ne tombent pas dans D (cf. proposition 2.11). Le coût de cet algorithme est proportionnel au nombre de passages dans la boucle principale, qui suit la loi géométrique de paramètre $v(D)/v(D')$. Dans le cas d'un domaine de $\mathbb{R}^d$, le coût de la méthode de rejet croît exponentiellement avec la dimension de l'espace. Prenons comme exemple la boule unité de $\mathbb{R}^d$.

$$B_d \;=\; \{x = (x_i) \in \mathbb{R}^d \,;\, \sum x_i^2 < 1\} \;.$$

Pour simuler la loi uniforme sur B_d, on peut l'inclure dans le cube C_d :

$$C_d \;=\; \{x = (x_i) \in \mathbb{R}^d \,;\; |x_i| < 1 \,,\; \forall i\} \;.$$

Le coût de l'algorithme de rejet correspondant est proportionnel au rapport du volume de C_d à celui de B_d. Si d est pair, ce rapport vaut :

$$\frac{v(C_d)}{v(B_d)} \;=\; \frac{2^d\,(d/2)!}{\pi^{d/2}} \;.$$

Pour $d = 10$ et $d = 20$, on trouve respectivement $4\,10^2$ et $4\,10^7$. Il n'existe pas d'algorithme connu pour simuler de manière exacte la loi uniforme sur un ensemble convexe borné K quelconque, avec un coût polynomial en la dimension de l'espace. Il n'en existe pas non plus pour calculer son volume.

Soit K un convexe borné de $\mathbb{R}^d$, h un pas de discrétisation et $K_h = K \cap h\mathbb{Z}^d$. On peut approcher le volume de K par $h^d|K_h|$, ce qui ramène le calcul de $v(K)$ à un problème de dénombrement. De même la loi uniforme sur K_h est une approximation de la loi uniforme sur K.

Nous avons vu en 4.1.1 une méthode générale de simulation approchée pour la loi uniforme sur un ensemble fini E quelconque. Il s'agit de suivre pendant assez longtemps une chaîne de Markov sur E dont la matrice de transition soit irréductible et symétrique. Dans la pratique, on inclut l'ensemble E dans l'ensemble des sommets d'un graphe régulier (E', A) (par exemple $K_h \subset h\mathbb{Z}^d$). On simule alors la marche aléatoire symétrique sur (E', A), partant dans E, en ne conservant que les pas qui restent dans E.

Initialiser X
$t \longleftarrow 0$
Répéter

$i \longleftarrow X$
choisir j au hasard parmi les voisins de i dans E'
Si $(j \in E)$ alors $X \longleftarrow j$
finSi
$t \longleftarrow t+1$
Jusqu'à (arrêt de la simulation)

On peut voir cet algorithme comme un cas particulier de l'algorithme de Metropolis (proposition 4.5), ou comme une extension de la technique classique de rejet. Soit (X_t) la chaîne de Markov produite par l'algorithme ci-dessus. Comment l'utilise-t-on pour évaluer la taille de E ? Soit A un sous-ensemble de E. D'après le théorème 4.12, la proportion de temps que la chaîne passe dans A converge vers la probabilité de A pour la mesure réversible.

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{1}_A(X_t) \;=\; \frac{|A|}{|E|} \,. \tag{4.3}$$

Si le cardinal de A est connu ou a été calculé auparavant, on en déduira une évaluation du cardinal de E, à condition que le rapport $|A|/|E|$ ne soit pas trop petit. En pratique, on déterminera une suite emboîtée d'ensembles

$$A_1 \subset A_2 \subset \cdots \subset A_k \subset E \,.$$

Le premier sous-ensemble A_1 est choisi de sorte qu'il puisse être dénombré exactement. Puis les rapports $|A_i|/|A_{i+1}|$ sont estimés par (4.3). On multiplie alors ces rapports pour en déduire une estimation de la taille de E.

4.1.3 Simulation exacte d'une mesure stationnaire

Une méthode récente, et extrêmement astucieuse, due à J. Propp et D. Wilson [72], permet si l'on souhaite simuler une loi de probabilité $p = (p_i)$ et que l'on dispose d'une matrice de transition $P = (p_{ij})$ pour laquelle p est stationnaire, de construire un algorithme de simulation exacte de p, avec un test d'arrêt explicite. Pour comprendre cette méthode, repartons de la définition algorithmique 3.1.

Soit $(U_t)_{t\in\mathbb{N}}$ une suite de VAIID, à valeurs dans un espace mesurable $\mathcal{U}$ et $\mathbb{P}$ leur loi de probabilité. Soit $E = \{i, j, \ldots\}$ un ensemble fini, et Φ une application de $E \times \mathcal{U}$ dans E telle que pour tout $i, j \in E$ et pour tout $t \in \mathbb{N}$:

$$\mathbb{P}[\, \Phi(i, U_t) = j \,] = p_{i,j} \,.$$

L'application de E dans E qui à i associe $\Phi(i, U_t)$ est une application aléatoire de E dans E. Si on compose deux telles applications, on obtient une nouvelle application aléatoire, et l'image de cette nouvelle application a au plus autant d'éléments que les images de chacune des deux composantes. Si on compose entre elles suffisamment de ces applications, on peut s'attendre à ce que l'image de la composée finisse par ne contenir qu'un seul élément. L'idée de Propp et Wilson consiste à composer successivement *à gauche* (vers le passé) les applications $\Phi(i, U_t)$ jusqu'à ce que la composée devienne constante. Miraculeusement, la valeur de cette constante est i avec probabilité p_i.

La suite d'applications aléatoires que l'on souhaite définir est une chaîne de Markov à valeurs dans E^E. Soit F une application aléatoire de E dans E (variable aléatoire à valeurs dans E^E), indépendante de la suite (U_t). On définit la suite $(F_t)_{t\in\mathbb{N}}$ d'applications aléatoires de E dans E par $F_0 = F$ et pour tout $t \geq 0$:

$$F_{t+1}(i) = F_t(\Phi(i, U_t)) \ , \quad \forall i \in E \ .$$

Autrement dit, en notant Φ_t l'application aléatoire $\Phi(\cdot, U_t)$:

$$F_t = F \circ \Phi_1 \circ \cdots \circ \Phi_{t-1} \ .$$

On vérifie immédiatement que la suite $(F_t)_{t\in\mathbb{N}}$ est bien une chaîne de Markov sur E^E. On notera $\mathbb{P}_F$ sa loi.

On suppose que la matrice P est irréductible et apériodique. Pour tout $i \in E$, on note $f^{(i)}$ l'application constamment égale à i ($f^{(i)}(j) = i \ , \ \forall j \in E$). Le fait que P soit irréductible et apériodique entraîne que les applications $f^{(i)}$, $i \in E$ sont les seuls états absorbants de la chaîne de Markov (F_t). Mais cela ne suffit pas pour affirmer que la chaîne (F_t) est absorbée avec probabilité 1 dans l'un de ces états. Voici un contre exemple. Soit $E = \{a, b\}$ et P la matrice carrée d'ordre 2 dont les quatre coefficients valent 1/2. Soit (U_t) une suite de VAIID de Bernoulli, avec $\mathbb{P}[U_t = 0] = \mathbb{P}[U_t = 1] = 1/2$. Définissons Φ par :

$$\Phi(a, 0) = a \ , \ \Phi(b, 0) = b \ , \ \Phi(a, 1) = b \ , \ \Phi(b, 1) = a \ .$$

On a bien $\mathbb{P}[\Phi(i, U_t) = j] = 1/2$. Pourtant, si F est l'application identique alors pour tout t, F_t est égale à l'application identique ou bien à la transposition de a et b chacune avec probabilité 1/2, et n'est donc jamais constante. Pour éviter ce piège nous imposons que le cardinal de l'image puisse toujours diminuer.

$$\forall E' \subset E \ , \ |E'| > 1 \Longrightarrow \mathbb{P}[\ |\Phi(E', U_t)| < |E'|\ ] > 0 \ . \tag{4.4}$$

Théorème 4.6 *Soit T le temps d'absorption de la chaîne (F_n) :*

$$T = \inf\{t \in \mathbb{N}, \ F_t \ \textit{est constante}\} \ .$$

Sous l'hypothèse (4.4)*, T est une variable aléatoire presque sûrement finie, d'espérance finie. De plus, si I désigne l'application identique de E dans E, alors, pour tout $i \in E$:*

$$\mathbb{P}_I[F_T = f^{(i)}] = p_i \ .$$

Démonstration : Par l'hypothèse (4.4), toutes les applications autres que les applications constantes sont des états transients de la chaîne (F_t), et la chaîne atteint nécessairement l'un des états absorbants au bout d'un temps fini. En fait on peut donner une évaluation grossière du temps d'absorption T. L'hypothèse que la matrice de transition $P = (p_{i,j})_{i,j\in E}$ est irréductible et apériodique, entraîne que toutes les puissances de P à partir d'un certain rang ont tous leurs coefficients positifs. On notera m le plus petit

entier tel que la matrice P^m a tous ses coefficients strictement positifs. Il existe donc α strictement inférieur à 1 tel que :

$$\mathbb{P}_F[T > m] < \alpha \ ,$$

quelle que soit la loi de F. Par la propriété de Markov, pour tout $k \geq 1$:

$$\mathbb{P}_F[T > km] < \alpha^k \ .$$

Et donc :

$$\begin{aligned} \mathbb{E}[T] &= \sum_{h=0}^{\infty} \mathbb{P}[T > h] = \sum_{k=0}^{\infty} \sum_{h=0}^{m-1} \mathbb{P}[T > km + h] \\ &< \sum_{k=0}^{\infty} m\alpha^k = \frac{m}{1-\alpha} \ . \end{aligned}$$

On peut donc s'attendre à ce que la chaîne soit absorbée en un temps raisonnable. Reste à montrer que c'est bien la mesure stationnaire p qui est atteinte au moment de l'absorption. Pour toute application f fixée, de E dans E, notons :

$$p_f(i) = \mathbb{P}_f[\, F_T = f^{(i)} \,] \ .$$

Dire que pour $F_0 \equiv f$, $F_T = f^{(i)}$ est équivalent à $\Phi_1 \circ \cdots \circ \Phi_N \in f^{-1}(i)$, soit aussi à $I \circ \Phi_1 \circ \cdots \circ \Phi_N \in f^{-1}(i)$. Mais si la chaîne, partant de I arrive à l'instant N dans l'ensemble $f^{-1}(i)$, elle sera nécessairement absorbée en un point de cet ensemble. Réciproquement, si la chaîne partant de I est absorbée en un point de $f^{-1}(i)$, alors, en composant à droite avec f, la chaîne partant de f sera absorbée en I. On a donc :

$$p_f(i) = \sum_{j \in f^{-1}(i)} p_I(j) \ .$$

Mais en décomposant sur les valeurs de la première application aléatoire Φ_1 :

$$\begin{aligned} p_I(i) &= \sum_{f \in E^E} \mathbb{P}[\Phi_1 = f] p_f(i) \\ &= \sum_{f \in E^E} \mathbb{P}[\Phi_1 = f] \sum_{j \in f^{-1}(i)} p_I(j) \\ &= \sum_{j \in E} p_I(j) \sum_{\substack{f \in E^E \\ f(j)=i}} \mathbb{P}[\Phi_1 = f] \\ &= \sum_{j \in E} p_I(j) \, \mathbb{P}[\Phi_1(j) = i] \\ &= \sum_{j \in E} p_I(j) \, p_{ji} \ . \end{aligned}$$

La mesure p_I est donc bien égale à la mesure stationnaire p (qui est unique puisque P est irréductible). □

Pour simuler exactement la mesure p, il suffit donc en théorie de composer les applications aléatoires $\Phi_1 \circ \cdots \circ \Phi_t$ jusqu'à ce que la composée soit constante. La valeur de la constante obtenue est distribuée selon la loi p. Rappelons qu'on n'utilise une méthode MCMC pour simuler une loi de probabilité que quand l'espace d'états est trop grand pour que l'on puisse utiliser une méthode directe. Calculer explicitement les images de tous les éléments de l'ensemble E est donc exclu. Là intervient un point crucial de la méthode de Propp et Wilson. Dans la définition de Φ, on dispose d'une grande latitude. Rien ne dit en particulier que les images $(\Phi(i, U))$ doivent être des variables indépendantes. On pourra donc choisir Φ de manière à ce qu'il suffise que les images par $\Phi_1 \circ \cdots \circ \Phi_t$ de quelques états seulement coïncident, pour pouvoir affirmer que l'application est constante. L'adaptation de la méthode à toutes sortes de problèmes concrets a fait l'objet de nombreuses publications ces 5 dernières années. Nous ne la développerons pas ici.

On pourrait penser qu'il aurait été plus naturel de composer les applications aléatoires à droite (vers le futur), plutôt qu'à gauche. On définirait ainsi une chaîne de Markov, $(G_t)_{t\in\mathbb{N}}$, à valeurs dans E^E, par $G_0 = G$, indépendante de la suite (U_t), et pour tout $t \geq 0$:

$$G_{t+1}(i) = \Phi(G_t(i), U_t) \ , \quad \forall i \in E \ .$$

Sous l'hypothèse (4.4), l'ensemble des applications constantes $\{f^{(i)}, \ i \in E\}$ est l'unique classe récurrente de cette chaîne de Markov. Le temps d'atteinte S de cette classe récurrente est encore presque sûrement fini. On pourrait s'attendre à ce que l'application constante G_M soit $f^{(i)}$ avec probabilité p_i. Ce n'est pas nécessairement le cas. Supposons par exemple qu'il existe un couple $(i, j) \in E$ tel que $p_{i,j} = 1$. On a alors :

$$\mathbb{P}_G[\, G_S = f^{(j)} \,] = 0 \ .$$

4.1.4 Convergence vers une mesure réversible

En dehors de la simulation exacte du paragraphe 4.1.3, le moyen de simuler une mesure stationnaire est de simuler une chaîne de Markov pendant un nombre de pas suffisant pour que la mesure stationnaire soit à peu près atteinte. Le problème est de déterminer ce "nombre de pas suffisant". La question de la vitesse de convergence est d'une grande importance pratique. Elle n'est encore qu'imparfaitement résolue. Nous donnons quelques éléments de réponse dans le cas réversible (voir Saloff-Coste [85] pour plus de détails).

Nous considérons ici une matrice de transition P sur un ensemble fini E, admettant comme mesure réversible la mesure p, strictement positive sur E. Si (X_t) est une chaîne de Markov de matrice de transition P, partant de l'état i en $t = 0$, alors la loi de X_t est la i-ième ligne de la matrice P^t, que nous noterons $P_i^{(t)}$, son coefficient d'ordre j étant noté $p_{ij}^{(t)}$ (probabilité de transition de i à j en t pas). Notre but dans ce paragraphe est d'étudier le comportement asymptotique de $P_i^{(t)}$.

Nous commençons par une description du spectre de P. Nous notons encore D la

matrice diagonale

$$D \;=\; Diag(\;(\sqrt{p_i}\;),\; i \in E\;)\;.$$

La matrice DPD^{-1} est symétrique. Notons I la matrice identité sur E.

Lemme 4.7 *Les formes quadratiques de matrices $I - DPD^{-1}$ et $I + DPD^{-1}$ sont positives.*

Démonstration : La forme quadratique de matrice $I - DPD^{-1}$ associe au vecteur $u = (u_i)_{i\in E}$ la quantité :

$$\mathcal{Q}(u,u) \;=\; {}^t u\,(I - DPD^{-1})\,u \;=\; \frac{1}{2}\sum_{i,j\in E} p_i\,p_{ij}\left(\frac{u_i}{\sqrt{p_i}} - \frac{u_j}{\sqrt{p_j}}\right)^2 . \tag{4.5}$$

La forme quadratique de matrice $I+DPD^{-1}$ associe au vecteur $u = (u_i)_{i\in E}$ la quantité :

$$\mathcal{Q}'(u,u) \;=\; {}^t u\,(I + DPD^{-1})\,u \;=\; \frac{1}{2}\sum_{i,j\in E} p_i\,p_{ij}\left(\frac{u_i}{\sqrt{p_i}} + \frac{u_j}{\sqrt{p_j}}\right)^2 . \tag{4.6}$$

□

Théorème 4.8 *Soit P une matrice de transition p-réversible. Alors*

1. *Les valeurs propres de la matrice P sont toutes réelles et comprises entre -1 et 1.*
2. *La valeur propre 1 est simple si et seulement si :*

$$\forall i,j \in E\;,\; \exists k_1,\ldots,k_\ell \in E\;,\quad p_{ik_1}p_{k_1k_2}\ldots p_{k_\ell j} > 0\;. \tag{4.7}$$

3. *Si l'un des coefficients diagonaux p_{ii} de P est non nul, et si (4.7) est vrai, alors -1 n'est pas valeur propre de P.*

Ces propriétés du spectre des matrices de transition réversibles sont des cas particuliers de propriétés plus générales que l'on déduit classiquement du théorème de Perron-Frobenius (voir par exemple [23]). L'hypothèse de réversibilité permet d'en donner une démonstration élémentaire.

Démonstration : Comme DPD^{-1} est symétrique, ses valeurs propres sont toutes réelles, et ce sont celles de P. Les valeurs propres de $I-DPD^{-1}$ et de $I+DPD^{-1}$ sont positives ou nulles. Donc celles de P sont comprises entre -1 et 1. Un vecteur u non nul annule (4.5) si et seulement si pour tout couple i,j tel que $p_{ij} > 0$, on a

$$\frac{u_i}{\sqrt{p_i}} = \frac{u_j}{\sqrt{p_j}}\;.$$

L'ensemble des vecteurs u vérifiant ceci est de dimension 1 si et seulement si tout état i est relié à tout autre état j par une suite de transitions, de probabilités strictement

positives (propriété (4.7)). On dit alors que la chaîne (ou sa matrice de transition) est *irréductible*.

Supposons que la forme quadratique (4.6) s'annule pour un vecteur u. Pour tout i tel que $p_{ii} > 0$, on doit avoir $u_i = 0$, mais aussi $u_j = 0$ pour tout j tel que $p_{ij} > 0$. Supposons que (4.7) soit vrai, alors le vecteur u doit être nul. Donc la forme quadratique (4.6) est strictement positive et -1 n'est pas valeur propre de P. Si (4.7) est vrai et si -1 est valeur propre de P, on dit que la chaîne (ou sa matrice de transition) est *périodique* de période 2. Dans le cas contraire, on dit que la chaîne (ou sa matrice de transition) est *apériodique*. □

Nous supposerons désormais que P est irréductible et apériodique. Dans ce cas, 1 est valeur propre simple de P, et toutes les autres valeurs propres sont strictement comprises entre -1 et 1. De plus p est la seule mesure stationnaire de la chaîne.

Notons d le cardinal de E et $(\lambda_\ell)\,,\ \ell = 1, \ldots, d$ les valeurs propres de P, rangées par ordre décroissant, avec leur multiplicité.

$$\lambda_1 = 1 > \lambda_2 \geq \cdots \geq \lambda_d > -1\ .$$

Les matrices symétriques DP^tD^{-1} sont diagonalisables dans une même base orthonormée de vecteurs propres. Soit $(\phi_\ell)\,,\ \ell = 1, \ldots, d$ une telle base, où ϕ_ℓ est un vecteur propre associé à λ_ℓ. Pour ϕ_1 on choisira :

$$\phi_1 \ = \ (\sqrt{p_j})_{j \in E}\ .$$

Pour tout couple (u, v) de vecteurs de $\mathbb{R}^d$, on a :

$$^t u\, DP^tD^{-1}\, v \ = \ \sum_{\ell=1}^{d} {}^t u \phi_\ell \ {}^t v \phi_\ell \ \lambda_\ell^t\ . \tag{4.8}$$

Comme cas particulier, on obtient le résultat suivant.

Proposition 4.9

$$p_{ij}^{(t)} \ = \ p_j \ + \ \frac{\sqrt{p_j}}{\sqrt{p_i}} \sum_{\ell=2}^{d} \phi_\ell(i)\phi_\ell(j)\ \lambda_\ell^t\ . \tag{4.9}$$

Démonstration : Il suffit d'appliquer (4.8) à

$$u \ = \ \frac{1}{\sqrt{p_i}} \mathbb{1}_{\{i\}} \quad \text{et} \quad v \ = \ \sqrt{p_j}\, \mathbb{1}_{\{j\}}\ .$$

□

La formule (4.9) montre que la suite des matrices P^t converge à vitesse exponentielle vers la matrice dont toutes les lignes sont égales à ${}^t p$. Autrement dit, quel que soit le point de départ de la chaîne (valeur de X_0), la loi de X_t converge vers la mesure réversible p. De plus, pour s fixé et t tendant vers l'infini, les variables aléatoires X_s

et X_{s+t} sont asymptotiquement indépendantes. Le problème dans les applications est d'évaluer précisément la valeur de t à partir de laquelle on peut considérer avec une marge d'erreur fixée que la loi de X_t est p, ou que X_s et X_{s+t} sont indépendantes. La formule (4.9) n'est pas d'un grand secours car, sauf cas très particulier, on ne peut pas diagonaliser explicitement la matrice DPD^{-1}. Il existe dans la littérature de nombreuses majorations, qui expriment en substance la même idée de convergence à vitesse exponentielle vers la mesure d'équilibre, que ce soit dans le cas réversible ou dans le cas général, pour des chaînes à temps discret ou à temps continu (voir Saloff-Coste [85] ou Diaconis et Saloff-Coste [27]). Plusieurs notions de distance permettent de mesurer l'écart entre la loi de la chaîne à l'instant t et sa mesure réversible. Nous n'utiliserons que deux d'entre elles, la distance en variation totale et la distance du chi-deux. La distance en variation totale entre $P_i^{(t)}$ et p est :

$$\|P_i^{(t)} - p\| = \max_{F \subset E} |P_i^{(t)}(F) - p(F)| = \frac{1}{2} \sum_{j \in E} |P_i^{(t)}(j) - p_j| \; . \tag{4.10}$$

La distance du chi-deux est :

$$\chi(P_i^{(t)}, p) = \sum_{j \in E} \frac{(P_i^{(t)}(j) - p_j)^2}{p_j} \; . \tag{4.11}$$

Par l'inégalité de Cauchy-Schwarz, on a :

$$\|P_i^{(t)} - p\| \leq \frac{1}{2} (\chi(P_i^{(t)}, p))^{1/2} \; . \tag{4.12}$$

La proposition suivante donne la majoration la plus simple pour la distance du chi-deux. Nous utiliserons plutôt la distance en variation totale au paragraphe 4.1.5.

Proposition 4.10 *Soit p une mesure de probabilité strictement positive sur E et P une matrice de transition irréductible apériodique et p-réversible. Soient λ_ℓ, $\ell = 1, \ldots, d$ les valeurs propres de P rangées par ordre décroissant et*

$$\alpha \; = \; \max\{|\lambda_\ell| \, , \; \ell = 2, \ldots, d\} \; < \; 1 \; .$$

Alors

$$\chi(P_i^{(t)}, p) \; \leq \; \frac{1}{p_i} \alpha^{2t} \; . \tag{4.13}$$

Démonstration : Ecrivons (4.9) sous la forme suivante.

$$\sqrt{p_j} \left(\frac{p_{ij}^{(t)}}{p_j} - 1 \right) \; = \; \frac{1}{\sqrt{p_i}} \sum_{\ell=2}^{d} \phi_\ell(i) \phi_\ell(j) \; \lambda_\ell^t \; .$$

Ceci est la j-ième coordonnée d'un vecteur de $\mathbb{R}^d$ dont les coordonnées dans la base (ϕ_ℓ) sont

$$\frac{1}{\sqrt{p_i}} \, \phi_\ell(i) \lambda_\ell^t \; , \quad \ell = 2, \ldots, d \; .$$

Comme la base (ϕ_ℓ) est orthonormée, le carré de la norme de ce vecteur vaut

$$\sum_{j\in E} p_j \left(\frac{p_{ij}^{(t)}}{p_j} - 1\right)^2 \;=\; \frac{1}{p_i}\sum_{\ell=2}^{d} \phi_\ell^2(i)\;\lambda_\ell^{2t}\;. \tag{4.14}$$

En majorant chacun des facteurs λ_ℓ^{2t} par α^{2t}, on obtient

$$\begin{aligned}\chi(P_i^{(t)},p) &\le \frac{1}{p_i}\,\alpha^{2t}\sum_{\ell=2}^{d}\phi_\ell^2(i)\\ &\le \frac{1}{p_i}\,\alpha^{2t}\;.\end{aligned}$$

□

Voici une autre inégalité du même type, pour l'espérance d'une fonction quelconque de X_t.

Proposition 4.11 *Avec les hypothèses de la proposition* 4.10*, soit f une application de E dans $\mathbb{R}$. Notons :*

$$\mathbb{E}_p[f] \;=\; \sum_{j\in E} f(j)p_j \qquad et \qquad Var_p[f] \;=\; \sum_{j\in E}(f(j)-\mathbb{E}_p[f])^2 p_j\;.$$

Soit $(X_t), t\in\mathbb{N}$ une chaîne de Markov de matrice de transition P. Alors pour tout $i\in E$ et tout $t\in\mathbb{N}$, on a :

$$\Big(\,\mathbb{E}[f(X_t)\,|\,X_0=i]-\mathbb{E}_p[f]\,\Big)^2 \;\le\; \frac{1}{p_i}\,\alpha^{2t}\,Var_p[f]\;. \tag{4.15}$$

Démonstration : Appliquons (4.8) aux vecteurs u et v, où u est le vecteur $\mathbb{1}_{\{i\}}/\sqrt{p_i}$, et $v=(v_j)_{j\in E}$ est défini par :

$$v_j \;=\; \sqrt{p_j}(f(j)-\mathbb{E}_p[f])\;.$$

Alors

$${}^t u\,DP^tD^{-1}\,v \;=\; \sum_{j\in E} p_{ij}^{(t)}\Big(f(j)-\mathbb{E}_p[f]\Big) \;=\; \mathbb{E}[f(X_t)\,|\,X_0=i]-\mathbb{E}_p[f]\;.$$

Mais aussi

$$\begin{aligned}{}^t u\,DP^tD^{-1}\,v &= \sum_{\ell=1}^{d} {}^t u\phi_\ell\,{}^t v\phi_\ell\,\lambda_\ell^t\\ &= 0+\sum_{\ell=2}^{d}\frac{\phi_\ell(i)}{\sqrt{p_i}}\,{}^t v\phi_\ell\;\lambda_\ell^t\;.\end{aligned}$$

Par l'inégalité de Cauchy-Schwarz, on obtient

$$\begin{aligned}\Big(\mathbb{E}[f(X_t)\,|\,X_0=i]-\mathbb{E}_p[f]\Big)^2 &\le \sum_{\ell=2}^{d}\frac{\phi_\ell^2(i)}{p_i}\lambda_\ell^{2t}\;\sum_{\ell=2}^{d}({}^t v\phi_\ell)^2\\ &\le \frac{1}{p_i}\,\alpha^{2t}\sum_{\ell=1}^{d}\phi_\ell^2(i)\;\sum_{\ell=1}^{d}({}^t v\phi_\ell)^2\;.\end{aligned}$$

Dans cette dernière relation, la première somme vaut 1 (la base (ϕ_ℓ) est orthonormée), et la seconde vaut

$$\|v\|^2 \;=\; \sum_{j\in E} p_j \,(f(j) - I\!E_p[f])^2 \;=\; Var_p[f]\;.$$

□

En général, l'espace d'états E est très grand, et on ne cherche pas à estimer simultanément toutes les probabilités p_j. On souhaite par contre estimer l'espérance d'une fonction f de la chaîne à l'équilibre (par exemple une fonction de coût). Les quantités $I\!E_p[f]$ et $Var_p[f]$ sont l'espérance et la variance de f sous la mesure réversible, c'est-à-dire les limites de $I\!E[f(X_t)]$ et $Var[f(X_t)]$ quand t tend vers l'infini. En pratique, la simulation de la chaîne part d'un état i fixé, et on cherche à savoir au bout de combien de temps l'équilibre est atteint avec une précision donnée. Les propositions 4.10 et 4.11 montrent que ce temps est d'autant plus court que α est loin de 1. Malheureusement, on ne connaît pas en général la valeur de α. On est alors amené à en donner des majorations, et de nombreuses techniques ont été inventées pour cela. Nous ne développerons pas cet aspect, pour lequel nous renvoyons à [85, 27].

Au vu de la proposition 4.11, il paraît naturel d'estimer $I\!E_p[f]$ par une moyenne des valeurs prises par f sur une trajectoire de la chaîne, suivie suffisamment longtemps. Ceci est justifié par le théorème suivant.

Théorème 4.12 *Soit p une loi de probabilité strictement positive sur E et P une matrice de transition p-réversible et irréductible. Soit f une fonction de E dans $I\!R$ et (X_t) une chaîne de Markov de matrice de transition P. Alors presque sûrement,*

$$\lim_{t\to\infty} \frac{f(X_1) + \cdots + f(X_t)}{t} \;=\; I\!E_p[f]\;.$$

La suite de variables aléatoires $(f(X_t))$ vérifie donc la loi des grands nombres. Mais pour appliquer le théorème 4.12, il faut pouvoir estimer l'erreur commise quand on estime $I\!E_p[f]$ par la moyenne empirique des $f(X_s)\,,\ s = 1, \ldots, t$. Dans le cas de variables indépendantes, nous disposons pour cela du théorème central limite pour en déduire des intervalles de confiance. Un théorème central limite est vrai pour la suite $(f(X_t))$ (voir [22] p. 94). Mais la variance asymptotique, qui détermine l'amplitude des intervalles de confiance, n'est pas $Var_p[f]$. Elle est en général impossible à calculer, et peut être très grande. Se pose d'autre part le problème de déterminer la valeur de n pour laquelle on peut considérer avec une bonne approximation que la moyenne des $f(X_s)\,,\ s = 1, \ldots, t$ est distribuée suivant une loi normale. Il est intuitivement clair qu'on a peu de chances d'obtenir de bons résultats si la chaîne n'a pas atteint son état d'équilibre bien avant n. On devrait donc effectuer une moyenne des valeurs $f(X_t)$ sur un échantillon de très grande taille, ce qui serait extrêmement coûteux.

Ce n'est pas ainsi que l'on procède en pratique. On fait partir la chaîne d'une valeur X_0 quelconque, et on la laisse d'abord évoluer pendant un nombre de pas t_0, suffisant pour que l'on puisse considérer que l'équilibre est atteint. C'est le *préchauffage*. Puis on continue en évaluant la fonction f à des instants séparés par t_1 pas, où t_1 est

suffisamment long pour que les variables aléatoires évaluées en ces instants puissent être considérées comme indépendantes (en général on choisit t_1 plus court que t_0). On observe donc en fait n valeurs extraites de la suite $(f(X_t))$:

$$\left(f(X_{t_0+kt_1})\right), \quad k = 0, \ldots, n-1 .$$

C'est par la moyenne de ces n valeurs, considérées comme des variables aléatoires indépendantes de moyenne $I\!E_p[f]$ et de variance $Var_p[f]$, que l'on estime $I\!E_p[f]$. Pour déterminer la précision de l'estimation, on applique la technique classique des intervalles de confiance. Bien que plusieurs heuristiques aient été proposées (voir le chapitre 6 de Robert [78]), peu de résultats rigoureux permettent une détermination claire de t_0 et t_1. Dans la section suivante, nous étudions la méthode consistant à exécuter en parallèle n copies indépendantes de la chaîne.

4.1.5 Convergence abrupte d'un échantillon

Ce qui suit est issu de [97, 98, 99], où figurent les démonstrations des résultats. Nous supposons toujours que l'objectif est d'obtenir un échantillon de taille n d'une loi de probabilité p, qui est réversible pour une matrice de transition P, irréductible et apériodique. L'idée ici est de simuler en parallèle n copies indépendantes d'une chaîne de Markov de matrice de transition P, partant toutes du même état $i \in E$. On simule alors une "chaîne-échantillon" $\{X^{(t)}\} = \{(X_1^{(t)}, \ldots, X_n^{(t)})\}$ sur E^n, à coordonnées, indépendantes, telle que :

$$(X_1^{(0)}, \ldots, X_n^{(0)}) = (i, \ldots, i) = \tilde{\imath} \in E^n .$$

La probabilité de transition de $(i_1, \ldots, i_n)$ à $(j_1, \ldots, j_n)$ est :

$$p_{i_1 j_1} \ldots p_{i_n j_n} .$$

Notons $\tilde{P}_{\tilde{\imath}}^{(t)}$ la loi de la chaîne-échantillon à l'instant t, par $\tilde{p}$ sa mesure d'équilibre, et par $\|\tilde{P}_{\tilde{\imath}}^{(t)} - \tilde{p}\|$ la distance en variation totale entre les deux.

$$\|\tilde{P}_{\tilde{\imath}}^{(t)} - \tilde{p}\| = \max_{F \subset E^n} |\tilde{P}_{\tilde{\imath}}^{(t)}(F) - \tilde{p}(F)| .$$

Le résultat principal de [97] est le suivant.

Théorème 4.13 *Soit c une constante positive.*

1. *Si*

$$t > \frac{\log n + c}{2 \log 1/\alpha}$$

alors

$$\|\tilde{P}_{\tilde{\imath}}^{(t)} - \tilde{p}\| < \frac{1}{2}\left(-1 + \exp\left(\frac{e^{-c}}{p_i}\right)\right)^{1/2} .$$

2. *Supposons* $w(i) = \sum_{\ell:\,|\alpha_\ell|=\alpha} v_\ell^2(i) > 0$. *Il existe* $n_0 > 0$ *tel que pour* $n > n_0$ *et*

$$t < \frac{\log n - c}{2\log(1/\alpha)} ,$$

alors

$$\|\tilde{P}_{\hat{\imath}}^{(t)} - \tilde{p}\| > 1 - 4\exp\left(\frac{-e^c w^2(i)}{8p_i(1-p_i)}\right) .$$

En d'autres termes, la chaîne-échantillon converge d'une manière très abrupte. Peu avant l'instant $\frac{\log n}{2\log(1/\alpha)}$, elle est encore très loin de l'équilibre. Immédiatement après, elle s'en rapproche exponentiellement vite. Paradoxalement, on peut donc dire que l'équilibre est atteint non pas quand t tend vers l'infini, mais à l'instant (déterministe) $\frac{\log n}{2\log(1/\alpha)}$. Si le but est d'obtenir un échantillon de taille n de la loi p, c'est-à-dire une réalisation de $\tilde{p}$, on doit simuler la chaîne échantillon jusqu'à l'instant de convergence, et il est essentiellement inutile de la simuler beaucoup plus longtemps. Cependant l'instant de convergence s'exprime à l'aide de la valeur α, qui dépend du spectre de P, et est donc inconnue en général. Il existe cependant plusieurs moyens de détecter cet instant de convergence, en utilisant les propriétés statistiques de l'échantillon que l'on est en train de simuler.

Soit f une fonction d'état, définie sur E à valeurs dans $\mathbb{R}$. Rappelons qu'en général, le but de la simulation est l'évaluation de l'espérance d'une telle fonction sous la mesure p. Considérons la moyenne empirique de f à l'instant t :

$$S_i^{(t)}(f) = \frac{1}{n}\sum_{m=1}^{n} f(X_m^{(t)}) .$$

Quand n tend vers l'infini, et pour t supérieur à l'instant de convergence $\frac{\log n}{2\log(1/\alpha)}$, cette moyenne empirique converge vers $\langle f, p\rangle = \sum_{j\in E} f(j)p_j$. On espère que $S_i^{(t)}(f)$ ne sera proche de $\langle f, p\rangle$ qu'à l'instant de convergence, permettant ainsi de le détecter. Il existe plusieurs manières de formaliser cette intuition. Nous commençons par la plus simple, le *temps d'atteinte*.

Définition 4.14 *Supposons* $f(i) < \langle f, p\rangle$. *Le* temps d'atteinte *associé à* i *et* f *est la variable aléatoire* $T_i(f)$ *définie par :*

$$T_i(f) = \inf\{\, t \geq 0 \;:\; S_i^{(t)}(f) \geq \langle f, p\rangle \,\} .$$

Au vu du théorème 4.1.5, il est naturel de s'attendre à ce que $T_i(f)$ soit proche de l'instant de convergence. C'est le cas lorsque $\langle f, P_i^{(t)}\rangle$ est une fonction croissante du temps. Dans la définition 4.14, supposer que $f(i) < \langle f, p\rangle$ n'est pas une restriction importante. Si ce n'est pas le cas, il suffit de remplacer f par $-f$. La même remarque vaut pour la proposition 4.15 : si $\langle f, P_i^{(t)}\rangle$ est une fonction décroissante du temps, alors $\langle -f, P_i^{(t)}\rangle$ est croissante.

Proposition 4.15 *Supposons que* i *et* f *soient tels que :*

- $w_i(f) = \sum_{\ell:\,|\alpha_\ell|=\alpha} \sum_{j\in E} f(j)\frac{\sqrt{p_j}}{\sqrt{p_i}} v_\ell(i)\, v_\ell(j) \neq 0$,
- $\langle f, P_i^{(t)}\rangle$ *est une fonction croissante de* t.

Alors :

$$T_i(f)\,\left(\frac{\log(n)}{2\log(1/\alpha)}\right)^{-1}$$

tend vers 1 *en probabilité quand* n *tend vers l'infini.*

En d'autres termes, $\log(n)/(2T_i(f))$ est un estimateur consistant de $\log(1/\alpha)$. L'hypothèse cruciale de cette proposition est que l'espérance de f sous $P_i^{(t)}$ est une fonction croissante de t. Dans certains cas (chaînes de naissance et mort, systèmes de spin) des résultats de monotonie stochastique permettent de vérifier cette hypothèse a priori. Mais en pratique, on dispose de très peu de renseignements sur P et p, et les hypothèses de la proposition 4.15 ne peuvent pas être validées. Pire encore, l'espérance de f sous p ne peut pas être calculée, et c'est précisément le but de la simulation que de l'évaluer. Le temps de mélange, défini ci-dessous, est une réponse à ce problème.

Définition 4.16 *Soient* i_1 *et* i_2 *deux éléments de* E *tels que :* $f(i_1) < \langle f, p\rangle$ *et* $f(i_2) > \langle f, p\rangle$. *Soient* $S_{i_1}^{(t)}(f)$ *et* $S_{i_2}^{(t)}(f)$ *les moyennes empiriques de* f *calculées sur deux échantillons indépendants de tailles* n_1 *et* n_2, *partant avec toutes leurs coordonnées égales à* i_1 *et* i_2 *respectivement. Le* temps de mélange *associé à* i_1, i_2 *et* f *est la variable aléatoire* $T_{i_1 i_2}(f)$ *définie par :*

$$T_{i_1 i_2}(f) = \inf\{\, t \geq 0 \,:\, S_{i_1}^{(t)}(f) \geq S_{i_2}^{(t)}(f)\,\}\ .$$

L'avantage du temps de mélange est qu'il n'est pas nécessaire de connaître la valeur de $\langle f, p\rangle$ pour le définir. Il vérifie un résultat de convergence analogue au temps d'atteinte.

Proposition 4.17 *Supposons que* i_1 i_2 *et* f *soient tels que :*

- *Au moins une des deux quantités* $w_{i_1}(f)$ *et* $w_{i_2}(f)$ *est non nulle.*
- $\langle f, P_{i_1}^{(t)}\rangle$ *est une fonction croissante de* t.
- $\langle f, P_{i_2}^{(t)}\rangle$ *est une fonction décroissante de* t.

Alors :

$$T_{i_1 i_2}(f)\,\left(\frac{\log(n_1+n_2)}{2\log(1/\alpha)}\right)^{-1}$$

tend vers 1 *en probabilité quand* n_1 *et* n_2 *tendent vers l'infini.*

La méthode proposée se résume donc ainsi. Il faut d'abord choisir une fonction d'état f. En pratique, ce choix s'impose souvent naturellement. Si la valeur cible $\langle f, p\rangle$ est connue, et que le problème est de simuler une réalisation de $\tilde{p}$, il suffit de choisir un état initial i tel que $S_i^{(t)}(f)$ soit monotone en moyenne, puis de simuler la chaîne échantillon jusqu'à ce que $S_i^{(t)}(f)$ atteigne la valeur cible. Dans la plupart des applications cependant, $\langle f, p\rangle$ est inconnu, et l'objectif est précisément d'en calculer une estimation. Il faut alors choisir deux états initiaux i_1 et i_2, et simuler deux chaînes-échantillons indépendantes, une partant de i_1, l'autre de i_2, jusqu'au premier instant où

les deux moyennes empiriques $S_{i_1}^{(t)}(f)$ et $S_{i_2}^{(t)}(f)$ coïncident. Le théorème 4.13 montre que le nombre total de pas sera de l'ordre de $n \log n$, pour obtenir un échantillon de taille n.

La critique principale que l'on peut faire à la méthode proposée, est qu'elle repose sur des résultats asymptotiques, pour une taille d'échantillon tendant vers l'infini et un espace d'états fixé. Or en pratique la taille de l'échantillon, de l'ordre de quelques centaines, est bien inférieure à celle de l'espace d'états. L'expérience montre que la méthode peut quand même donner des résultats satisfaisants dans ce cas. Pour l'illustrer, voici une application à des échantillons de taille 100 d'ensembles stables sur des graphes lignes. Le graphe $G = (S, B)$ a pour ensemble de sommets $S = \{1, \ldots, s\}$ et pour ensemble d'arêtes $B = \{\, \{w, w+1\}\,,\ w = 1, \ldots, s-1 \,\}$. L'espace d'états E est l'ensemble des stables du graphe G, dont le cardinal est le s-ième terme d'une suite de Fibonacci. La mesure p est la loi uniforme sur E et la matrice de transition est celle de la marche aléatoire symétrique sur E définie par l'algorithme donné en exemple dans la section 4.1.1. Deux fonctions d'états sur E, f_1 et f_2, ont été considérées. La première est l'indicatrice des ensembles contenant le premier sommet.

$$f_1(\eta) = 1 \text{ si } \eta(1) = 1\,,\ 0 \text{ sinon}\,.$$

La seconde est la taille d'un ensemble stable.

$$f_2(\eta) = \sum_{x=1}^{v} \eta(x)\ .$$

L'espérance de f_1 et f_2 sous la loi uniforme p se calculent explicitement. La table 1 donne les valeurs exactes de $|E|$ et $\langle f_2, p\rangle$ pour s de 10 à 100. La valeur de $\langle f_1, p\rangle$ converge exponentiellement vite vers $(3-\sqrt{5})/2 = 0.381966$. Des échantillons de taille

s	$\lvert E\rvert$	$\langle f_2, p\rangle$
10	144	2.9167
20	17711	5.6807
30	$2.1783\,10^{6}$	8.4446
40	$2.6791\,10^{8}$	11.2085
50	$3.2951\,10^{10}$	13.9724
60	$4.0527\,10^{12}$	16.7364
70	$4.9845\,10^{14}$	19.5003
80	$6.1306\,10^{16}$	22.2642
90	$7.5401\,10^{18}$	25.0282
100	$9.2737\,10^{20}$	27.7921

TAB. 1: Nombre de stables et taille moyenne d'un stable de loi uniforme pour le graphe ligne à s sommets.

100 ont été engendrés, en utilisant les temps de mélange comme test d'arrêt. Aucune

assurance ne peut être donnée quant aux hypothèses de la proposition 4.17. Dans une première série d'expériences, le test d'arrêt était basé sur f_1. Les 50 premières coordonnées de l'échantillon étaient initialisées à l'ensemble vide, les 50 suivantes, à l'ensemble formé du seul point 1. Donc les valeurs initiales de $S^{(0)}_{\eta_1}(f_1)$ et $S^{(0)}_{\eta_2}(f_1)$ étaient 0 et 1 respectivement. La simulation a été arrêtée au temps de mélange $T_{\eta_1\eta_2}(f_1)$. Pour valider l'échantillon obtenu, l'espérance de f_2 sous p a été estimée par la taille moyenne des 100 ensembles de l'échantillon à cet instant. L'expérience a été répété 100 fois pour chaque $s = 10, 20, \ldots, 100$, la moyenne et l'écart-type étant calculés sur les 100 répétitions. Les résultats figurent dans la table 2. Bien que les temps de mélange aient été très courts, la taille moyenne est correctement estimée (comparer avec la table 1).

s	*temps mélange (moyenne)*	*temps mélange (écart−type)*	*taille moyenne (moyenne)*	*taille moyenne (écart−type)*
10	21.00	8.70	2.92	0.11
20	39.32	18.34	5.63	0.17
30	60.65	25.83	8.36	0.18
40	81.90	32.67	11.13	0.21
50	102.59	39.96	13.86	0.23
60	129.12	49.88	16.61	0.27
70	136.36	58.18	19.25	0.41
80	148.28	56.86	22.04	0.42
90	187.13	67.89	24.86	0.39
100	203.90	77.24	27.55	0.50

TAB. 2: Résultats expérimentaux pour 100 échantillons de 100 stables sur le graphe ligne à s sommets. Temps de mélange et taille moyenne des stables.

Dans la série d'expériences suivante, les rôles de f_1 et f_2 étaient inversés. Le temps de mélange était basé sur f_2, et l'échantillon était utilisé pour estimer la probabilité de contenir le premier sommet : $\langle f_1, p\rangle$. La moitié de l'échantillon était initialisée à l'ensemble vide, l'autre moitié à l'ensemble $\{1, \ldots, s\}$ tout entier. L'expérience était répétée 100 fois, comme précédemment. Les résultats figurent dans la table 3. La probabilité de contenir le premier sommet (valeur théorique $\simeq 0.382$) est correctement estimée, avec un écart-type faible sur les 100 répétitions.

4.2 Recuit simulé

Les algorithmes de recuit simulé sont destinés à des problèmes de minimisation difficiles, au sens où une recherche systématique est rendue impossible par la taille de l'espace, et où la fonction à minimiser a un très grand nombre de minima locaux que l'on souhaite éviter pour trouver le minimum global. Ces algorithmes ont été testés sur tous les problèmes d'optimisation célèbres, et appliqués dans de nombreux contextes. Une importante littérature s'est développée autour de leurs performances et des résultats théoriques permettant de les justifier. Ce qui suit s'inspire de [5, 10, 27].

s	*temps mélange (moyenne)*	*temps mélange (écart−type)*	*fréquence du premier sommet (moyenne)*	*fréquence du premier sommet (écart−type)*
10	22.40	4.74	0.382	0.047
20	46.87	8.89	0.381	0.051
30	74.93	13.64	0.381	0.046
40	103.15	21.10	0.381	0.049
50	128.58	21.75	0.379	0.046
60	160.87	23.66	0.381	0.051
70	188.19	29.03	0.377	0.046
80	216.77	34.82	0.375	0.047
90	251.17	39.18	0.380	0.048
100	284.23	49.59	0.377	0.049

TAB. 3: Résultats expérimentaux pour 100 échantillons de 100 stables sur le graphe ligne à s sommets. Temps de mélange et fréquence des stables contenant le premier sommet.

4.2.1 Mesures de Gibbs

Définition 4.18 *Soit E un ensemble fini et $p = (p_i)_{i\in E}$ une loi de probabilité strictement positive sur E. L'*entropie *de la loi p est la quantité*

$$H(p) \;=\; -\sum_{i\in E} p_i \log p_i \ .$$

L'entropie est une mesure de l'incertitude, du manque d'information lié à la distribution de probabilité p. C'est en vertu d'un principe fondamental en physique que l'on choisit comme modèles, des distributions de probabilité d'entropie maximale. En l'absence de toute information, la distribution de probabilité qui maximise l'entropie est la loi uniforme sur E (incertitude maximale). L'entropie est minimale (et vaut 0) si l'un des p_i vaut 1 et les autres 0 (aucune incertitude).

En fait, en physique, à chaque état i d'un système est associée une énergie $f(i)$. On suppose connue en fonction des conditions extérieures, par estimation ou mesure, l'énergie moyenne du système.

$$I\!E_p[f] \;=\; \sum_{i\in E} f(i)p_i \ .$$

A énergie moyenne constante, la loi de probabilité qui maximise l'entropie est de la forme

$$p_i \;=\; a\exp(bf(i)) \ , \quad \forall i \in E \ ,$$

où a et b sont deux constantes. (Démonstration par le théorème des multiplicateurs de Lagrange). Cette relation entre distributions de probabilité et énergies s'appelle la *loi de Boltzmann*. Les physiciens donnent une signification aux constantes a et b.

Définition 4.19 *On appelle* mesure de Gibbs *associée à la fonction d'énergie f à température $T > 0$ la loi de probabilité $p^T = (p_i^T)_{i \in E}$ telle que :*

$$p_i^T \;=\; \frac{1}{Z(T)} \exp\left(-\frac{1}{T} f(i)\right) \;, \quad \forall i \in E \;,$$

où la constante de normalisation

$$Z(T) \;=\; \sum_{i \in E} \exp\left(-\frac{1}{T} f(i)\right) \;,$$

porte le nom de fonction de partition *du système.*

Les mesures de Gibbs ont la particularité de se concentrer, à basse température, sur les états d'énergie minimale.

Proposition 4.20 *Soit f une fonction d'énergie de E dans $\mathbb{R}$ et p^T la mesure de Gibbs associée à f à température T. Soit $E_{min} = \{i \in E \;:\; f(i) \leq f(j) \;\forall j \in E\}$. Alors :*

$$\lim_{T \to 0^+} p_i^T \;=\; \frac{1}{|E_{min}|} \mathbb{1}_{E_{min}}(i) \;.$$

Au vu de la proposition ci-dessus, il est naturel, pour minimiser la fonction f, de chercher à simuler la loi p^T pour une température T suffisamment basse. Les valeurs de f sur les états obtenus par une telle simulation seront proches de la valeur minimale de f sur E.

La fonction de partition Z de la définition 4.19 est en général impossible à calculer. Heureusement, il est inutile de la calculer pour simuler une mesure de Gibbs par l'algorithme de Metropolis. Supposons, comme c'est en général le cas, que E soit muni d'une structure de graphe connexe régulier. L'algorithme est alors le suivant.

Initialiser X
$n \longleftarrow 0$
Répéter
    $i \longleftarrow X$
    choisir j au hasard parmi les voisins de i
    Si $(f(j) \leq f(i))$
        Alors $X \longleftarrow j$
        Sinon
            Si (Random $< \exp((f(i) - f(j))/T)$) alors $X \longleftarrow j$
            finSi
    finSi
    $n \longleftarrow n+1$
Jusqu'à (arrêt de la simulation)

On peut voir cet algorithme comme une sorte d'algorithme de recherche locale, ou de descente de gradient, pour lequel les pas qui vont vers les basses valeurs de la fonction sont automatiquement effectués. S'il n'y avait que ceux-là, l'algorithme risquerait de rester piégé dans des minima locaux de la fonction f. (Un état i est un minimum local

si $f(i) \leq f(j)$ pour tous les voisins j de i sur le graphe). Pour sortir de ces pièges, on autorise l'algorithme à effectuer des pas dans la mauvaise direction, avec une faible probabilité. Le problème est que à trop basse température, la probabilité des pas vers le haut est très faible, et l'algorithme peut rester piégé très longtemps autour de minima locaux un tant soit peu profonds.

4.2.2 Schémas de température

Le mot *recuit* (annealing) désigne une opération consistant à fondre, puis laisser refroidir lentement un métal pour améliorer ses qualités. L'idée physique est qu'un refroidissement trop brutal peut bloquer le métal dans un état peu favorable. C'est cette même idée qui est à la base du recuit simulé. Pour éviter que l'algorithme ne reste piégé dans des minima locaux, on fait en sorte que la température $T = T(n)$ décroisse lentement en fonction du pas d'itération. L'ensemble E est supposé muni, comme précédemment, d'une structure de graphe $G = (E, A)$. On simule une chaîne de Markov non homogène (X_n) selon l'algorithme de Metropolis, en utilisant comme matrice de sélection celle de la marche aléatoire symétrique sur le graphe G et en changeant la température en fonction du pas d'itération.

$n \longleftarrow 0$
Initialiser X
Répéter
    $T \longleftarrow T(n)$
    $i \longleftarrow X$
    choisir j au hasard parmi les voisins de i
    Si $(f(j) \leq f(i))$
        Alors $X \longleftarrow j$
        Sinon
            Si (Random $< \exp((f(i) - f(j))/T(n))$) alors $X \longleftarrow j$
            finSi
    finSi
    $n \longleftarrow n+1$
Jusqu'à (arrêt de la simulation)

La fonction $T(n)$ s'appelle le *schéma de température* (cooling schedule). Le problème est évidemment celui du choix de $T(n)$. Il s'agit d'assurer que l'algorithme converge effectivement vers l'ensemble E_{min} des minima globaux de la fonction f, mais aussi de contrôler le temps d'atteinte de E_{min}.

Définition 4.21 *Soit (X_n) la chaîne de Markov non homogène produite par l'algorithme ci-dessus. On dit que l'algorithme converge si*

$$\lim_{n\to\infty} I\!P[X_n \in E_{min}] = 1 \, .$$

La convergence dépend de la profondeur des pièges éventuels.

Définition 4.22 *Soit $i \notin E_{min}$. On dit que i communique avec E_{min} à hauteur h, s'il*

existe un chemin dans le graphe, reliant i à E_{min}

$$j_0, j_1, \ldots, j_\ell \ : \ j_0 = i \ , \ j_\ell \in E_{min} \ , \quad \{j_m, j_{m+1}\} \in A \ , \ \forall m \ ,$$

tel que

$$\forall m = 0, \ldots, \ell \ , \quad f(j_m) \leq f(i) + h \ .$$

La hauteur de communication h^* de f est la plus petite hauteur à laquelle tout élément de $E - E_{min}$ communique avec E_{min}.

Le principal résultat théorique de convergence a été démontré par Hajek en 88.

Théorème 4.23 *L'algorithme du recuit simulé converge pour le schéma de température $T(n)$ si et seulement si*

$$\lim_{n\to\infty} T(n) = 0 \qquad et \qquad \sum_{n=1}^{\infty} \exp\left(-\frac{h^*}{T(n)}\right) = +\infty \ .$$

Les schémas de température qui semblent les plus naturels au vu du théorème ci-dessus ont une décroissance logarithmique. Ils sont du type

$$T(n) \ = \ h/\log(n) \ .$$

La condition du théorème de Hajek est vérifiée pour $h > h^*$. Evidemment, l'algorithme sera d'autant plus rapide que h sera plus proche de h^*. Mais dans une vraie application, la valeur exacte de h^* n'est pas connue. D'autre part aucune indication n'est donnée sur le temps d'atteinte du minimum avec une précision donnée. De sorte que le réglage du schéma de température se fait par ajustements expérimentaux.

La fonction log est chère en temps de calcul et varie lentement. Recalculer $T(n)$ à chaque pas de temps serait singulièrement inefficace. Il est préférable de maintenir la température constante sur des paliers de longueur exponentiellement croissante.

$$\forall k \in I\!N^* \ , \ \forall n \in \left]\, e^{(k-1)h} \, , \, e^{kh} \,\right[\ , \quad T(n) \ = \ \frac{1}{k} \ . \tag{4.16}$$

Pour un tel schéma de température, la chaîne de Markov est homogène sur des intervalles de temps de plus en plus longs. Nous allons montrer que pour $h > h^*$, et k assez grand, chacun de ces intervalles de temps est suffisamment long pour que la chaîne atteigne son équilibre, à savoir la mesure de Gibbs pour la température T du palier. Comme ces mesures de Gibbs convergent vers la loi uniforme sur E_{min} (proposition 4.20), cela justifie le fait que l'algorithme converge, ce qu'affirme le théorème de Hajek pour le schéma particulier (4.16). Nous admettrons pour l'instant l'estimation suivante sur les valeurs propres de la matrice de transition de l'algorithme de Metropolis.

Proposition 4.24 *Soient $\lambda_1 = 1 > \lambda_2(T) \geq \cdots \geq \lambda_d(T) > -1$ les valeurs propres de la matrice de transition de la chaîne homogène produite par l'algorithme de Metropolis pour la mesure de Gibbs à température T. Il existe une constante K telle que pour tout $\ell = 2, \ldots, d$, et pour tout $T > 0$*

$$\lambda_\ell^2(T) \ \leq \ 1 - K \exp\left(-\frac{h^*}{T}\right) \ .$$

Dans 4.2.3, nous décrirons le comportement asymptotique des fonctions $\lambda_\ell(T)$ quand T tend vers 0, justifiant ainsi la proposition 4.24.

Nous estimons le temps de convergence à partir de la proposition 4.10. Nous notons p^T la mesure de Gibbs et $p_{ij}^{(m)}$ les probabilités de transition en m pas pour l'algorithme de Metropolis.

$$\sum_{j\in E} p_j^T \left(\frac{p_{ij}^{(m)}}{p_j^T} - 1\right)^2 \;\leq\; \frac{1}{p_i^T}\left(1 - K\exp\left(-\frac{h^*}{T}\right)\right)^m \;. \tag{4.17}$$

Insérons dans la majoration ci-dessus la température $1/k$ et la durée du palier correspondant $m = e^{kh} - e^{(k-1)h}$ (*cf.* (4.16)). Le second membre devient

$$\frac{1}{p_i^T}\left(1 - K\exp\left(-kh^*\right)\right)^{\left(e^{kh}-e^{(k-1)h}\right)} \;\leq\; \exp(-K'\exp(k(h-h^*)))\;.$$

Il tend donc vers 0 quand k tend vers l'infini, pour $h > h^*$.

4.2.3 Spectre à basse température

Les résultats de cette section sont issus d'un travail en collaboration avec Y. Colin de Verdière et Y. Pan [24]. Nous supposerons ici que les valeurs de la fonction f sont multiples d'une même quantité δf et que les différences d'énergies entre sommets voisins valent 0 ou $\pm\delta f$.

$$\forall i \in E\;,\; f(i) \in \delta f\, \mathbb{N} \quad , \quad \forall \{i,j\} \in A\;,\; f(i) - f(j) \in \{-\delta f, 0, +\delta f\}\;. \tag{4.18}$$

Cette hypothèse simplifie beaucoup l'écriture des résultats qui vont suivre sans restreindre vraiment la portée des arguments. Dans les cas pratiques où il est possible de discrétiser la fonction à minimiser de manière à se ramener à cette hypothèse, l'algorithme de recuit simulé s'en trouve considérablement allégé.

```
UnsurT ⟵ 0
Initialiser X
n ⟵ 0
Répéter
    ε ⟵ exp(−δf ∗ UnsurT)
    UnsurT ⟵ UnsurT +1
    Palier ⟵ exp(UnsurT ∗ h)
    Répéter
        i ⟵ X
        choisir j au hasard parmi les voisins de i
        Si (f(j) ≤ f(i))
            Alors X ⟵ j
            Sinon
                Si (Random < ε) alors X ⟵ j
                finSi
```

finSi
$n \longleftarrow n+1$
Jusqu'à ($n \geq$ Palier)
Jusqu'à (arrêt de la simulation)

Dans ce qui suit, comme dans l'algorithme ci-dessus, on note

$$\varepsilon \;=\; \exp\left(-\frac{\delta f}{T}\right) \;.$$

Du fait de l'hypothèse 4.18, la mesure de Gibbs et la matrice de transition de l'algorithme de Metropolis s'expriment en fonction de ε et seront notées respectivement (p_i^ε) et (p_{ij}^ε).

$$p_i^\varepsilon \;=\; \frac{1}{Z(\varepsilon)}\,\varepsilon^{f(i)/\delta f} \;,$$

et

$$\begin{aligned} p_{ij}^\varepsilon &= \frac{1}{r} \quad \text{si } \{i,j\} \in A \;, f(i) \geq f(j) \;, \\ &= \frac{\varepsilon}{r} \quad \text{si } \{i,j\} \in A \;, f(i) < f(j) \;, \\ &= 0 \quad \text{si } \{i,j\} \notin A \;, \end{aligned}$$

où r désigne le degré maximal d'un sommet du graphe G (les coefficients diagonaux sont tels que la somme des coefficients d'une même ligne vaut 1).

Notre but ici est de décrire le comportement asymptotique des valeurs propres de la matrice P^ε à basse température, c'est-à-dire quand ε tend vers 0.

Nous aurons besoin de quelques notations pour décrire le paysage d'énergie sur le graphe. La relation de voisinage est notée $\sim$.

$$i \sim j \iff \{i,j\} \in A \;.$$

On notera $\mathcal{R}$ la relation d'équivalence regroupant les sommets du graphe qui communiquent à énergie constante.

$$\forall i,j \in E \;, \quad i\mathcal{R}j \iff \exists k_1 = i \sim k_2 \sim \cdots \sim k_\ell = j \text{ et } f(k_1) = \cdots = f(k_\ell) \;.$$

Les classes de cette relation seront notées $\alpha, \beta, \ldots$ On notera encore $f(\alpha)$ la valeur commune de la fonction d'énergie sur la classe α. On dira que deux classes α et β *communiquent* s'il existe une arête du graphe qui les relie (la différence $|f(\alpha) - f(\beta)|$ ne peut valoir que δf dans ce cas). Une classe α est dite minimale pour la fonction d'énergie f si elle ne communique pas avec des classes de niveau d'énergie inférieur. Nous définissons par récurrence une suite de fonctions d'énergie déduites de f par remplissage successif des classes minimales.

Définition 4.25 *Notons $f_0 = f$. Soit E_0 l'ensemble des classes minimales de f_0. Pour h entier positif, supposons définie une fonction d'énergie f_h sur E. Soit E_h l'ensemble*

de ses classes minimales. La fonction f_{h+1} est définie sur E par

$$\begin{aligned} f_{h+1}(i) &= f_h(i) + \delta f \quad si \ \exists \alpha \in E_h \ , \ i \in \alpha \ , \\ &= f_h(i) \qquad\quad sinon \ . \end{aligned}$$

La hauteur de communication h^* de f est le premier indice tel que $|E_h| = 1$.

La définition de h^* est bien la même qu'au paragraphe précédent. C'est la hauteur minimale qu'il faut franchir dans le paysage d'énergie pour pouvoir sortir de n'importe quel puits, sauf des minima globaux. Quand ε tend vers 0, parmi les valeurs propres de P^ε, un certain nombre tendent vers des limites strictement inférieures à 1, d'autres tendent vers 1. Ce sont celles-là qui contrôlent la vitesse d'accès à l'équilibre. Si $\lambda_\ell(\varepsilon)$ est une valeur propre de P^ε, on notera $\mu_\ell(\varepsilon) = 1 - \lambda_\ell(\varepsilon)$. La proposition suivante décrit les ordres de grandeur des $\mu_\ell(\varepsilon)$.

Proposition 4.26 *Quand ε tend vers 0, parmi les $\mu_\ell(\varepsilon)$*

- *$|E| - |E_0|$ valeurs convergent vers une limite strictement positive.*
- *Pour tout h entre 1 et h^*, $|E_{h-1}| - |E_h|$ valeurs sont équivalentes, à une constante près, à ε^h.*

On peut interpréter la proposition ci-dessus de la façon suivante. Déterminer les valeurs propres de P^ε ou de $I - P^\varepsilon$ (les $\mu_\ell(\varepsilon)$) à l'ordre ε, revient à changer l'échelle de temps, en se plaçant sur des intervalles d'observation de longueur suffisante pour que des transitions de probabilité ε/r (augmentations d'énergie) aient effectivement lieu. Ceci revient à augmenter de 1 exactement le niveau d'énergie de chacune des classes minimales, c'est-à-dire à remplacer f_0 par f_1. Le nombre de classes minimales ne peut que rester constant ou diminuer. S'il est resté constant, toutes les valeurs propres "petites" sont d'ordre ε^2 au moins. S'il a diminué, la différence entre $|E_0|$ et $|E_1|$ est le nombre de valeurs propres d'ordre ε exactement. Cette description se poursuit jusqu'à ce que toutes les classes minimales aient été fondues en une seule. Cela arrive pour la valeur h^*, la hauteur de communication de f. Toute valeur propre $\mu_\ell(\varepsilon)$ est donc d'ordre ε^h où $h \leq h^*$. Les plus petites d'entre elles sont d'ordre ε^{h^*} exactement, ce qui justifie la proposition 4.24. Cette description est classiquement déduite de la théorie de Freidlin et Wentzell [37]. Nous ne donnerons pas une démonstration complète de la proposition 4.26, mais simplement une idée de justification algébrique. En particulier, nous affirmons sans démonstration que les valeurs propres $\mu_\ell(\varepsilon)$ admettent un équivalent en 0 qui est une puissance entière de ε (on démontre en fait que ce sont des fonctions analytiques de ε).

Démonstration : Les $\mu_\ell(\varepsilon)$ sont valeurs propres de la matrice $I - P$ mais aussi de la matrice $I - DPD^{-1}$, introduite en 4.1.4. La forme quadratique associée à cette matrice (formule (4.5)) s'écrit ici de la façon suivante.

$$\mathcal{Q}(u,u) \ = \ \sum_{\substack{i \sim j \\ f(i) < f(j)}} \frac{1}{r} \left(\sqrt{\varepsilon} u_i - u_j\right)^2 \ + \ \sum_{\substack{i \sim j \\ f(i) = f(j)}} \frac{1}{r} \left(u_i - u_j\right)^2 \ . \tag{4.19}$$

Si $\mu_\ell(\varepsilon)$ est valeur propre de $I - P$, alors il existe un vecteur u de norme 1 tel que $\mathcal{Q}(u,u) = \mu_\ell(\varepsilon)$. Supposons que $\mu_\ell(\varepsilon)$ soit d'ordre $O(\varepsilon)$. Il est facile de voir sur (4.19) que le vecteur u correspondant est tel que ses coordonnées sont d'ordre $\sqrt{\varepsilon}$ en dehors des classes minimales, et constantes à $\sqrt{\varepsilon}$ près sur les classes minimales. L'espace vectoriel des vecteurs constants sur les classes minimales a pour dimension le nombre de classes minimales $|E_0|$. Une base est constituée par les vecteurs $\mathbb{1}_\alpha$, $\alpha \in E_0$. A chacun de ces vecteurs correspond une valeur propre $\mu_\ell(\varepsilon)$ d'ordre ε. Nous utilisons ici la continuité de la décomposition spectrale quand ε tend vers 0. On itère ensuite le raisonnement en cherchant lesquelles parmi les $|E_0|$ valeurs propres petites ainsi trouvées sont d'ordre ε^h. □

La proposition 4.26 ne fournit qu'une indication assez grossière sur le comportement des différentes valeurs propres. Pour évaluer plus précisément la vitesse de convergence de l'algorithme de Metropolis, il faudrait disposer d'équivalents exacts de ces valeurs propres, ainsi que des vecteurs propres associés (utilisation de la formule (4.14)). C'est possible en théorie, mais les calculs sont pour l'instant de complexité supérieure à la recherche systématique du minimum de f, donc sans intérêt pratique.

4.2.4 Implémentation

Finalement, l'algorithme de recuit simulé tel qu'il sera effectivement programmé sera le suivant.

```
UnsurT ⟵ 0
Initialiser X
n ⟵ 0
Répéter
    UnsurT ⟵ UnsurT +1
    Palier ⟵ exp(UnsurT * h)
    Répéter
        i ⟵ X
        choisir j au hasard parmi les voisins de i
        Si (f(j) ≤ f(i))
            Alors X ⟵ j
            Sinon
                Si (Random < exp((f(i) − f(j)) * UnsurT))
                    alors X ⟵ j
                finSi
        finSi
        n ⟵ n+1
    Jusqu'à (n ≥ Palier)
Jusqu'à (arrêt de la simulation)
```

Par rapport à la théorie, le point essentiel est que la hauteur de communication h^* n'est pas connue. Il faut voir le paramètre h comme un des "boutons de réglage" de l'algorithme. Il devra être ajusté expérimentalement. Un autre bouton de réglage est la

structure de graphe choisie sur l'espace. En général plusieurs choix sont possibles et on pourra opter pour une structure de voisinage qui permette une exploration plus rapide de l'espace si cela n'alourdit pas trop l'algorithme. A titre d'exemple, nous traitons ci-dessous un des problèmes les plus célèbres de l'optimisation combinatoire.

Exemple : Le problème du voyageur de commerce [60, 73].

Les éléments de $\{1, \ldots, d\}$ sont des villes dont les distances sont données.

$$\delta(a, b) \geq 0 \ , \quad \forall a, b \in \{1, \ldots, d\} \ .$$

On note f l'application qui à une permutation cyclique σ de $\{1, \ldots, d\}$ associe

$$f(\sigma) \ = \ \sum_{i=1}^{d-1} \delta(\sigma^i(1), \sigma^{i+1}(1)) \ .$$

La quantité $f(\sigma)$ représente la longueur totale du trajet qui visite les d villes dans l'ordre spécifié par σ. Le problème est de trouver une permutation σ qui minimise $f(\sigma)$. Nous supposerons que les d villes sont rangées dans un tableau d'entiers de longueur d : ordre. Les distances $\delta(a, b)$ sont rangées dans un tableau distance. La fonction f pour un ordre donné est calculée comme suit.

```
Fonction f(ordre)
        f ⟵ distance[ordre[1],ordre[d]]
        Pour i de 1 à d − 1
                f ⟵ f+ distance[ordre[i],ordre[i + 1]]
        finPour
        Retourner f
finFonction
```

Nous avons besoin également d'une fonction d'exploration de l'espace d'états, qui choisisse au hasard entre un certain nombre d'ordres "voisins" de l'ordre courant. Le plus simple consiste à échanger deux villes consécutives

```
Fonction permuter(ordre)
        σ ⟵ ordre
        i ⟵ Random({1, . . . , d})
        j ⟵ i + 1 modulo d
        Echanger σ[i] et σ[j]
        Retourner σ
finFonction
```

On obtiendra une exploration plus efficace en permutant 3 ou 4 villes consécutives à la fois.

L'algorithme principal est le suivant.

```
UnsurT ⟵ 0
Pour i de 1 à d
        ordre[i] ⟵ i
```

```
finPour
f1 ⟵ f(ordre)
n ⟵ 0
Répéter
    UnsurT ⟵ UnsurT +1
    Palier ⟵ exp(UnsurT * h)
    Répéter
        ordre2 ⟵ permuter(ordre)
        f2 ⟵ f(ordre2)
        Si (f2 ≤ f1)
            Alors
                ordre⟵ordre2
                f1 ⟵ f2
            Sinon
                proba ⟵ exp((f1 − f2) * UnsurT)
                Si (Random < proba)
                    alors ordre ⟵ ordre2
                    f1 ⟵ f2
                finSi
        finSi
        n ⟵ n+1
    Jusqu'à (n ≥ Palier)
Jusqu'à (arrêt de la simulation)
```

4.3 Algorithmes génétiques

Les algorithmes génétiques sont apparus environ dix ans avant les algorithmes de recuit simulé, à partir d'une autre analogie naturelle. L'idée de base est la même. Il s'agit de coupler une recherche systématique de l'optimum avec une exploration aléatoire de l'espace d'états, que l'on peut qualifier de *générateur de diversité*. Celui-ci doit être suffisamment puissant pour sortir l'algorithme des pièges que constituent les optima locaux de la fonction. Dans le cas des algorithmes génétiques, le générateur de diversité est calqué sur celui de la sélection naturelle qui utilise le croisement et la mutation pour produire de nouveaux chromosomes. L'analogie avec la théorie de l'évolution constitue un attrait psychologique important de ces algorithmes, sur lesquels une littérature extrêmement étendue s'est développée en peu de temps (voir [6, 44, 39, 65, 66]). Les résultats mathématiques rigoureux sont encore rares. C'est R. Cerf [19, 20, 21] qui dans sa thèse a le premier développé une théorie asymptotique fondée sur la théorie des perturbations de Freidlin et Wentzell [37]. Nous suivons sa présentation sans rentrer dans les détails mathématiques, qui sont très techniques.

4.3.1 Version classique

Les algorithmes génétiques s'inspirent des mécanismes de la sélection naturelle, comme le recuit simulé s'inspire de principes physiques. La fonction à optimiser est ici l'*adaptation* et c'est son maximum que l'on recherche. Nous présentons d'abord la version originale, qui est la plus proche de l'analogie naturelle.

La fonction f à maximiser est définie sur $E = \{0,1\}^d$, à valeurs dans $\mathbb{R}^+$. L'idée consiste à faire évoluer un ensemble de taille fixe m d'éléments de E, comme s'il s'agissait d'une population naturelle de chromosomes, la fonction f mesurant l'adaptation d'un chromosome à l'environnement. On définit une chaîne de Markov (X_n), à valeurs dans E^m. La variable X_n est donc un m-uplet $(X_n^1, \ldots, X_n^m)$ d'éléments de E, à savoir un m-uplet de mots binaires de longueur d : les chromosomes. L'algorithme décrivant le passage de X_n à X_{n+1} se décompose en trois étapes.

$$X_n \overset{\text{mutation}}{\longrightarrow} Y_n \overset{\text{croisement}}{\longrightarrow} Z_n \overset{\text{sélection}}{\longrightarrow} X_{n+1} \ .$$

Ces trois étapes sont les suivantes.

- $X_n \longrightarrow Y_n$: *mutation*. La probabilité de mutation $p \in]0,1[$ est fixée au départ. La mutation consiste à décider pour chaque lettre de chaque chromosome de la modifier avec probabilité p, les md choix étant indépendants. La nouvelle population de chromosomes est Y_n.
- $Y_n \longrightarrow Z_n$: *croisement*. La probabilité de croisement q est également fixée. A partir de la population Y_n, $m/2$ couples sont formés (par exemple en appariant les individus consécutifs : m est supposé pair). Pour chacun des couples, on décide indépendamment avec probabilité q si le croisement a lieu. Si c'est le cas, un site de coupure est tiré au hasard uniformément entre 1 et $d-1$, et les segments finaux des deux chromosomes sont échangés. Par exemple :

  $$\begin{array}{l|l} 000\ldots00011 & 01101\ldots001 \\ 100\ldots01110 & 00110\ldots111 \end{array} \quad \text{devient} \quad \begin{array}{l|l} 000\ldots00011 & 00110\ldots111 \\ 100\ldots01110 & 01101\ldots001 \end{array} \ .$$

  Un nouveau couple d'individus est ainsi formé. La population Z_n est constituée des couples d'individus, dont certains ont été croisés (avec probabilité q), d'autres sont restés inchangés (avec probabilité $1-q$).
- $Z_n \longrightarrow X_{n+1}$: *sélection*. Les individus de la population X_{n+1} sont choisis aléatoirement et indépendamment parmi les individus de la population Z_n. La loi de probabilité favorise les individus les mieux adaptés. Le choix classique consiste à prendre des probabilités proportionnelles aux valeurs de la fonction d'adaptation. Supposons que la population Z_n soit constituée des chromosomes $\eta_1, \ldots, \eta_m$. La probabilité de choisir le chromosome η_ℓ est alors

  $$\frac{f(\eta_\ell)}{f(\eta_1) + \cdots + f(\eta_m)} \ .$$

  Mais on peut aussi composer f avec n'importe quelle fonction croissante positive. Le choix le plus cohérent avec le recuit simulé consiste à définir la probabilité de

choix de η_ℓ comme

$$\frac{\exp(kf(\eta_\ell))}{\exp(kf(\eta_1)) + \cdots + \exp(kf(\eta_m))} \ ,$$

où le paramètre k joue le rôle de l'inverse de la température.

Des trois étapes de l'algorithme seule la dernière fait intervenir la fonction f. Les deux autres sont des étapes d'exploration de l'espace d'états. C'est le générateur de diversité. Sans elles, l'étape de sélection convergerait au bout d'un nombre fini de pas vers une population pour laquelle la fonction f serait constante, et en général non optimale.

On définit ainsi une chaîne de Markov sur l'espace d'états E^m, dont on peut vérifier qu'elle est irréductible (tous les états peuvent être atteints) et apériodique. Elle n'admet pas de mesure réversible en général, et les résultats de 4.1.4 ne s'appliquent pas directement. Elle admet cependant une mesure stationnaire unique vers laquelle la suite converge en loi. Il est impossible d'expliciter cette mesure stationnaire. On espère qu'elle se concentre sur les populations de chromosomes à adaptations élevées, voire maximales, donnant ainsi une valeur approchée du maximum de f. Cet espoir est fondé sur l'intuition, l'expérimentation, une solide croyance en l'harmonie de la nature, mais pas sur un résultat mathématique.

L'algorithme génétique qui vient d'être décrit présente un autre inconvénient lié au codage. Si la fonction à traiter n'est pas définie sur $\{0,1\}^d$, il faudra s'y ramener, ce qui ne pose pas a priori de problème sur un ordinateur. Mais les deux premières étapes d'exploration sont très liées à la représentation des chromosomes. Mise à part l'analogie biologique, on conçoit difficilement pourquoi changer une ou plusieurs coordonnées (mutation), ou bien échanger des segments (croisement) serait particulièrement adéquat, si la fonction f ne dépend pas de façon régulière de la représentation binaire des états.

R. Cerf a proposé une vision des algorithmes génétiques à la fois plus générale, plus raisonnable algorithmiquement, et surtout rigoureusement fondée sur le plan mathématique. C'est cette version que nous décrivons dans la section suivante.

4.3.2 Théorie asymptotique

L'ensemble fini E est maintenant quelconque, et f est une fonction de E dans $\mathbb{R}$. On note E_{max} l'ensemble des maxima globaux de f.

$$E_{max} = \{i \in E \ : \ f(i) \geq f(j) \ , \ \forall j \in E\} \ .$$

On souhaite définir une chaîne de Markov (X_n) à valeurs dans l'ensemble E^m des populations de taille m d'éléments de E (les individus). L'objectif est que la loi de X_n se concentre quand n tend vers l'infini sur les populations de E^m_{max}. La logique de la définition est proche de celle du recuit simulé. Nous allons définir tout d'abord une famille de chaînes de Markov homogènes (X^T_n), dépendant d'un paramètre T destiné à tendre vers 0. Nous examinerons d'abord la convergence des mesures stationnaires des chaînes (X^T_n) quand T tend vers 0, vers une mesure concentrée sur les populations

optimales. Ensuite nous définirons un schéma de température $T(n)$ comme dans 4.2.2 et nous examinerons les conditions sous lesquelles la chaîne non homogène $(X_n^{T(n)})$ converge en loi vers une loi ne chargeant que les populations optimales.

Commençons par définir les chaînes homogènes (X_n^T). Le paramètre $T > 0$ (température) est fixé. L'algorithme de définition de (X_n^T) est décomposé en trois étapes, mutation, croisement et sélection, comme précédemment.

- $X_n^T \longrightarrow Y_n^T$: *mutation*. Une matrice de transition $P = (p_{ij})$ sur E et un paramètre $a > 0$ sont fixés. On examine indépendamment chacun des m individus de la population X_n^T. Si i est un de ces individus, on décide
  - de le changer en l'individu j avec probabilité $e^{-a/T} p_{ij}$,
  - ou de ne pas le changer avec probabilité $1 - e^{-a/T}$.

  Une fois les m décisions prises, la nouvelle population est Y_n^T.
- $Y_n^T \longrightarrow Z_n^T$: *croisement*. Une matrice de transition $Q = (q((i_1, i_2), (j_1, j_2)))$ sur $E \times E$ et un paramètre $b > 0$ sont fixés. On examine indépendamment les $m/2$ couples formés d'éléments consécutifs de la population Y_n^T. Si (i_1, i_2) est l'un de ces couples, on décide
  - de le changer en le couple (j_1, j_2) avec probabilité $e^{-b/T} q((i_1, i_2), (j_1, j_2))$,
  - ou de ne pas le changer avec probabilité $1 - e^{-b/T}$.

  Une fois les $m/2$ décisions prises, la nouvelle population est Z_n^T.
- $Z_n^T \longrightarrow X_{n+1}^T$: *sélection*. Un paramètre $c > 0$ est fixé. Les individus de la population X_{n+1}^T sont choisis aléatoirement et indépendamment parmi les individus de la population Z_n^T. Supposons que la population Z_n soit constituée des individus $i_1, \ldots, i_m$, la probabilité de choisir l'individu i_ℓ est

$$\frac{\exp((c/T) f(i_\ell))}{\exp((c/T) f(i_1)) + \cdots + \exp((c/T) f(i_m))} .$$

Lorsque $T = 0$, la mutation et le croisement n'interviennent pas. Seule compte la sélection. Elle consiste alors à tirer au hasard parmi les seuls individus de la population Z_n^T dont l'adaptation est maximale (à comparer avec la proposition 4.20). Au bout d'un seul pas d'itération, l'adaptation de tous les individus est donc la même (sous-optimale en général) et au bout d'un nombre fini de pas tous les individus sont identiques. Ceci est à rapprocher d'un algorithme de recuit simulé pour lequel on fixerait la température à 0 dès le départ : il resterait bloqué dans le premier optimum local rencontré.

La définition de la chaîne (X_n^T) pour $T > 0$ doit être comprise comme une perturbation de ce cas extrême. Cette perturbation est introduite non seulement dans la mutation et le croisement, mais aussi dans la sélection où chaque individu, même sous-optimal a une probabilité non nulle de survivre dans la population suivante. Il est important que les ordres de grandeur logarithmiques des trois perturbations soient les mêmes si on souhaite que les trois aient un effet. C'est la raison d'être des termes en $e^{-a/T}$, $e^{-b/T}$ et $e^{c/T}$ dans les expressions ci-dessus.

Tel qu'il vient d'être défini, l'algorithme comporte un très grand nombre de paramètres qu'il convient d'examiner séparément.

Tout d'abord les matrices de transition P et Q, ou plutôt les mécanismes markoviens sur E et $E \times E$ auxquels elles correspondent. Dans le cas $E = \{0,1\}^d$, nous avons vu dans le paragraphe précédent des exemples de mutations et de croisements. Bien d'autres mécanismes sont envisageables. Il ne semble pas avantageux de compliquer encore une situation qui est déjà assez lourde algorithmiquement. Un mécanisme de mutation le plus simple possible est conseillé. En ce qui concerne le croisement, un moyen très simple de le définir pourrait être le suivant.

$$\forall i_1, i_2, j_1, j_2 \in E \ , \quad q((i_1,i_2),(j_1,j_2)) \ = \ p_{i_1 j_1}\, p_{i_2 j_2} \ .$$

Cela revient à examiner chacun des deux éléments du couple indépendamment et à le changer selon la matrice P. Mais ce ne serait rien d'autre qu'un nouveau passage par l'étape de mutation. En fait, le croisement n'est pas indispensable au bon fonctionnement d'un algorithme génétique, contrairement à ce que croyaient la plupart des praticiens.

Une fois les matrices P et Q choisies, les autres paramètres, en dehors de T, sont la taille de la population m et les intensités de perturbations a, b et c. Ce sont en quelque sorte les boutons de réglage de l'algorithme.

La chaîne de Markov (X_n^T) est irréductible et apériodique. Elle converge donc en loi vers une mesure stationnaire unique p^T (voir par exemple [50]). On souhaite vérifier que lorsque T tend vers 0, cette mesure stationnaire se concentre sur les populations optimales. Contrairement au cas du recuit simulé, ce n'est pas toujours vrai.

Théorème 4.27 *Il existe une taille de population critique m^*, dépendant des autres paramètres de l'algorithme ainsi que de la fonction f, telle que pour $m > m^*$,*

$$\lim_{T \to 0^+} \ p^T(E_{max}^m) \ = \ 1 \ .$$

La démonstration est assez technique et il nous est impossible d'en donner un aperçu significatif dans le cadre restreint de ce cours : voir [19] p. 27 et p. 72. La valeur critique m^* y est décrite en fonction de a, b, c, et f de manière relativement précise, mais malheureusement elle n'est pas calculable en pratique. Une idée importante est à retenir du théorème 4.27, que l'on peut schématiser en disant que c'est parce que la population contient beaucoup d'individus que l'algorithme fonctionne. Quand des choix algorithmiques devront être faits, il vaudra mieux trancher en faveur d'une taille de population plus élevée, quitte à gagner du temps en supprimant le croisement et en simplifiant les mutations. Pour comprendre le rôle crucial de la taille m de la population, il faut imaginer le comportement de la chaîne (X_n^T) à basse température et à l'équilibre.

L'étape de sélection tend à former des populations homogènes d'individus, en général sous-optimaux. La chaîne doit visiter de manière répétée des ensembles de populations proches de tous les ensembles homogènes possibles. Comment procède-t-elle pour aller de l'un à l'autre ? Dans une population homogène apparaissent, par mutation ou croisement, des individus différents, à adaptation peut-être inférieure à celle du reste de la population. Si les intensités de perturbations et la taille de la population sont suffisamment grands, ces individus pionniers seront suffisamment fréquents

et nombreux pour survivre à la sélection, évoluer encore et finalement atteindre de meilleures valeurs de la fonction d'adaptation. Une fois qu'une telle valeur est atteinte par un individu parti en éclaireur, la sélection ramène à lui l'ensemble de la population.

A partir des chaînes de Markov (X_n^T), l'idée de l'algorithme est la même que pour le recuit simulé. Nous allons définir un schéma de température $T(n)$ et simuler la chaîne de Markov non homogène $(X_n^{T(n)})$.

Définition 4.28 *On dit que l'algorithme converge si*

$$\lim_{n\to\infty} I\!P[X_n^{T(n)} \in E_{max}^m] = 1 \ .$$

Le théorème de convergence le plus général démontré par Cerf ([19] p. 87-88) est le suivant. Il donne une condition nécessaire et une condition suffisante de convergence.

Théorème 4.29 *Il existe 4 constantes h_1, h_2, h_1^*, h_2^* dépendant des paramètres de l'algorithme et de la fonction f telles que les conditions* (4.20) *et* (4.21) *ci-dessous soient respectivement nécessaire et suffisante pour que l'algorithme converge.*

$$\sum_{n=1}^{\infty} \exp\left(-\frac{h_1}{T(n)}\right) = +\infty \qquad et \qquad \sum_{n=1}^{\infty} \exp\left(-\frac{h_2}{T(n)}\right) < +\infty \ , \tag{4.20}$$

$$\sum_{n=1}^{\infty} \exp\left(-\frac{h_1^*}{T(n)}\right) = +\infty \qquad et \qquad \sum_{n=1}^{\infty} \exp\left(-\frac{h_2^*}{T(n)}\right) < +\infty \ . \tag{4.21}$$

L'algorithme peut donc converger si $h_1^* < h_2^*$, ce qui n'est vérifié que quand la taille de la population est assez grande. Les constantes critiques h_1, h_2, h_1^*, h_2^* jouent un rôle comparable à la constante h du théorème 4.23. Elles s'expriment à l'aide de la fonction f et des paramètres de l'algorithme, mais ne sont pas calculables en pratique.

Pour les raisons mathématiques et algorithmiques qui ont déjà été discutées en 4.2.2, le schéma de température sera choisi constant sur des paliers de longueurs exponentiellement croissantes.

$$\forall k \in I\!N^* \ , \ \forall n \in \left]\, e^{(k-1)h} \, , \, e^{kh} \,\right[\ , \quad T(n) \ = \ \frac{1}{k} \ . \tag{4.22}$$

La question à résoudre est celle du choix de h, pour assurer que sur le k-ième palier (k assez grand), la chaîne atteigne effectivement sa mesure stationnaire. Au vu du théorème 4.29, on devrait choisir h entre h_1^* et h_2^*. Ces deux quantités sont inconnues mais l'intervalle $[h_1^*, h_2^*]$ est d'autant plus grand que m est grand. D'où l'intérêt, encore une fois, de choisir une taille de population assez grande. Le résultat suivant est un cas particulier qui nous semble correspondre à une implémentation efficace. Il est très proche du théorème 4.23.

Théorème 4.30 *Sous les hypothèses suivantes :*

- *la matrice P est symétrique et irréductible,*
- *il n'y a pas de croisement ($Q = I$ ou bien $b = \infty$),*
- *la taille m de la population est supérieure à la taille critique m^*,*

il existe une hauteur critique h^*, qui est une fonction bornée de m telle que l'algorithme converge si et seulement si

$$\lim_{n\to\infty} T(n) = 0 \qquad et \qquad \sum_{n=1}^{\infty} \exp\left(-\frac{h^*}{T(n)}\right) = +\infty \ .$$

Pour le schéma de température (4.22), il est donc nécessaire et suffisant de choisir $h > h^*$.

4.3.3 Implémentation

Voici résumés quelques conseils algorithmiques que l'on peut tirer de la théorie de Cerf pour implémenter simplement un algorithme génétique sur un problème réel.

1. Choisir sur E une structure de graphe naturelle G de sorte que la marche aléatoire symétrique sur G soit facile à simuler. Ce sont des pas de cette marche aléatoire qui seront simulés dans l'étape de mutation : chaque individu est remplacé s'il y a lieu par un de ses voisins sur le graphe, tiré au hasard.
2. Supprimer l'étape de croisement ($b = \infty$).
3. Choisir le schéma de température (4.22), de sorte que le paramètre T reste constant sur des intervalles dont la longueur croît exponentiellement.

Il reste alors quatre paramètres à régler. Ces paramètres sont :

- *La taille m de la population.* Choisir la valeur la plus élevée possible, compatible avec une vitesse d'exécution raisonnable.
- *Les intensités a et c des perturbations.* A choisir de sorte qu'au début de l'exécution (T grand) la chaîne sorte rapidement du voisinage d'une population homogène.
- *La vitesse de décroissance h du schéma de température* (4.22). Il faut l'ajuster de manière que la durée des paliers ne soit pas trop longue, tout en maintenant une variation suffisante dans les premiers paliers.

Exemple : Le problème du voyageur de commerce.

Considérons l'exemple du voyageur de commerce décrit dans 4.2.4. Nous reprenons les mêmes notations pour la fonction f (c'est $-f$ que l'on maximise) et pour la fonction permuter qui choisit un ordre "voisin" de l'ordre courant. L'objet de base est une population de taille m d'ordres (tableau $d\times m$ d'entiers que nous noterons encore ordres : pour tout i entre 1 et m, ordres[i] contient une permutation des d villes, qui constitue le i-ième élément de la population. Les images par f des éléments de la population seront rangées dans un tableau noté F : $F[i] = f(\text{ordre}[i])$. Afin de limiter le nombre de calculs d'exponentielles, nous utiliserons un tableau EF de même taille qui contient les valeurs de $\exp(-cF[i]/T)$, ainsi qu'une variable somme_EF qui en contient la somme. Pour l'étape de sélection, nous utiliserons des variables auxiliaires correspondant à ordres, F et EF. Elles seront désignées par ordres$_2$, F_2 et EF_2 L'algorithme principal est le suivant.

```
UnsurT ⟵ 0
Pour i de 1 à m
        ordres[i] ⟵ permutation aléatoire de {1, ..., d}
        F[i] ⟵ f(ordre[i])
        EF[i] ⟵ 1
finPour
somme_EF ⟵ m
n ⟵ 0
Répéter
        UnsurT ⟵ UnsurT +1
        Palier ⟵ exp(UnsurT * h)
        Répéter
{Mutations}
                proba_mutation⟵ exp(−a∗UnsurT)
                Pour i de 1 à m
                        Si (Random < proba_mutation)
                                alors
                                        ordres[i]⟵ permuter(ordres[i])
                                        somme_EF ⟵ somme_EF − EF[i]
                                        F[i] ⟵ f(ordre[i])
                                        EF[i] ⟵ exp(−c * F[i]∗UnsurT)
                                        somme_EF ⟵ somme_EF + EF[i]
                        finSi
                finPour
{Sélections}
                Pour i de 1 à m
                        proba_sélection[i]⟵ EF[i]/somme_EF)
                finPour
                Pour i de 1 à m
                        choisir j avec probabilité proba_sélection[j]
                        ordres2[i] ⟵ ordres[j]
                        F2[i] ⟵ F[j]
                        EF2[i] ⟵ EF[j]
                finPour
                somme_EF ⟵ 0
                Pour i de 1 à m
                        ordres[i] ⟵ ordres2[i]
                        F[i] ⟵ F2[i]
                        EF[i] ⟵ EF2[j]
                        somme_EF ⟵ somme_EF + EF[i]
                finPour
                n ⟵ n+1
        Jusqu'à (n ≥ Palier)
Jusqu'à (arrêt de la simulation)
```

4.4 Algorithme MOSES

4.4.1 Définition et convergence

Les algorithmes de recuit simulé et les algorithmes génétiques tels qu'ils ont été présentés jusqu'ici, présentent un inconvénient majeur. Il est impossible en pratique d'utiliser le résultat de convergence théorique pour régler l'algorithme. On est conduit à des réglages expérimentaux des paramètres, qui sont trop nombreux pour qu'on puisse garantir un réglage optimal. Le principe général des algorithmes d'optimisation stochastique est simple (coupler une recherche de l'optimum avec une exploration aléatoire de l'espace d'états) et autorise de multiples variantes. Parmi toutes celles qui ont été proposées, l'algorithme MOSES (Mutation-Or-Selection Evolutionary Strategy), d'O. François [36], présente de nombreux avantages. On peut le voir comme une variante d'un algorithme génétique sans mutation, mais il est beaucoup plus simple, à la fois sur le plan mathématique et sur le plan pratique. Sa caractéristique principale est que les conditions théoriques de convergence ne dépendent pas du paysage d'énergie, et peuvent donc être vérifiées en pratique.

Comme pour un algorithme génétique, il s'agit de travailler sur une population d'individus, parmi lesquels on recherche le maximum d'une fonction f tout en opérant des mutations aléatoires. La fonction f à maximiser est toujours définie sur un ensemble d'états E fini, mais potentiellement très grand. Comme pour les algorithmes précédents, le choix d'une structure de graphe sur E, régulier et connexe, est crucial. Les conditions de convergence des théorèmes 4.31 et 4.32 s'expriment de manière très simple en termes de la distance sur ce graphe. Si x et y sont deux éléments de E, un chemin de longueur n de x à y est une suite $x_0 = x, x_1, \ldots, x_n = y$ de points de E tels que deux points consécutifs sont voisins. La distance de x à y est la plus petite longueur d'un chemin entre x et y. Le *diamètre* du graphe est la plus grande distance entre deux points de E. Nous le noterons D.

Les mutations seront des pas de la marche aléatoire symétrique sur ce graphe. On considère une population de m individus de E. L'algorithme définit donc une chaîne de Markov sur E^m. Une transition consiste à faire muter un certain nombre, aléatoire, d'individus, tandis que les autres sont affectés à l'optimum de la population précédente. Le nombre de mutants est tiré au hasard selon une loi binomiale $\mathcal{B}(m, p)$, dont le paramètre p est destiné à tendre vers 0. Pour conserver l'homogénéité de notation avec les algorithmes précédents, on posera $p = p_T = \exp(-1/T)$, le paramètre T jouant le rôle de la température.

Pour T fixé, on définit donc la chaîne de Markov (X_n^T) à valeurs dans E comme la suite des valeurs successives de la population X dans l'algorithme suivant.

Initialiser $X = X[1, \ldots, m]$
$n \longleftarrow 0$
Répéter
    Déterminer le meilleur élément x^* dans la population X

Tirer un entier M selon la loi $\mathcal{B}(m, e^{-1/T})$
Si $(M > 0)$ alors
  Pour i de 1 à M
    choisir y au hasard parmi les voisins de $X[i]$
    $X[i] \longleftarrow y$
  finPour
finSi
Si $(M < m)$ alors
  Pour i de $M+1$ à m
    $X[i] \longleftarrow x^*$
  finPour
finSi
$n \longleftarrow n+1$
Jusqu'à (arrêt de la simulation)

Comme pour les algorithmes précédents, deux résultats de convergence sont disponibles. Le premier porte sur la mesure stationnaire de la chaîne de Markov homogène (X_n^T), qui doit se concentrer sur les populations optimales. Le second porte sur la convergence de la chaîne non homogène $(X_n^{T(n)})$, pour un schéma de température $T(n)$ fixé. Rappelons la définition de E_{max} :

$$E_{max} = \{i \in E \ : \ f(i) \geq f(j) \ , \ \forall j \in E\} \ .$$

La chaîne de Markov (X_n^T) est irréductible et apériodique. Elle converge donc en loi vers une mesure stationnaire unique p^T.

Théorème 4.31 *Si la taille m de la population est supérieure au diamètre D du graphe, alors :*

$$\lim_{T \to 0^+} \ p^T(E_{max}^m) \ = \ 1 \ .$$

La condition de convergence pour la chaîne non homogène $(X_n^{T(n)})$ est elle aussi très simple.

Théorème 4.32 *Soit D le diamètre du graphe. Si $m > D$ et si $T(n)$ vérifie :*

$$\sum_{n=1}^{\infty} \exp\left(-\frac{D}{T(n)}\right) = +\infty \ ,$$

alors :

$$\lim_{n \to \infty} \ I\!P[X_n^{T(n)} \in E_{max}^m] = 1 \ .$$

La comparaison de ce résultat avec les théorèmes 4.23 et 4.30 montre bien l'avantage de l'algorithme MOSES, pour lequel les conditions de convergence ne dépendent que de la géométrie du graphe, et pas des variations de la fonction f, qui sont inconnues.

4.4.2 Implémentation

L'implémentation de l'algorithme MOSES est elle aussi très simple. Contrairement aux algorithmes génétiques, elle ne nécessite pas d'évaluer la fonction sur tous les

éléments de la population à chaque étape, mais seulement sur ceux qui ont été modifiés. De même, il n'est pas nécessaire de conserver en mémoire la population complète des m individus, mais seulement ceux qui ont été modifiés par le tirage binomial. Dans la mesure où la probabilité de modification p_T tend vers 0, cela entraîne un gain important en place mémoire. Comme précédemment, on est amené à choisir un schéma de température $T(n)$ qui décroît par paliers.

$$\forall k \in I\!N^* \ , \ \forall n \in] \, e^{(k-1)D} \, , \, e^{kD} \, [\ , \quad T(n) \ = \ \frac{1}{k} \ . \tag{4.23}$$

La différence est que le paramètre de règlage de la longueur des paliers, qui est le diamètre du graphe, est déterminé par la structure de graphe que l'on a choisie. Nous illustrons l'implémentation sur le même problème que précédemment.

Exemple : Le problème du voyageur de commerce

Les notations sont celles de la section 4.2.4. C'est la fonction permuter qui choisit un ordre "voisin" de l'ordre courant et qui détermine donc la structure de graphe et la valeur du diamètre D. Pour le cas particulier où la fonction échange deux villes consécutives, on peut vérifier qu'il faut $D = d(d-1)/2$ appels au maximum pour passer d'un ordre quelconque à un autre. Mais ce n'est évidemment pas un choix optimal. Permuter circulairement 3 ou 4 villes, conduira à une diminution de la valeur de D et à un temps d'exécution beaucoup plus faible. L'algorithme utilise une fonction de simulation binomiale qui retourne une variable aléatoire suivant la loi $\mathcal{B}(m, p)$. La probabilité p étant faible sur la plus grande partie du temps d'exécution, on aura intérêt à la programmer suivant la méthode d'inversion (section 2.3.2). Cette fonction détermine la taille de la population des mutants, qui sont seuls concernés par la fonction permuter. La population courante est stockée dans le tableau de taille m ordres dont les premiers éléments sont des mutants, les autres étant tous égaux à l'optimum de la population précédente (Meilleur). Quand le nombre de mutants est faible, une grande partie de la population est composée d'éléments identiques. L'utilisation d'un tableau de taille fixe entraîne donc une perte de place mémoire et beaucoup d'opérations inutiles. Une liste, ou une chaîne de pointeurs serait une structure de données mieux adaptée qu'un tableau.

```
(* Initialisations *)
Fmeilleur ⟵ d * max{δ(a,b)}
Pour i de 1 à m
      ordres[i] ⟵ permutation aléatoire de {1,...,d}
      F ⟵ f(ordres[i])
      Si (F < Fmeilleur) alors
            Fmeilleur ⟵ F
            Meilleur ⟵ ordres[i]
      finSi
finPour
UnsurT ⟵ 0
n ⟵ 0
(* Boucle principale *)
Répéter
      UnsurT ⟵ UnsurT +1
      p ⟵ exp(−UnsurT)
      Palier ⟵ exp(UnsurT * D)
      Répéter
            Répéter
                  n ⟵ n+1
                  Nombre_Mutants ⟵ binomiale(m,p)
            Jusqu'à (Nombre_Mutants > 0)
            Si (Nombre_Mutants < m) alors
                  Pour i de Nombre_Mutants+1 à m
                        ordres[i] ⟵ Meilleur
                  finPour
            finSi
            Pour i de 1 à Nombre_Mutants
                  ordres[i] ⟵ permuter(ordres[i])
                  F ⟵ f(ordres[i])
                  Si (F< Fmeilleur) alors
                        Fmeilleur ⟵ F
                        Meilleur ⟵ ordres[i]
                  finSi
            finPour
      Jusqu'à (n ≥ Palier)
Jusqu'à (arrêt de la simulation)
```

4.5 Exercices

Exercice 4.1 Ecrire un algorithme de simulation approchée par chaîne de Markov pour la loi uniforme sur :

1. L'ensemble des vecteurs $(k_1, \ldots, k_d)$, à coefficients entiers positifs ou nuls, tels que $k_1 + \cdots + k_d = n$ (les entiers d et n sont fixés).
2. La sphère unité de $\mathbb{R}^d$.
3. L'ensemble des sous ensembles à n éléments d'un ensemble à d éléments.
4. L'ensemble des tables de contingence de taille d, de marges fixées. Une table de contingence est une matrice $d \times d$ à coefficients entiers positifs ou nuls, où $L = A\mathbb{1}$ (sommes par lignes) et $C = {}^tA\mathbb{1}$ (sommes par colonnes) sont des vecteurs fixés (tels que ${}^t\mathbb{1}L = {}^t\mathbb{1}C$).
5. L'ensemble des arbres à d sommets.
6. L'ensemble des graphes connexes à d sommets.

Exercice 4.2 Ecrire un algorithme de Metropolis pour la simulation approchée des lois de probabilité suivantes.

1. La loi sur l'ensemble des vecteurs d'entiers $(k_1, \ldots, k_d)$ de somme n qui est telle que la probabilité d'un vecteur soit proportionnelle à sa première coordonnée.
2. La loi sur la sphère unité de $\mathbb{R}^d$ dont la densité est proportionnelle au carré de la première coordonnée.
3. La loi sur l'ensemble des sous-ensembles à n éléments de $\{1, \ldots, d\}$, telle que la probabilité d'un sous-ensemble soit proportionnelle à la somme de ses éléments.
4. La loi sur l'ensemble des tables de contingence de taille d, de marges fixées, telle que la probabilité d'une table de contingence soit proportionnelle à la somme des éléments de sa diagonale principale.
5. La loi sur l'ensemble des arbres à d sommets, telle que la probabilité d'un arbre soit proportionnelle à son diamètre (nombre maximum d'arêtes dans un chemin minimal joignant deux sommets).
6. La loi sur l'ensemble des graphes connexes à d sommets, telle que la probabilité d'un graphe connexe soit proportionnelle à son nombre d'arêtes.

Exercice 4.3 Le but de l'exercice est d'étudier la vitesse de convergence de marches aléatoires sur l'hypercube $E = \{0,1\}^d$. C'est un des rares cas où l'on sache diagonaliser explicitement la matrice de transition, et donc évaluer précisément la vitesse de convergence.

Les éléments de E seront notés $\eta, \zeta, \xi \ldots$

$$\eta = (\eta(i)) \ , \quad \eta(i) \in \{0,1\} \quad \forall i = 1, \ldots, d \ .$$

L'ensemble E est naturellement muni d'une structure de graphe $G = (E, A)$, pour laquelle deux sommets η et ζ sont voisins si et seulement si ils diffèrent en une coordonnée et une seule.

$$\{\eta, \zeta\} \in A \iff \sum_{i=1}^{d} |\eta(i) - \zeta(i)| = 1 \ .$$

Soit $\beta \in [0,1]$ un réel fixé. On définit la matrice de transition $P = (p_{\eta\zeta})$ par

$$\begin{aligned} p_{\eta\zeta} &= \beta/d \quad \text{si } \eta \text{ et } \zeta \text{ sont voisins ,} \\ &= 1-\beta \quad \text{si } \eta = \zeta \ , \\ &= 0 \quad \text{dans tous les autres cas .} \end{aligned}$$

Pour $\beta = 0$, la matrice P est la matrice identité, et la chaîne correspondante est constante. Pour $\beta = 1$ la matrice P est celle de la marche aléatoire symétrique sur l'hypercube G, qui à chaque pas modifie une coordonnée choisie au hasard. Cette marche aléatoire est périodique de période 2. La probabilité de revenir à l'état de départ au bout de m pas, $p_{\eta\eta}^{(m)}$, est nulle si m est impair. Pour éviter ces deux cas particuliers, nous supposerons que $0 < \beta < 1$.

1. Soit ξ un élément quelconque de E. Le caractère χ_ξ est la fonction de E dans $\{-1,1\}$ qui à η associe
$$\chi_\xi(\eta) \ = \ (-1)^{{}^t\xi\,\eta} \ .$$
Montrer que la famille $(\chi_\xi)_{\xi\in E}$ est une base orthogonale. Quelle est la norme de χ_ξ ?
2. Montrer que le caractère χ_ξ est associé à la valeur propre
$$\lambda_\xi \ = \ 1 - \frac{2\beta}{d}\sum_{i=1}^{d}\xi(i) \ .$$
3. En déduire que les valeurs propres de P sont
$$1 \ , \ 1-\frac{2\beta}{d} \ , \ 1-\frac{4\beta}{d} \ , \ldots \ , \ 1-2\beta \ ,$$
la valeur propre $1-\dfrac{2k\beta}{d}$ étant de multiplicité $\binom{d}{k}$.
4. Déterminer, en fonction de β, la valeur de $\alpha \ = \ \max\{|\lambda_\xi| \ , \ \xi \in E \ , \ \lambda_\xi \neq 1\}$.

Nous supposerons désormais $\beta \leq d/(1+d)$. Comme base orthonormée de vecteurs propres, on choisira $(\phi_\xi)_{\xi\in E}$, où
$$\phi_\xi(\eta) \ = \ 2^{-d/2}\chi_\xi(\eta) \ .$$
Notre but est d'évaluer le nombre de pas nécessaires pour qu'une chaîne de Markov de matrice de transition P atteigne l'équilibre avec une précision fixée.
5. En utilisant la relation (4.13), montrer l'estimation suivante.
$$\sum_{\zeta\in E} 2^{-d}\left(\frac{p_{\eta\zeta}^{(m)}}{2^{-d}}-1\right)^2 \ < \ \varepsilon \quad \text{si} \quad m > d^2\frac{\log(2)}{4\beta} - d\frac{1}{4\beta}\log(\varepsilon) \ .$$

6. En utilisant la relation (4.9), montrer l'estimation suivante.

$$\sum_{\zeta\in E} 2^{-d}\left(\frac{p_{\eta\zeta}^{(m)}}{2^{-d}}-1\right)^2 \;<\; \varepsilon \quad \text{si} \quad m > \frac{d\log(d)}{4\beta} - \frac{d}{4\beta}\log(\log(1+\varepsilon))\,.$$

Donc l'équilibre est atteint pour un nombre de pas de l'ordre de $d\log(d)/(4\beta)$, c'est-à-dire très inférieur à la taille de l'espace (2^d).

7. On considère la chaîne de Markov sur $E = \{0,1\}^d$ dont l'algorithme de simulation est le suivant.

```
Initialiser η
t ⟵ 0
Répéter
    choisir i, de loi uniforme sur {1,...,d}
    η(i) ⟵ 1 − η(i)
    Si (Random < 0.5) alors
        choisir j, de loi uniforme sur {1,...,d}
        η(j) ⟵ 1 − η(j)
    finSi
    t ⟵ t + 1
Jusqu'à (arrêt de la simulation)
```

Utiliser les résultats des questions précédentes pour calculer les valeurs propres de la matrice de transition, puis évaluer la vitesse de convergence de cette chaîne de Markov.

Exercice 4.4 Pour chacun des problèmes d'optimisation suivants, écrire un algorithme de recuit simulé, un algorithme génétique et un algorithmes MOSES.

1. *Affectation de postes.* Une matrice de coûts indicée par $\{1,\dots,d\}$ est donnée.

$$\delta(i,j) \geq 0\,, \quad \forall i,j \in \{1,\dots,d\}\,.$$

Le réel $\delta(i,j)$ est le coût de formation de l'ouvrier i au poste de travail j. On note f l'application qui à une permutation σ de $\{1,\dots,d\}$ associe

$$f(\sigma) \;=\; \sum_{i=1}^{d} \delta(i,\sigma(i))\,.$$

La quantité $f(\sigma)$ représente le coût total de formation des d ouvriers pour l'affectation σ. Le problème est de trouver une permutation σ qui minimise $f(\sigma)$.

2. *Partitionnement d'un graphe.* Un graphe $G = (S,A)$ est donné. A une partition $\{S_1,S_2\}$ de S en deux sous-ensembles, on associe la quantité

$$f(S_1,S_2) \;=\; |A_{12}| + \lambda\Big|\,|S_1| - |S_2|\,\Big|\,,$$

où $|A_{12}|$ est le nombre total d'arêtes entre S_1 et S_2 et $\lambda > 0$ est un facteur donné de pondération. Le problème consiste à trouver une partition $\{S_1,S_2\}$ qui minimise $f(S_1,S_2)$.

3. *Coloration d'un graphe.* Un graphe $G = (S, A)$ est donné. Une coloration du graphe G est une application c de S dans $\{1, \ldots, |S|\}$ telle que

$$\forall i, j \in S\ , \quad \{i, j\} \in A \Longrightarrow c(i) \neq c(j)\ .$$

Le nombre de couleurs est

$$f(c) \ = \ |c(S)|\ .$$

Le problème consiste à trouver une coloration utilisant le plus petit nombre de couleurs possible.

4. *Modèle de Ising.* Un graphe $G = (S, A)$ est donné. On considère l'ensemble $E = \{0, 1\}^S$, et l'application f qui à une configuration $\eta \in E$ associe le nombre d'arêtes du graphe joignant des sommets d'états opposés.

$$f(\eta) \ = \ \frac{1}{2}\left(|A| - \sum_{\{i,j\}\in A} (-1)^{\eta(i)+\eta(j)}\right)\ .$$

Le problème consiste à trouver une configuration η qui minimise $f(\eta)$.

Références

[1] D.J. Aldous. On the Markov chain simulation method for uniform combinatorial distributions and simulated annealing. *Probab. Eng. and Inf. Sciences*, 1 :33–46, 1987.

[2] D.J. Aldous. Approximate counting via Markov chains. *Statistical Science*, 8(1) :16–19, 1993.

[3] I. Aleksander and H.B. Morton. *An introduction to neural computing.* Chapman and Hall, London, 1990.

[4] A. Antoniadis and G. Oppenheim, editors. *Wavelets and statistics*, number 103 in L.N. in Stat. Springer-Verlag, New York, 1995.

[5] R. Azencott. Simulated annealing. *Séminaire Bourbaki*, 697 :161–175, 1988.

[6] T. Bäck. *Evolutionary algorithms in theory and practice.* Oxford University Press, Oxford, 1996.

[7] A.T. Barucha-Reid. *Elements of the theory of Markov processes and their Applications.* McGraw-Hill, London, 1960.

[8] A. Benveniste, M. Métivier, and P. Priouret. *Algorithmes adaptatifs et approximations stochastiques.* Masson, Paris, 1987.

[9] M.A. Berger. *An introduction to probability and stochastic processes.* Springer-Verlag, New York, 1993.

[10] D. Bertsimas and J. Tsitsiklis. Simulated annealing. *Statistical Science*, 8 :10–15, 1993.

[11] N. Biggs. *Algebraic Graph Theory.* Cambridge University Press, 1973.

[12] N. Bouleau. *Probabilités de l'ingénieur, variables aléatoires et simulation.* Hermann, Paris, 1985.

[13] N. Bouleau. *Processus stochastiques et applications.* Hermann, Paris, 1988.

[14] N. Bouleau and D. Lépingle. *Numerical methods for stochastic processes.* Wiley, New York, 1994.

[15] L. Breiman. *Probability.* Addison-Wesley, Reading, 1968.

[16] P. Bremaud. *Markov chains, Gibbs fields, Monte-Carlo simulation and queues.* Spinger-Verlag, New York, 1999.

[17] R. Cairoli and R.C. Dalang. *Sequential stochastic optimization.* Wiley, New-York, 1996.

[18] O. Catoni. Rough large deviation estimates for simulated annealing : Application to exponential schedules. *Ann. Probab.*, 20(3) :1109–1146, 1992.

[19] R. Cerf. *Une théorie asymptotique des algorithmes génétiques.* PhD thesis, Université Montpellier II, 1994.

[20] R. Cerf. The dynamics of mutation-selection algorithms with large population sizes. *Ann. Inst. H. Poincaré, Probab. Stat.*, 32(4) :455–508, 1996.

[21] R. Cerf. A new genetic algorithm. *Ann. Appl. Probab.*, 6(3) :778–817, 1996.

[22] K.L. Chung. *Markov chains with stationary transition probabilities.* Springer-Verlag, New York, 1960.

[23] E. Çinlar. *Introduction to stochastic processes.* Prentice Hall, New York, 1975.

[24] Y. Colin de Verdière, Y. Pan, and B. Ycart. Singular limits of Schrödinger operators and Markov processes. *J. of Operator Theory*, 41 :151–173, 1999.

[25] G.M. Del Corso. Randomization and the parallel solution of linear algebra problems. *J. Comput. Math. Appl.*, 30(11) :59–72, 1995.

[26] L. Devroye. *Non-uniform random variate generation.* Springer-Verlag, New York, 1986.

[27] P. Diaconis and L. Saloff-Coste. What do we know about the Metropolis algorithm ? *J. Comp. Syst. Sci.*, 55(1) :20–36, 1998.

[28] P. Doyle and J. Snell. *Random walks and electric networks.* M. A. A., Washington, 1984.

[29] E.J. Dudewicz and T.G. Ralley. *The handbook of random number generation and testing with TESTRAND computer code.* American Sciences Press Inc., Columbus., 1981.

[30] M. Duflo. *Méthodes récursives aléatoires.* Masson, Paris, 1990.

[31] M. Duflo. *Algorithmes stochastiques.* Number 23 in Mathématiques et applications. Springer-Verlag, Berlin, 1996.

[32] R.L. Eubank. *Spline smoothing and nonparametric regression.* Marcel Dekker, New York, 1988.

[33] W. Feller. *An introduction to probability theory and its applications*, volume I. Wiley, London, 1968.

[34] W. Feller. *An introduction to probability theory and its applications*, volume II. Wiley, London, 1971.

[35] G.S. Fishman. *Monte-Carlo concepts algorithms and applications.* Springer-Verlag, New York, 1996.

[36] O. François. An evolutionary strategy for global minimization and its Markov chain analysis. *IEEE trans. on Evolutionary Computation*, 2(3) :77–90, 1998.

[37] M.I. Freidlin and A.D. Wentzell. *Random perturbations of dynamical systems.* Springer-Verlag, New York, 1984.

[38] J.E. Gentle. *Random number generation and Monte-Carlo methods.* Springer-Verlag, New York, 1998.

[39] D. Goldberg. *Genetic algorithms in search, optimization and machine learning.* Addison-Wesley, New York, 1989.

[40] C. Graham, T.G. Kurtz, S. Méléard, P.E. Protter, M. Pulvirenti, and D. Talay. *Probabilistic Models for Nonlinear Partial Differential Equations.* Number 1627 in L. N. in Math. Springer-Verlag, New York, 1996.

[41] R.L. Graham, D.E. Knuth, and O. Patashnik. *Concrete mathematics : a foundation for computer science.* Addison-Wesley, Reading, 1989.

[42] J.M. Hammersley and D.C. Handscomb. *Monte-Carlo methods.* Methuen, London, 1964.

[43] M. Hofri. *Probabilistic analysis of algorithms.* Springer-Verlag, New York, 1987.

[44] J.H. Holland. *Adaptation in natural and artificial systems.* The University of Michigan Press, Ann Arbor, 1975.

[45] I. Karatzas and S.E. Shreve. *Brownian motion and stochastic calculus.* Springer-Verlag, New York, 1991.

[46] I. Karatzas and S.E. Shreve. *Methods of mathematical finance.* Springer-Verlag, New York, 1998.

[47] S. Karlin. *A first course in stochastic processes.* Academic Press, San Diego, 1966.

[48] S. Karlin and H.M. Taylor. *A second course in stochastic processes.* Academic Press, San Diego, 1981.

[49] F.P. Kelly. *Reversibility and Stochastic Networks.* Wiley, London, 1979.

[50] J.G. Kemeny and J.L. Snell. *Finite Markov chains.* Van Nostrand, Princeton, 1960.

[51] W.J. Kennedy and J.E. Gentle. *Statistical computing.* Marcel Dekker, Inc., New York, 1980.

[52] J.P.C. Kleijnen. *Statistical techniques in simulation, Part I.* Marcel Dekker, Inc., New York, 1974.

[53] J.P.C. Kleijnen and W. Van Groenendaal. *Simulation, a statistical perspective.* Wiley, New York, 1992.

[54] P.E. Kloeden and E. Platen. *Numerical solution of stochastic differential equations.* Springer-Verlag, New York, 1992.

[55] P.E. Kloeden, E. Platen, and H. Schurz. *Numerical solution of SDE through computer experiment.* Springer-Verlag, New York, 1997.

[56] D.E. Knuth. *The art of computer programming*, volume 2, seminumerical algorithms. Addison-Wesley, Reading, 1981.

[57] H.J. Kushner and G.G. Yin. *Approximation algorithms and applications.* Springer-Verlag, New York, 1997.

[58] D. Lamberton and B. Lapeyre. *Introduction au calcul stochastique appliqué à la finance.* Ellipses, Paris, 1991.

[59] B. Lapeyre, E. Pardoux, and R. Sentis. *Méthodes de Monte-Carlo pour les équations de transport et de diffusion.* Number 29 in Mathématiques et applications. Springer-Verlag, Berlin, 1997.

[60] E.L. Lawler, J.K. Lensra, A.H.G. Rinnooy Kan, and D.B. Shmoys. *The traveling salesman problem.* Wiley, New York, 1987.

[61] G. Marsaglia and A. Zaman. Toward a universal random number generator. *Stat. Prob. Lett.*, 8 :35–39, 1990.

[62] G. Marsaglia and A. Zaman. A new class of random number generators. *Ann. Appl. Probab.*, 1 :462–480, 1991.

[63] T. Masters. *Practical neural network recipes in C++.* Academic Press, Boston, 1993.

[64] G.J. McLachlan and T. Krishnan. *The EM algorithm and extensions.* Wiley, New York, 1997.

[65] Z. Michalewicz. *Genetic algorithms + Data structures = Evolution programs, 3rd ed.* Springer-Verlag, New-York, 1996.

[66] M. Mitchell. *An introduction to genetic algorithms.* MIT Press, Cambridge, MA, 1996.

[67] B.J.T. Morgan. *Elements of simulation.* Chapman and Hall, London, 1984.

[68] R. Motwani and P. Raghavan. *Randomized algorithms.* Cambridge University Press, 1995.

[69] M. Musiela and M. Rutkowski. *Martingale methods in financial modelling.* Springer-Verlag, Berlin, 1997.

[70] M.F. Neuts. *Matrix-geometric solutions in stochastic models.* The John Hopkins University Press, London, 1981.

[71] M.F. Neuts. *Algorithmic Probability : a collection of problems.* Chapman and Hall, London, 1995.

[72] P. Propp and D.B. Wilson. Exact sampling with coupled markov chains and applications to statistical mechanics. *Rand. Struct. Algo.*, 9(2) :223–252, 1996.

[73] G. Reinelt. *The traveling salesman : computational solutions for TSP applications.* Number 840 in L. N. in Computer Science. Springer-Verlag, New York, 1994.

[74] B.D. Ripley. *Stochastic simulation.* Wiley, New York, 1987.

[75] B.D. Ripley. Thoughts on pseudorandom numbers. *J. Comput. Appl. Math.*, 31 :153–163, 1990.

[76] B.D. Ripley. *Neural networks and pattern recognition.* Cambridge University Press, 1996.

[77] C.P. Robert. *L'Analyse Statistique Bayésienne.* Economica, Paris, 1992.

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

[79] C.P. Robert and G. Casella. *Monte-Carlo statistical methods.* Springer-Verlag, New York, 1999.

[80] T.G. Robertazzi. *Computer networks and systems : queuing theory and performance evaluation.* Springer-Verlag, New York, 1990.

[81] R.Y. Rubinstein. *Simulation and the Monte-Carlo method.* Wiley, New York, 1981.

[82] R.Y. Rubinstein. *Monte-Carlo optimization, simulation and sensitivity of queuing networks.* Wiley, New York, 1986.

[83] R.Y. Rubinstein and B. Melamed. *Efficient simulation and Monte-Carlo methods.* Wiley, New York, 1997.

[84] R.Y. Rubinstein and A. Shapiro. *Discrete event systems : sensitivity analysis and stochastic optimization by the score function method.* Wiley, New York, 1993.

[85] L. Saloff-Coste. Lectures on finite Markov chains. In P. Bernard, editor, *Ecole d'été de probabilité de Saint-Flour XXVI*, number 1665 in L.N. in Math., pages 301–413. Springer-Verlag, New York, 1997.

[86] R. Serforzo. *Introduction to stochastic networks.* Springer-Verlag, New York, 1999.

[87] R. Shonkwiler and E. Van-Vleck. Parallel speed-up of Monte-Carlo methods for global optimization. *J. Complexity*, 10(1) :64–95, 1994.

[88] A. Sinclair. *Algorithms for random generation and counting : a Markov chain approach.* Birkhäuser, Boston, 1993.

[89] J.L. Snell. *Introduction to probability.* Random House, New York, 1988.

[90] W.J. Stewart. *Introduction to the numerical solution of Markov chains.* Princeton University Press, 1995.

[91] D. Talay. Simulation and numerical analysis of stochastic differential systems : a review. In P. Krée and W. Wedig, editors, *Probabilistic Methods in Applied Physics*, volume 451 of *L. N. in Physics*, chapter 3, pages 54–96. Springer-Verlag, New York, 1995.

[92] K.S. Trivedi. *Probability and Statistics with Reliability, Queuing and Computer Science Applications.* Prentice-Hall, New York, 1982.

[93] A. Trouvé. *Parallélisation massive du recuit simulé.* PhD thesis, Université Paris XI, 1993.

[94] K. Watkins. *Discrete event simulation in C.* McGraw-Hill, London, 1993.

[95] G. Winkler. *Image analysis, random fields and dynamic Monte-Carlo methods.* Springer-Verlag, Berlin, 1995.

[96] R.W. Wolff. *Stochastic Modelling and the Theory of Queues.* Prentice-Hall, Englewood Cliff, 1989.

[97] B. Ycart. Cutoff for samples of Markov chains. *ESAIM Probability-Statistics*, 3 :89–107, 1999.

[98] B. Ycart. Cutoff for Markov Chains : some examples and applications. FIESTA'98 summer school on stochastic and dynamical systems, à paraître, 2000.

[99] B. Ycart. Stopping tests for Monte-Carlo Markov chain methods. Meth. and Comp. in Appl. Probab., à paraître, 2000.