

Algorithme d'Hasting Metropolis - Probl me du voyageur de commerce

PEROTTINO Tony, VAILLANT Corentin,
LE BER Tom

10 mai 2025

Table des matières

1	Introduction	3
1.1	Informations générales et description historique	3
1.2	Pourquoi HM permet-il de résoudre le problème du voyageur de commerce?	4
1.3	Objectif de ce rapport	5
2	Chapitres de Markov à l'état fini	6
2.1	Définitions fondamentales	6
2.2	Propriétés fondamentales	8
2.2.1	Matrice de Transition	8
2.2.2	Représentation sous forme de Graphe Orienté	9
2.2.3	Matrice de transition en k -pas	10
2.3	Classes d'équivalence	12
2.3.1	Classification des états	12
2.3.2	Classes de communication	13
2.4	Probabilités invariantes et théorème ergodique	16
2.4.1	probabilités invariantes	16
2.4.2	Théorème ergodique	18
2.5	Comportement en temps long	19
3	Fonctionnement de l'Algorithme de Hasting-Metropolis	21
3.1	Qu'est-ce qu'une méthode dite MCMC?	21
3.2	Définitions	22
3.3	Description des étapes de l'algorithme	22
3.4	Preuve de l'algorithme	22
3.5	Exemples d'applications simples de HM	22
3.6	Application au cas du voyageur de commerce	24
3.7	Comparaison avec d'autres méthodes d'échantillonnage	25
4	Conclusion	26
5	Annexe	27
5.1	Code source	27
5.1.1	Code utilitaires	27
5.1.2	Implémentation de l'algorithme de Hastings-Metropolis	28
5.1.3	Applications du code d'Hasting Metropolis	30
5.2	Sources	35

Préambule

Ce rapport s'inscrit dans le cadre de la matière "Projet de Mathématiques" de l'université de Toulouse (Paul Sabatier).

L'objectif de ce rapport est de présenter et de prouver l'algorithme d'Hastings-Metropolis. À la fois d'expliquer en détail son fonctionnement, mais aussi de comprendre l'intérêt qu'il présente dans les sciences et en quoi il prend sa valeur.

Pour illustrer l'algorithme, nous nous intéresserons au problème du voyageur de commerce et étudierons sa complexité.

1 Introduction

Afin de comprendre en profondeur ce sujet, il est nécessaire de s'en faire une idée globale, même naïve, pour pouvoir suivre convenablement la ligne directrice de notre discours.

1.1 Informations générales et description historique

L'algorithme d'Hastings-Metropolis (noté "HM" dans ce rapport) consiste en une méthode d'échantillonnage stochastique permettant, à partir d'une distribution de probabilité donnée, de pouvoir décrire son comportement et d'obtenir des statistiques dessus. Cet algorithme prend sa valeur quand la distribution est difficile à analyser (systèmes multidimensionnels, par exemple) et permet de résoudre de nouveaux problèmes avec une complexité temporelle réduite.

Cette méthode est marquante, car elle a été conceptualisée très tôt dans l'histoire de l'informatique et a mis des décennies avant d'être prouvée et expliquée entièrement.

Historiquement, c'est en 1949 que l'écriture de l'algorithme a été publiée dans un article de Nicholas Metropolis et Stanisław Ulam. La paternité de l'algorithme est soumise à débat, car l'algorithme s'inscrit sous le nom de son chef de projet (Metropolis), alors que l'équipe composée de Nicholas Metropolis, Arianna et Marshall Rosenbluth, Augusta et Edward Teller a contribué à cette méthode. Ils étudiaient alors plus particulièrement le cas de la distribution de Boltzmann, une des distributions les plus utilisées en physique statistique, dans des travaux datant de 1953.

Cela illustre une dynamique fréquente dans les sciences, où le mérite est disproportionnellement attribué à une personne alors qu'il s'agit des efforts de toute une équipe. En particulier, Arianna Rosenbluth était considérée comme brillante par le monde scientifique.

En 1970, W. K. Hastings (1930-2016) a étendu l'algorithme au cas d'une distribution quelconque, et c'est cette version généralisée qui est

connue sous le nom d'algorithme de Metropolis-Hastings. Cette extension a eu de nombreuses applications dans divers domaines scientifiques, comme en statistique bayésienne (espaces complexes multidimensionnels), en biologie computationnelle (analyse des séquences génétiques), en économie et en finance (modèles stochastiques MCMC en général), etc.

Quant à lui, le problème du voyageur de commerce (dit "TSP", comme Travelling Salesman Problem en anglais) est un problème classique et bien connu pour être un problème NP-difficile, ce qui signifie qu'il n'existe actuellement aucune méthode connue capable de le résoudre de manière optimale en temps polynomial pour toutes les tailles d'instances, et qu'il est au moins aussi difficile que les problèmes de NP. L'origine du problème est assez incertaine : il a été formulé pour la première fois vers 1850 dans un manuel d'un commerçant voyageant en Suisse et en Allemagne. Ce n'est que dans les années 1930 que le problème fut annoncé d'abord comme un casse-tête (par William Rowan et Thomas Kirkman), puis étudié (par, entre autres, Thomas Kirkman, Jillian Beardwood, J. H. Halton et John Hammersley). Le problème consiste à déterminer le chemin le plus court passant par tous les points d'un graphe une seule fois chacun, en terminant par le point de départ (recherche d'un cycle hamiltonien le plus court). Les distances peuvent être dites symétriques ou asymétriques, c'est-à-dire que la distance entre eux varie en fonction de la direction du déplacement. On peut illustrer ce problème grâce à un voyageur de commerce devant vendre ses produits dans chacune des villes en un minimum de temps. Ce problème peut donc être naturellement représenté par un graphe.

1.2 Pourquoi HM permet-il de résoudre le problème du voyageur de commerce ?

Le principe mathématique derrière HM repose sur la construction dynamique d'une chaîne de Markov, qui n'est pas connue au préalable mais se développe au fil des itérations. À mesure que l'algorithme progresse, le comportement de cette chaîne converge vers la distribution cible, permettant d'obtenir un échantillon fiable de solutions par rapport aux états de la distribution.

Quant au problème du voyageur de commerce, il peut être représenté par un graphe orienté ayant un sommet pour chaque ville et une arête pour chaque temps de trajet. Nous verrons par la suite qu'une chaîne de Markov peut être associée à un graphe orienté, c'est-à-dire que le problème du voyageur de commerce est soluble par HM.

Par soluble, il est important de préciser qu'il s'agit d'une solution approximative parce que le TSP est NP-difficile comme mentionné précédemment. En conséquence, l'objectif de HM n'est pas d'apporter la réponse exacte au problème, mais une approximation fiable de la solution, ce qui peut

sembler contre-intuitif. Cette approximation est la caractéristique principale des méthodes MCMC que nous aborderons dans leur partie d'ici.

Il convient de souligner que, dans le cas du TSP, c'est la version d'optimisation que l'on s'intéresse, par opposition au problème décisionnel, qui, lui, est NP-complet. La différence entre ces deux formulations repose sur leur objectif : dans le TSP décisionnel, on pose une question du type "Existe-t-il un chemin de coût inférieur ou égal à une valeur donnée ?", à laquelle il s'agit de répondre par oui ou non. Alors que le TSP d'optimisation cherche à partir de la situation initiale la meilleure solution : donc optimiser le chemin au fil des itérations. C'est cette version du problème pour laquelle l'approche par l'échantillonnage stochastique est envisageable. Précisons toutefois que HM n'est habituellement pas la méthode que l'on utilise pour le TSP en pratique. Usuellement, le nombre de villes est grand (suffisamment pour que la solution par brute force ne soit pas trouvable) mais pas absurde (une centaine par exemple), on préfère utiliser des versions déterministes optimisées comme l'algorithme de Held-Karp de complexité $O(2^n \times n^2)$, ou dans le cas de situation avec encore plus de villes, des méthodes similaires à HM plus adaptées comme l'échantillonnage de Gibbs.

En somme, il est important de comprendre que HM est un outil adapté à la résolution de problèmes dont l'espace des solutions est trop vaste pour être exploré exhaustivement dans un temps raisonnable. C'est donc par l'approximation de la solution optimale, à travers des échantillons, que cette méthode permet de trouver une solution satisfaisante. Cette nuance est essentielle, car elle met en lumière la sophistication de HM en tant qu'outil d'approximation pour des problèmes complexes.

1.3 Objectif de ce rapport

L'objectif de ce rapport est d'apporter une solution fiable au TSP grâce à HM, nous prouverons alors le fonctionnement de ce dernier grâce à la théorie mathématique des chaînes de Markov. Nous en expliciterons les définitions et propriétés fondamentales dans un premier temps, qui permettront de prouver l'algorithme d'HM dans un second temps. Nous présenterons alors à la fin du rapport une implémentation personnelle de l'algorithme.

2 Chaînes de Markov à États finis

2.1 Définitions fondamentales

Temps Discret et Temps Continu

Soit T un ensemble d'indices numériques représentant le temps.

Il existe deux types de modélisation temporelle :

- (1) Un processus à **temps discret** signifie que l'on considère les valeurs de la modélisation comme espacées régulièrement dans le temps. On peut prendre des ensembles dénombrables comme $T = \mathbb{N}$ ou $T = \mathbb{Z}$ où chaque instant est distinct.
- (2) Un processus à **temps continu** signifie que T n'est pas dénombrable. Il existe toujours un temps intermédiaire entre deux indices de T . Cela signifie que le processus évolue en tout instant dans un continuum temporel, on peut prendre des ensembles non dénombrables comme $T = \mathbb{R}_+$ où chaque instant est continu.

Dans ce rapport nous nous intéresserons uniquement au temps discret, pour pouvoir modéliser le problème du voyageur de commerce. De plus, cette distinction joue un rôle fondamental dans la classification et l'analyse des **processus stochastiques** et n'implique pas les mêmes théorèmes.

Processus Stochastique

Soit $(\Omega, \mathcal{F}, \mathbb{P})$ un espace de probabilité.

Soit T un ensemble d'indices discret ou continu (souvent $T = \mathbb{N}$ ou \mathbb{R}_+).

Un **processus stochastique** est une famille de variables aléatoires $\{X_t\}_{t \in T}$ définies sur $(\Omega, \mathcal{F}, \mathbb{P})$ et à valeurs dans un espace d'états E (appelé espace d'états du processus).

Un processus stochastique permet de modéliser un système évoluant de manière aléatoire en fonction du temps.

Différents types de processus peuvent être étudiés en fonction des propriétés de dépendance temporelle et de la nature de l'espace d'états E . Au cours de ce rapport, nous nous concentrerons l'un des principaux processus stochastiques à temps discret et à espace d'états fini ; les **chaînes de Markov**.

Chaîne de Markov

Soit E un ensemble fini ou dénombrable.

Soit $(X_n)_{n \in \mathbb{N}}$ une suite de variables aléatoires à valeurs dans l'espace d'états E .

$(X_n)_{n \in \mathbb{N}}$ est appelée **chaîne de Markov** si et seulement si :

- (1) Sa loi de probabilité initiale X_0 est bien définie.
- (2) Elle respecte la **propriété de Markov**, telle que :

$$\forall n \geq 0, \quad \exists x_1, x_2, \dots, x_{n+1} \in E,$$

$$\mathbb{P}(X_{n+1} = x_{n+1} \mid X_1 = x_1, \dots, X_n = x_n) = \mathbb{P}(X_{n+1} = x_{n+1} \mid X_n = x_n).$$

Une chaîne de Markov est un processus de Markov à temps discret ou à temps continu et à espace d'états discret. Un processus de Markov est un processus stochastique possédant la propriété de Markov : l'information utile pour la prédiction du futur est entièrement contenue dans l'état présent du processus et n'est pas dépendante des ses états antérieurs.

Autrement dit, la loi de probabilité \mathbb{P} régissant la transition de l'état présent X_n vers l'état futur X_{n+1} dépend uniquement du dernier terme X_n , et reste totalement indépendante des tous ses états antérieurs $\{X_0, X_1, \dots, X_{n-1}\}$.

Cette propriété, que l'on peut qualifier de "sans mémoire" ou de propriété de Markov, constitue la caractéristique fondamentale de ces processus stochastiques.

Chaîne de Markov Homogène

Un chaîne de Markov est dite homogène quand $\forall n \in \mathbb{N}$:

$$\mathbb{P}[X_{n+1} = j \mid X_n = i] = \mathbb{P}[X_1 = j \mid X_0 = i].$$

En somme, une chaîne de Markov homogène ne dépend pas des états précédents (propriété de Markov) et garantit que son comportement reste inchangé au fil du temps, c'est-à-dire que les probabilités de transition restent constantes quelque soit $t \in T$ (homogénéité de la chaîne de Markov). Dans ce rapport, toutes les chaînes de Markov seront considérées comme homogènes.

2.2 Propriétés fondamentales

2.2.1 Matrice de Transition

Matrice de Transition

Soit $(X_n)_{n \in \mathbb{N}}$ une chaîne de Markov homogène à valeurs dans un ensemble fini ou dénombrable E .

Soit la famille de nombres réels $(X_n)_{n \in \mathbb{N}}$ permettant de passer d'un état à un autre, définie comme :

$$\forall n \geq 0, \forall i, j \in E, \quad P_{ij} = \mathbb{P}(X_{n+1} = j \mid X_n = i).$$

La **matrice de transition** P regroupe ces probabilités tel que chaque élément P_{ij} représente la probabilité de passer de l'état i à l'état j en une seule étape. Lorsque l'ensemble des états E est discret, par exemple $E = \{1, 2, \dots, N\}$, la matrice s'exprime sous la forme suivante :

$$P = \begin{bmatrix} P_{11} & P_{12} & \cdots & P_{1N} \\ P_{21} & P_{22} & \cdots & P_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ P_{N1} & P_{N2} & \cdots & P_{NN} \end{bmatrix}$$

Matrice stochastique par lignes/colonnes

Une matrice **stochastique par lignes** (appelée aussi **stochastique à droite**) est une matrice dont la somme des probabilités de ses lignes vaut 1 chacune et toutes ses probabilités sont positives :

$$\forall i, \quad \sum_{j=1}^N P_{ij} = 1 \quad \text{et} \quad 0 \leq P_{ij} \leq 1.$$

Respectivement, une matrice est **stochastique par colonnes** (dite aussi **stochastique à gauche**) lorsque :

$$\forall j, \quad \sum_{i=1}^N P_{ij} = 1 \quad \text{et} \quad 0 \leq P_{ij} \leq 1.$$

Une matrice est dite **bistochastique** lorsqu'elle est à la fois stochastique par lignes et par colonnes.

Dans le cadre des chaînes de Markov homogènes, toutes les matrices de transition P sont stochastiques par lignes car les probabilités de transition depuis chaque état sont normalisées (chaque ligne représentant le vecteur

des probabilités de transition depuis un état donné vers l'ensemble des autres états). C'est-à-dire que la somme des probabilités pour passer d'un état à un autre vaut 1.

De manière analogue, si l'espace des états est infini dénombrable (par exemple $E = \{1, 2, 3, \dots\}$), on indexe les états de la même façon et la condition

$$\forall i \in E, \quad \sum_{j \in E} P_{ij} = 1,$$

reste valable.

2.2.2 Représentation sous forme de Graphe Orienté

Graphe Orienté d'une Chaîne de Markov

Une chaîne de Markov homogène peut toujours être représentée sous forme d'un graphe orienté $G = (V, A)$, où :

- (1) V est l'ensemble des sommets, correspondant aux états de l'espace d'états E .
- (2) A est l'ensemble des arcs, où chaque arc possède une pondération correspondant à la probabilité de transition $P_{i,j}$. Un arc de i vers j est représenté que si $P_{i,j} > 0$.

Représenter graphiquement une chaîne de Markov homogène permet de clarifier visuellement les différentes dynamiques de transitions entre chaque état du système et de comprendre la structure du processus stochastique.

Exemple

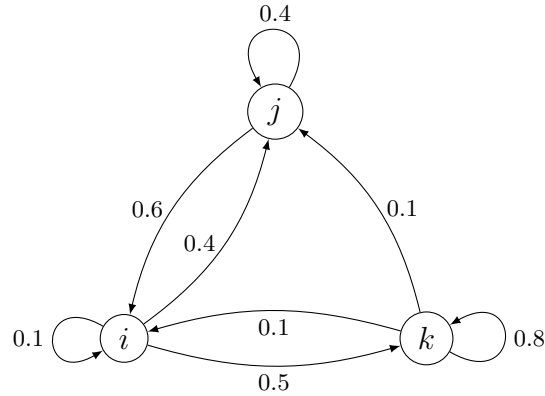
Considérons par exemple une chaîne de Markov ayant trois états i, j, k , ordonnés respectivement, et dont la matrice de transition est donnée par :

$$P = \begin{bmatrix} 0.1 & 0.4 & 0.5 \\ 0.6 & 0.4 & 0 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}$$

Dans cette configuration, la probabilité de transition de l'état i vers l'état j est donnée par :

$$P_{i,j} = P_{1,2} = \mathbb{P}(X_{n+1} = j \mid X_n = i) = 0.4$$

Nous obtenons ainsi le graphe suivant :



2.2.3 Matrice de transition en k -pas

Matrice de transition pour k -transitions

Soit $(X_n)_{n \in \mathbb{N}}$ une chaîne de Markov homogène d'espace d'états E fini ou dénombrable tel que $\{1, 2, \dots, N\}$. On note P sa matrice de transition.

Pour tout état E_i et E_j et pour tout entier naturel $k \geq 1$, le coefficient de la ligne i et de la colonne j de la **matrice P^k** est la **probabilité de passer de l'état E_i à celui E_j en k transitions**.

Les chaînes de Markov homogènes permettent de déterminer l'état d'un système après k transitions, c'est-à-dire au bout du k -ième mouvement dans la chaîne, en élevant la matrice de transition à la puissance k . Plus formellement, on a :

$$P_{i,j}^{(k)} = \mathbb{P}(X_k = j \mid X_0 = i).$$

Cet résultat vient de la propriété de Markov, qui indique que la probabilité de transition d'un état i à un état j dépend uniquement du dernier pas réalisé, et non de tous les précédents.

Démonstration

La démonstration de cette propriété passe par celle de l'équation de Chapman-Kolmogorov, telle que $\forall i, j \in E$ et $\forall n, m \in \mathbb{N}$:

$$\mathbb{P}(X_{n+m} = j \mid X_0 = i) = \sum_{k \in E} \mathbb{P}(X_m = j \mid X_0 = k) \cdot \mathbb{P}(X_n = k \mid X_0 = i).$$

Ce qui équivaut en termes matriciels :

$$P_{i,j}^{(n+m)} = \sum_{k \in E} P_{i,k}^{(n)} \cdot P_{k,j}^{(m)}.$$

Cette relation peut s'interpréter en disant que pour passer de i à j en $n + m$ étapes il a fallu en n étapes aller de i à un certain k puis en m étapes aller de k à j .

On reconnaît alors l'expression de l'associativité du produit matriciel tel que :

$$P^{n+m} = \underbrace{P \cdot \dots \cdot P}_{n \text{ fois}} \cdot \underbrace{P \cdot \dots \cdot P}_{m \text{ fois}} = P^n P^m.$$

Exemple

Reprenons par exemple la matrice P donnée dans la sous-section 2.2.2 ci-dessus, la matrice de transition en 5 étapes, notée P^5 , représente l'ensemble des probabilités permettant de passer de chaque état i à un état j au bout d'exactly 5 étapes :

$$P^5 = \begin{bmatrix} 0,22350 & 0,24385 & 0,53265 \\ 0,24150 & 0,26140 & 0,49710 \\ 0,20595 & 0,22252 & 0,57153 \end{bmatrix}$$

Toujours en reprenant notre exemple, la probabilité de passer de l'état i à l'état j au bout d'exactly 5 étapes est donc de :

$$P_{i,j}^{(5)} = \mathbb{P}(X_{k+5} = j \mid X_k = i) = 24,385\%$$

2.3 Classes d'Équivalence

2.3.1 Classification des États

Dans une chaîne de Markov, chaque État peut être classifié en différentes catégories en fonction de ses liaisons avec les autres États. Ces différentes classifications nous permettront d'analyser le comportement de la chaîne en temps long.

Accessibilité et communication

Soient i et j deux États de E . On dit que j est **accessible** à partir de i si :

$$\exists n \geq 1, \quad P_{i,j}^{(n)} > 0.$$

Ce que l'on note $i \rightarrow j$.

On dit que les deux États i et j **communiquent** si $i \rightarrow j$ et $j \rightarrow i$. On a ainsi :

$$\exists n \geq 0, \quad P_{i,j}^{(n)} > 0 \quad \text{et} \quad \exists m \geq 0, \quad P_{j,i}^{(m)} > 0.$$

Ce que l'on note $i \leftrightarrow j$.

En particulier, $i \rightarrow j$ si et seulement s'il existe une suite d'États (i_0, i_1, \dots, i_n) avec $i_0 = i$ et $i_n = j$ telle que :

$$P(X_n = j, \dots, X_1 = i_1 \mid X_0 = i) = P(i_{n-1}, j) \cdots P(i, i_1) > 0.$$

Ce qui veut dire qu'il existe un chemin de probabilité strictement positive qui mène de i à j .

État récurrent/transient

Un État i est dit **récurrent** si, en partant de cet État, il est certain d'y retourner en un nombre fini de pas. Un État non récurrent est dit **transient**.

On pose :

$$\tau_i = \inf\{\exists n \in \mathbb{N}^*, n \geq 1 \mid X_n = i\}$$

On dit alors que i est récurrent si :

$$\mathbb{P}_i(\tau_i < \infty) = \mathbb{P}(\tau_i < \infty \mid X_0 = i) = 1$$

Et on dit que i est transient si :

$$\mathbb{P}_i(\tau_i < \infty) = \mathbb{P}(\tau_i < \infty \mid X_0 = i) < 1$$

Ces propriétés impliquent que si l'État i est récurrent (respectivement

transient), on y retournera un nombre infini (respectivement fini) de fois pour une infinité de transitions.

État absorbant/Évanescent

Un État i est appelé **absorbant** si, une fois atteint, la chaîne reste dans cet État avec probabilité 1. Formellement, cet État satisfait :

$$P_{i,i} = 1 \quad \text{et} \quad \forall j \neq i, \quad P_{i,j} = 0.$$

A l'inverse, tout État ne respectant pas cette condition est appelé État **non-absorbant** ou encore État **Évanescent**.

État périodique/apériodique

Un État i est dit **périodique** si la chaîne ne peut revenir à cet État qu'après un nombre d'étapes multiple (un certain entier $d > 1$), appelé la période de i . La période $d(i)$ est définie comme :

$$d(i) = \text{PGCD}\{n \geq 1 \mid P_{i,i}^{(n)} > 0\}.$$

Si $d(i) = 1$, l'État i est dit **apériodique**, ce qui signifie qu'il est possible de revenir à cet État à chaque nouvelle transition / sans contrainte de périodicité.

2.3.2 Classes de communication

Classe de communication

Soit deux États communicants i et j . Selon la définition précédente, on a que \leftrightarrow est une relation d'équivalence. De plus dans le contexte des chaînes de Markov, ces classes d'équivalences sont appelées **classes de communications**. Une classe de communication est donc un ensemble d'États accessibles les uns depuis les autres.

Démonstration

Montrons que la relation \leftrightarrow est une relation d'équivalence.

Il nous suffit alors de vérifier les trois propriétés suivantes :

1. **Réflexivité** : Pour tout État $i \in E$, $i \leftrightarrow i$.
Pour tout État $i \in E$, il existe trivialement un chemin $i \rightarrow i$. En effet, en considérant le chemin de longueur zéro (via le fait que $P_{ii}^{(0)} = 1$ en définissant $P^{(0)}$ comme la matrice identité), on

conclut directement que $i \leftrightarrow i$.

2. **Symétrie** : Si $i \leftrightarrow j$, alors $j \leftrightarrow i$.

Si $i \leftrightarrow j$, par définition il existe un chemin $i \rightarrow j$, (et inversement un chemin de $j \rightarrow i$). Ainsi, la relation est symétrique : $i \leftrightarrow j \Rightarrow j \leftrightarrow i$.

3. **Transitivité** : Si $i \leftrightarrow j$ et $j \leftrightarrow k$, alors $i \leftrightarrow k$.

Supposons que $i \leftrightarrow j$ et $j \leftrightarrow k$. En composant ces deux chemins, on obtient un chemin $i \rightarrow k$. De manière analogue, la composition des chemins $k \rightarrow j$ et $j \rightarrow i$ nous fournit un chemin $k \rightarrow i$. Ainsi, $i \leftrightarrow k$.

Comme la relation \leftrightarrow satisfait les propriétés de réflexivité, de symétrie et de transitivité, elle est bien une relation d'équivalence sur E . Les classes d'équivalences ainsi obtenues sont alors appelées **classes de communication**.

L'ensemble des classes de communication, engendrées par \leftrightarrow , d'une chaîne de Markov sont **disjointes** et forment une **partition** de l'ensemble des états de cette chaîne. Autrement dit, la relation d'équivalence \leftrightarrow partitionne l'ensemble des états d'une chaîne de Markov en classes de communication disjointes.

Propriétés de classe

On appelle **propriété de classe** une propriété structurelle commune à tous les états d'une même classe de communication.

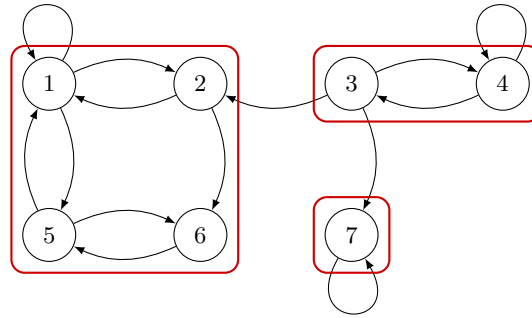
Les propriétés de récurrence ou transience d'un état, énoncées dans la partie 2.3.1 sont des propriétés de classe.

Cela signifie donc qu'au sein d'une même classe de communication, tous les états sont soit récurrents, soit transients. On parle alors de classes de récurrence ou de transience.

L'intérêt de ces distinctions est de diviser les chaînes de Markov en sous-ensembles analytiquement indépendants, ce qui permet ensuite de tirer des conclusions sur les sous-ensembles présentant ces propriétés.

Exemple

Prenons par exemple la chaîne de Markov suivante, dont la représentation sous forme de graphe orienté est donnée ci-dessous :



Dans cette chaîne de Markov, nous avons trois classes de communication :

- La classe 1, formée par les états $\{1, 2, 5, 6\}$.
 - Cette classe est fermée car il n'est pas possible de quitter cette classe une fois qu'un état de celle-ci est atteint. En effet, il n'existe pas d'état j tel que $P_{i,j} > 0$ pour $i \in \{1, 2, 5, 6\}$ et $j \notin \{1, 2, 5, 6\}$.
 - Chaque état de la classe est alors récurrent car il est certain de revenir à un des états $i \in \{1, 2, 5, 6\}$ après un nombre infini de transitions.
- La classe 2, formée par les états $\{3, 4\}$.
 - Cette classe est ouverte car il est possible de la quitter via $3 \rightarrow 2$ et $3 \rightarrow 7$.
 - Chaque état de la classe est alors transient car nous ne reviendrons pas aux états $i \in \{3, 4\}$ après un nombre infini de transitions.
- Et la classe 3, formée par l'état $\{7\}$.
 - Cette classe est fermée car il n'est pas possible de quitter cette classe une fois que l'état 7 est atteint. En effet, en partant de l'état 7, nous n'avons que $7 \rightarrow 7$.
 - Chaque état de la classe est alors récurrent car une fois que l'état 7 est atteint, la chaîne reste dans cet état avec probabilité 1.

2.4 Probabilités invariantes et théorème ergodique

2.4.1 probabilités invariantes

La loi stationnaire peut permettre sous certaines conditions d'exprimer le comportement à long terme de la chaîne de Markov.

Probabilités invariantes/stationnaire

Soit une chaîne de Markov $(X_n)_{n \in \mathbb{N}}$ à espace d'état E , et ayant pour matrice de transition P .

Une probabilité $\pi = (\pi_i)_{i \in E}$ sur l'espace d'état E est dite invariante (ou stationnaire) si elle satisfait la relation $\pi P = \pi$.

C'est-à-dire

$$\forall y \in E, \quad \sum_{x \in E} \pi(x) P_{x,y} = \pi(y).$$

Cela signifie que π est un vecteur propre à droite de la transposée P^\top , associé à la valeur propre 1 :

$$\pi P = \pi \iff P^\top \pi^\top = \pi^\top.$$

De plus dans notre cas, comme π est une probabilité, elle respecte les conditions suivantes :

$$\forall i \in E, \quad \pi_i \geq 0 \quad \text{et} \quad \sum_{i \in E} \pi_i = 1.$$

Existence d'une probabilité invariante

Soit $(X_n)_{n \in \mathbb{N}}$ une chaîne de Markov homogène sur un espace d'états E fini, de matrice de transition P . Alors $(X_n)_{n \in \mathbb{N}}$ admet au moins une probabilité invariante.

Démonstration

Par définition, π est invariante si $\pi P = \pi$, ou encore $P^\top \pi^\top = \pi^\top$, c'est-à-dire que π^\top est un vecteur propre de P^\top associé à la valeur propre 1. Or P et sa transposée P^\top ont même valeurs propres (car même polynôme caractéristique), et comme P est une matrice stochastique, on a $P \mathbf{1} = \mathbf{1}$, où $\mathbf{1}$ désigne le vecteur de $\mathbb{R}^{|E|}$ dont tous les coefficients sont égaux à 1. Donc 1 est vecteur propre de P et donc de P^\top , donc il existe un vecteur ligne μ non nul tel que $\mu P = \mu$. Soit π le vecteur

ligne tel que $\pi(x) = |\mu(x)|$, pour tout $x \in E$, et montrons que π est encore vecteur propre à gauche de P . Comme les coefficients de π sont positifs, il suffira alors de diviser π par la somme de ses coefficients pour obtenir une probabilité invariante. On a, pour tout $x \in E$,

$$\begin{aligned}\pi(x) &= |\mu(x)| = |\mu P_x| = \left| \sum_{y \in E} \mu(y) P_{y,x} \right|, \\ &\leq \sum_{y \in E} |\mu(y)| P_{y,x} = \sum_{y \in E} \pi(y) P_{y,x} = \pi P_x,\end{aligned}$$

donc $\pi(x) \leq \pi P_x$, pour tout $x \in E$. De plus,

$$\begin{aligned}\sum_{x \in E} \pi(x) &= \sum_{x \in E} |\mu(x)| = \sum_{x \in E} \left| \sum_{y \in E} \mu(y) P_{y,x} \right|, \\ &\leq \sum_{x \in E} \sum_{y \in E} |\mu(y)| P_{y,x} = \sum_{y \in E} \pi(y) \sum_{x \in E} P_{y,x} = \sum_{y \in E} \pi(y).\end{aligned}$$

Finalement, $\pi(x) \leq \pi P_x$ pour tout $x \in E$, et les vecteurs ont même somme, c'est donc qu'ils sont égaux : on a

$$\sum_{x \in E} (\pi P_x - \pi(x)) = 0,$$

et comme $\pi P_x - \pi(x) \geq 0$ pour tout $x \in E$, on a $\pi P_x = \pi(x)$. Ainsi, π renormalisé par la somme de ses coefficients est bien une probabilité invariante.

Chaîne de Markov irréductible

Une chaîne de Markov est dite **irréductible** si le graphe qui lui est associé est fortement connexe.

Autrement dit, pour tous les couples (i, j) d'états de E , avec $i \neq j$, il existe un chemin (de probabilité strictement positive) permettant d'aller de i à j , et réciproquement : $i \leftrightarrow j$.

Probabilités invariantes d'une chaîne de Markov irréductible

Toute chaîne de Markov homogène irréductible admet une mesure invariante strictement positive sur E et toutes les probabilités invariantes sont proportionnelles entre elles.

2.4.2 Théorème ergodique

Théorème ergodique (admis)

Soit E fini et (X_n) une chaîne de Markov irréductible, pour toute fonction f , et toute loi initial ν_0

$$\forall k > 0 : \lim_{n \rightarrow \infty} \frac{1}{n - k + 1} \sum_{i=k}^n f(X_i) = \int_E f(x) d\mu(x)$$

avec μ la probabilité invariante.

On peut donc estimer $\mu(x)$ avec $\mathbb{1}_{\{x\}}$, où :

$$\mathbb{1}_A : B \rightarrow \{0, 1\}, \quad A \subseteq B$$

$$\mathbb{1}_A(x) = \begin{cases} 1, & \text{si } x \in A \\ 0, & \text{sinon} \end{cases}$$

Le théorème ergodique nous dit donc que la moyenne de f le long de la trajectoire de la chaîne se rapproche empiriquement, lorsque ce temps n est grand, de sa moyenne spatiale par rapport à la loi invariante. En particulier, ceci donne une nouvelle interprétation de la mesure invariante comme la fréquence du temps d'occupation d'un état en temps long.

Le théorème ergodique peut également être perçu comme une sorte de loi des grands nombres pour les chaînes de Markov.

2.5 Comportement en temps long

Maintenant que nous avons défini la loi stationnaire, nous pouvons énoncer le théorème de convergence d'une chaîne de Markov homogène.

Théorème de convergence

Soit $(X_n)_{n \in \mathbb{N}}$ une chaîne de Markov homogène, irréductible et apériodique. L'espace d'états fini. Soit P sa matrice de transition.

Alors, il existe une unique mesure invariante π , et pour tout état initial $i \in E$ et tout $j \in E$,

$$\lim_{n \rightarrow \infty} P_{i,j}^n = \pi_j.$$

Ce qui implique :

$$\lim_{n \rightarrow \infty} \frac{1}{n} \sum_{k=0}^{n-1} \pi_i P^k = \pi_j.$$

Donc la loi de X_n converge vers π indépendamment de l'état initial, on a une convergence en loi.

Démonstration

Commençons par montrer l'unicité de la probabilité invariante. La transposée d'une probabilité invariante est un vecteur propre de P^\top associé à la valeur propre 1. Comme les valeurs propres de P et de P^\top sont les mêmes et ont même multiplicité, il suffit donc de montrer que l'espace propre associé à P pour la valeur propre 1 est de dimension 1.

Soit f un vecteur propre de P pour la valeur propre 1, i.e., $Pf = f$. Comme le vecteur 1 est vecteur propre, il s'agit de montrer que f est constante.

Soit x_i le point où f atteint son maximum, i.e. $f(x_i) = \max\{f(x) \mid x \in E\}$. Soit y un autre état. Comme P est irréductible, y communique avec x_i , donc il existe $n \geq 1$ tel que $P^n(x_i, y) > 0$. Comme $Pf = f$, on a aussi $P^n f = f$. Alors, d'une part,

$$f(x_i) = P^n f(x_i) = \sum_{y \in E} f(y) P^n(x_i, y),$$

et d'autre part,

$$f(x_i) = \sum_{y \in E} f(x_i) P^n(x_i, y),$$

donc on a

$$\sum_{y \in E} P^n(x_i, y) (f(x_i) - f(y)) = 0,$$

qui est une somme de termes positifs. Ainsi, comme $P^n(x_i, y) > 0$, on a $f(y) = f(x_i)$, et donc, f est constante, et la chaîne admet une unique probabilité invariante, notée π_j .

Valeur de convergence

Soit $(X_n)_{n \in \mathbb{N}}$ une chaîne de Markov homogène irréductible. Soit j . Alors, pour toute mesure initiale λ et pour tout état j :

$$\frac{1}{n} \sum_{k=1}^n 1_{\{X_k=j\}} \rightarrow \pi_j,$$

où π est l'unique probabilité invariante.

Convergence des méthodes de Monte-Carlo

Soit X une chaîne de Markov homogène irréductible, alors pour toute mesure initiale λ et pour toute fonction mesurable bornée f , on a :

$$\frac{1}{n} \sum_{k=1}^n f(X_k) \xrightarrow{n \rightarrow \infty} \int f d\pi \quad \text{p.s. sous } \mathbb{P}_\lambda.$$

Il s'agit de la méthode utilisée dans les algorithmes de Monte-Carlo.

Ces théorèmes nous permettront de conclure quant à la preuve de Hasting-Metropolis.

3 Fonctionnement de l'Algorithme de Hastings-Metropolis

3.1 Qu'est-ce qu'une méthode dite MCMC ?

Les méthodes MC dites de Monte-Carlo ont pour but d'approcher de manière empirique des valeurs par un processus aléatoire répété un nombre suffisant de fois. Elles interviennent quand une solution déterministe est trop difficile à obtenir, comme pour calculer des intégrales ou générer des échantillons de distributions statistiques. Les méthodes MC sont très fiables malgré l'aléatoire utilisé, pourvu que l'algorithme ait les bonnes optimisations telles que le nombre de répétitions, la précision exigée, la vitesse de convergence vers les valeurs cherchées, etc.

Les méthodes MC sont utilisées dans beaucoup de domaines des sciences (comme la physique, la chimie, la biologie, les mathématiques statistiques, l'intelligence artificielle, la finance et la cryptographie).

Les méthodes MCMC (Monte-Carlo par chaînes de Markov) sont une sous-famille des algorithmes MC qui construisent, dans leur fonctionnement, une chaîne de Markov dont la distribution stationnaire est celle que l'on souhaite estimer.

Il devient ainsi possible d'exploiter les propriétés des chaînes de Markov, notamment leur convergence vers une distribution stationnaire et leur propriété d'ergodicité, qui peuvent être plus utiles que la loi des grands nombres seule, utilisée dans les méthodes MC classiques.

La principale différence entre les méthodes MC et MCMC réside dans la dépendance entre les points générés. Les méthodes MC choisissent aléatoirement des points de l'espace, réalisant ainsi un échantillonnage direct de la distribution cible, avec des échantillons indépendants les uns des autres. Leur seul point commun est d'appartenir à l'intervalle des valeurs possibles de la distribution de départ. En revanche, les méthodes MCMC reposent sur une marche aléatoire : elles génèrent chaque nouveau point à partir du précédent selon une règle de transition déterminée. Cela crée une chaîne de points corrélés, qui possède les propriétés des chaînes de Markov.

Ainsi, lorsque la distribution est concentrée sur une certaine partie de l'espace, l'utilisation de MCMC sera généralement préférable, car elle permet d'obtenir, dans un temps raisonnable, des échantillons représentatifs de cette distribution.

On pourra citer les méthodes MCMC d'échantillonnage de Gibbs (et ses variantes comme le Blocked Gibbs Sampling ou l'Adaptive Gibbs Sampling), qui prennent leur intérêt lorsque les variables de la distribution sont conditionnelles. Il existe aussi l'échantillonnage par tranches, qui échantillonne progressivement la distribution par "tranches" de valeurs, ou bien des optimisa-

tions de HM, telles que la méthode hamiltonienne de Monte-Carlo, qui calcule le gradient pour faciliter la marche du programme et rejeter moins de valeurs, tout en gagnant en performances.

3.2 Définitions

TODO

3.3 Description des étapes de l'algorithme

Voici une description de cet algorithme :

Hasting Metropolis

1. Il faut avant tout choisir un point x_0 , comme étant le premier échantillon de notre loi cible, et aussi une probabilité de transition g , en donnant les fonctions correspondant à $g_y(x)$ et $g(x)$, $\forall x, y \in E$.
2. Ensuite nous itérons sur t allant de 0 à N (notre nombre d'itérations voulu)
 - On tire x avec $g_{x_t}(x)$
 - On pose $\alpha := \frac{\pi(x)g_x(x_t)}{\pi(x_t)g_{x_t}(x)}$ (notons que si nous ne possédons qu'une densité proportionnelle à π , f nous pouvons poser $\alpha := \frac{f(x)g_x(x_t)}{f(x_t)g_{x_t}(x)}$)
 - On tire $u \in [0; 1]$, tel que $u \sim \mathcal{U}(0, 1)$
 - Si $u \leq \alpha$ alors : $x_{t+1} := x$
 - Sinon : on conserve l'état précédent $x_{t+1} := x_t$
3. La séquence $\{x_0, x_1, \dots, x_{N-1}\}$ constitue donc l'échantillon obtenu à partir de la chaîne de Markov associée à la loi π

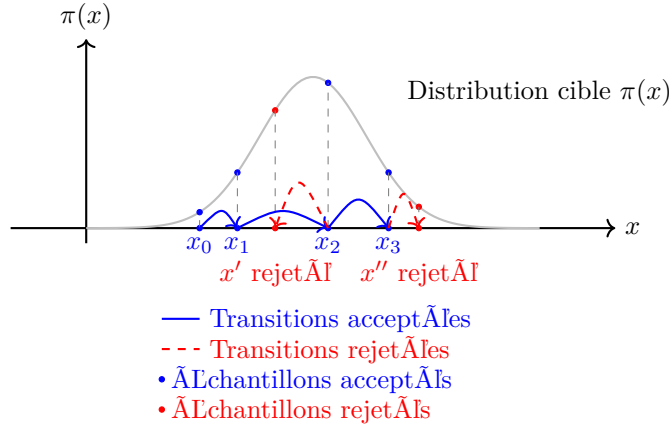
3.4 Preuve de l'algorithme

TODO

3.5 Exemples d'applications simples de HM

Afin de comprendre le déroulement de l'algorithme, on peut s'appuyer sur un graphique illustrant le début d'une simulation de Metropolis-Hastings dans un espace d'états continu. Bien que cet exemple diffère du reste du rapport qui traite d'états discrets, les propriétés fondamentales de l'algorithme restent les mêmes.

Ce cas a été choisi pour sa clarté visuelle : il permet de suivre facilement les étapes de l'algorithme, voici le schéma :



Dans cet exemple, la distribution cible $\pi(x)$ est représentée par la courbe grise, ici une distribution normale. L'axe des ordonnées indique les valeurs prises par cette densité, tandis que l'axe des abscisses correspond aux différents états que la chaîne de Markov peut atteindre au fil des itérations.

Nous choisissons arbitrairement l'état x_0 comme départ et cherchons à créer un échantillon de taille 4. Cet état fait donc partie de l'échantillon. L'algorithme propose un nouvel état x_1 , tiré avec la loi de transition $g(x_1 | x_0)$ à partir de l'état courant. On tire alors le taux d'acceptation $\alpha = \min\left(1, \frac{\pi(x_1)g(x_0|x_1)}{\pi(x_0)g(x_1|x_0)}\right)$ connu à un coefficient près, et un tirage de la loi uniforme $u \sim \mathcal{U}(0, 1)$. Si $u \leq \alpha$ on accepte ce nouvel état, sinon on le rejette. Sur le schéma, x_1 est proposé puis accepté, suivi de x_2 également accepté. Un autre candidat x' est ensuite rejeté, ce qui conduit à conserver x_2 pour l'itération suivante, et ainsi de suite.

À noter que certains états peuvent être sélectionnés même s'ils semblent se situer dans des zones de faible densité (c'est le cas ici de x_3).

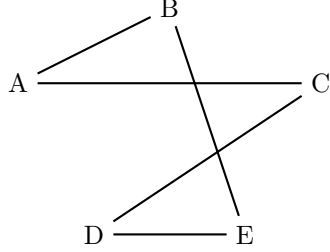
Considérons maintenant un cas simple du TSP avec 5 villes A, B, C, D et E , donc dans un système d'états finis par lequel on essaie d'obtenir le chemin de longueur minimale.

La différence notable avec le précédent exemple est que l'on ne cherche plus à échantillonner des points sur une courbe de densité mais des chemins entre les villes. Chaque possibilité est donc un ordre de visite des villes, et il en existe $5! = 120$ au total.

À l'issue de l'exploration, on obtient alors un échantillon de chemins optimisés les plus courts qui ont été sélectionnés plus fréquemment. On peut ainsi approcher de manière probabiliste une solution optimale au problème du voyageur de commerce.

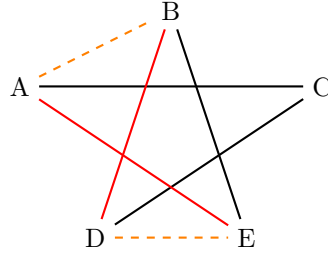
Le schéma suivant représente la situation initiale x :

Diagramme initial à optimiser



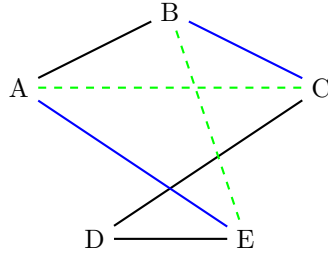
Voici un exemple de chemin refusé par HM, en noir et rouge, les pointillés représentant les différences avec l'état d'origine :

Exemple de proposition rejetée par HM



Et un exemple de chemin accepté par HM, en noir et bleu :

Exemple de proposition retenue par HM



3.6 Application au cas du voyageur de commerce

Dans le cas du problème du voyageur de commerce (dit “TSP”), l’algorithme nécessite une application adaptée. Notre espace d’état E correspond aux permutations des points. Par exemple, dans le cas de 3 points x_1, x_2, x_3 , $E = \{(x_1, x_2, x_3), (x_2, x_1, x_3), (x_2, x_3, x_1), (x_1, x_3, x_2), (x_3, x_2, x_1), (x_3, x_1, x_2)\}$, ce qui, avec notre implémentation actuelle, nécessiterait de manipuler une matrice 6×6 . Lorsque le nombre de villes augmente, nous aurions besoin d’une matrice carrée de taille $n!$, où n est le nombre de points, ce qui devient rapidement irréaliste.

Nous définissons donc des fonctions adaptées sur l'espace d'état E :

- Une fonction de distance totale sur les permutations des coordonnées des villes $D : E \rightarrow \mathbb{R}$,

$$D(p) = \sum_{i=1}^n \|x_i - x_{i+1}\| : \forall p = (x_1, x_2, \dots, x_n) \in E, x_{n+1} = x_1$$

- Une distribution cible, avec $\beta \in \mathbb{R}$ un paramètre ajustable

$$\pi(p) = e^{-\beta \cdot D(p)}$$

- Une variable aléatoire $S : E \rightarrow E$, suivant une loi de chaîne de Markov. Nous pouvons modéliser cette transition par une fonction `permutation`($[x_1, \dots, x_n]$), qui renvoie la même liste avec seulement deux villes permutes.

Nous pouvons donc redéfinir l'algorithme avec le pseudo code suivant :

Hasting Metropolis TSP

1. On prend p_0 , une permutation donnée
2. Ensuite nous itérons sur t allant de 0 à N (notre nombre d'itérations voulu)
 - On tire p avec la variable aléatoire S_{p_t} , on peut utiliser la fonction "permutation" définie plus tôt.
 - Calculer $\alpha := \frac{\pi(p)}{\pi(p_t)}$ si la transition est symétrique, c'est-à-dire $\mathbb{P}(S_{p_t} = p) = \mathbb{P}(S_p = p_t)$.
 - Sinon $\alpha := \frac{\pi(p)\mathbb{P}(S_p = p_t)}{\pi(p_t)\mathbb{P}(S_{p_t} = p)}$
 - On tire $u \in [0; 1]$, tel que $u \sim \mathcal{U}(0, 1)$
 - Si $u \leq \alpha$ alors : $p_{t+1} := p$
 - Sinon : on conserve l'état précédent $p_{t+1} := p_t$
3. La séquence $\{p_0, p_1, \dots, p_{N-1}\}$ constitue donc l'échantillon obtenu à partir de la chaîne de Markov associée à la loi π

On peut retirer de la séquence $\{p_0, p_1, \dots, p_{N-1}\}$ le meilleur résultat en prenant la $p_m \in E : D(p_m) = \min(D(E))$

3.7 Comparaison avec d'autres méthodes d'échantillonnage

TODO

4 Conclusion

TODO

5 Annexe

5.1 Code source

Cette section contient des implémentations des algorithmes abordés. Le code source complet est disponible à l'adresse suivante : <https://github.com/CorentinVaillant/ProjetMath-HastingMetropolis>

5.1.1 Code utilitaires

Voici le code source de 'utilis.py' qui contient des fonctions utiles au reste du projet. Seul le code des utilitaires mathématiques est présent ici et les fonctions utilitaires liées aux graphiques sont disponibles directement dans le d'application.

```
import numpy as np
import networkx as nx
import matplotlib.pyplot as plt

from math import factorial, comb

def poisson(l):
    return lambda k: ((1**k)/(factorial(k)))*np.exp(-1)

def binomial(n, p):
    return lambda k: comb(n, k) * p**k * (1-p)**(n-k)

def geo(p):
    return lambda k: p * ((1-p)**(k-1))
```

5.1.2 Implémentation de l'algorithme de Hastings-Metropolis

L'implémentation de l'algorithme de Hastings-Metropolis, comme décrit en 3.2 et relativement idiomatique, voici son implémentation, en différentes fonctions :

La fonction `next_state_markov(x, transitions)` renvoie un état y (un entier), avec une probabilité donnée par $g_x(y)$:

```
def next_state_markov(x: int, transitions: np.array) -> int:
    return np.random.choice(len(transitions), p=transitions[x])
```

La fonction `markov_chain_from_tirage(tirage, nb_etats)` transforme un tirage en une matrice de transition, représentant une chaîne de Markov :

```
def markov_chain_from_tirage(tirage, nb_etats) -> np.array:
    sommes = np.full((nb_etats, nb_etats), 0)
    derniere_etat = tirage[0]
    for col in tirage[1:]:
        sommes[derniere_etat][col] += 1
        derniere_etat = col
    result = []
    for i in range(nb_etats):
        denominateur = sum(sommes[i])
        result.append([
            sommes[i][j] / denominateur if denominateur != 0 else 0
            for j in range(nb_etats)
        ])
    return np.array(result)
```

La fonction `hasting_metropolis_tirage(x0, distribution_cible, proposition, iterations)` renvoie un tirage en suivant l'algorithme de Hastings-Metropolis :

```
def hasting_metropolis_tirage(x0: int, distribution_cible,
                             proposition: np.array, iterations
                             : int = 1000) -> np.array:

    x_t = x0
    tirage = [x_t]

    for _ in range(iterations):
        x = next_state_markov(x_t, proposition)
        pi_xt = distribution_cible(x_t)
        pi_x = distribution_cible(x)
        g_xt_x = proposition[x_t][x]
        g_x_xt = proposition[x][x_t]

        if pi_xt * g_xt_x == 0:
            alpha = 1 # Évite la division par zero
        else:
            alpha = min(1, (pi_x * g_x_xt) / (pi_xt * g_xt_x))

        u = np.random.uniform(0, 1)

        if u <= alpha:
            x_t = x
        tirage.append(x_t)
```

```
return np.array(tirage)
```

Enfin, la fonction `hasting_metropolis` combine tout pour générer directement une chaîne de Markov à partir de l'algorithme :

```
def hasting_metropolis(x0: int, distribution_cible,
                      proposition: np.array, iterations: int =
                                                                    1000) ->
                                                                    np.array:
    tirage = hasting_metropolis_tirage(x0, distribution_cible,
                                       proposition, iterations)
    return markov_chain_from_tirage(tirage, proposition.shape[0])
```

5.1.3 Applications du code d'Hasting Metropolis

Dans cette section vous trouverez des exemples d'application des différents algorithmes implémentés

Exemples Hastings-Metropolis Voici les exemples d'application d'Hasting-Metropolis que nous avons effectués, à partir de la librairie matplotlib, le code source est trouvable sur <https://github.com/CorentinVaillant/ProjetMath-HastingMetropolis>.

Application d'une loi binomiale :

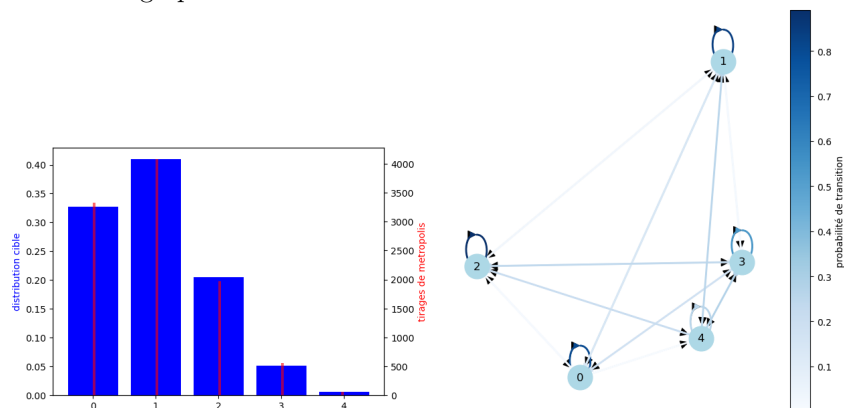
```
N = 5

distribution_cible = binomial(N,0.2)

#matrice de transition quelconque
proposition = np.array([ ligne / ligne.sum()
                        for ligne in [np.random.rand(N) for _ in range(N)]
])

tirage = hasting_metropolis_tirage(0,distribution_cible,proposition
                                   ,iterations=10_000)
mk_c = markov_chain_from_tirage(tirage,N)
```

Résultat et graphe de la matrice de transition :



Application À une loi de Poisson :

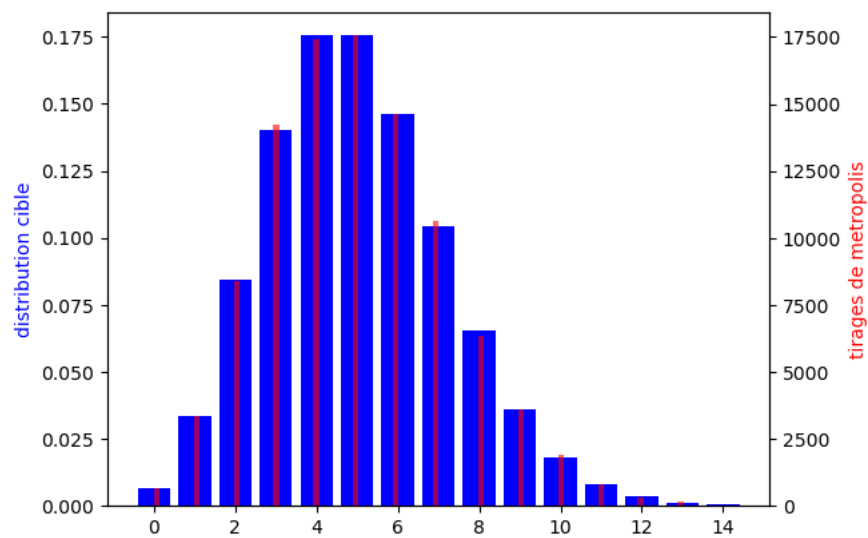
```
N = 15

distribution_cible = poisson(5)

#matrice de transition uniforme de taille NxN
proposition = np.full((N,N),1/N)

tirage = hasting_metropolis_tirage(0,distribution_cible,proposition
                                   ,iterations=100_000)
mk_c = markov_chain_from_tirage(tirage,N)
```

résultats :



Application À une loi uniform :

```

N = 15

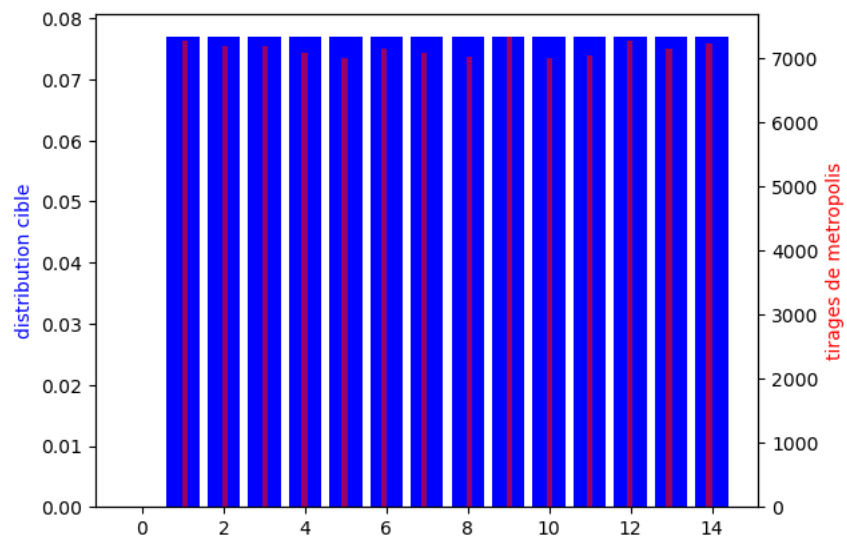
distribution_cible = lambda k : 1/(N-2) if 0<k and k<N else 0

#matrice de transition aleatoire
proposition = np.array([ ligne / ligne.sum()
                        for ligne in [np.random.rand(N) for _ in range(N)]
])

tirage = hasting_metropolis_tirage(0,distribution_cible,proposition
                                   ,iterations=100_000)
mk_c = markov_chain_from_tirage(tirage,N)

```

résultats :



Application d'une loi géométrique :

```

N = 20

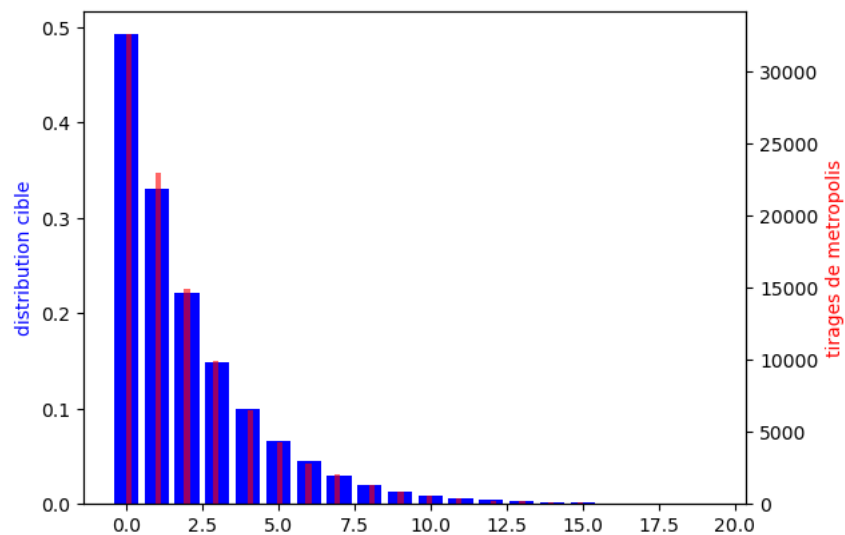
distribution_cible = geo(0.33)

# matrice de transition aleatoire
proposition = np.array([ ligne / ligne.sum()
                        for ligne in [np.array([abs(np.random.normal(i,1)) for _ in
                                                range(N)]) for i in range(N)]
])

tirage = hasting_metropolis_tirage(0,distribution_cible,proposition
                                   ,iterations=100_000)
mk_c = markov_chain_from_tirage(tirage,N)

```

résultats :



Application Ã une loi quelconque :

```

N = 30

distribution = np.random.rand(N)
distribution = distribution / distribution.sum()

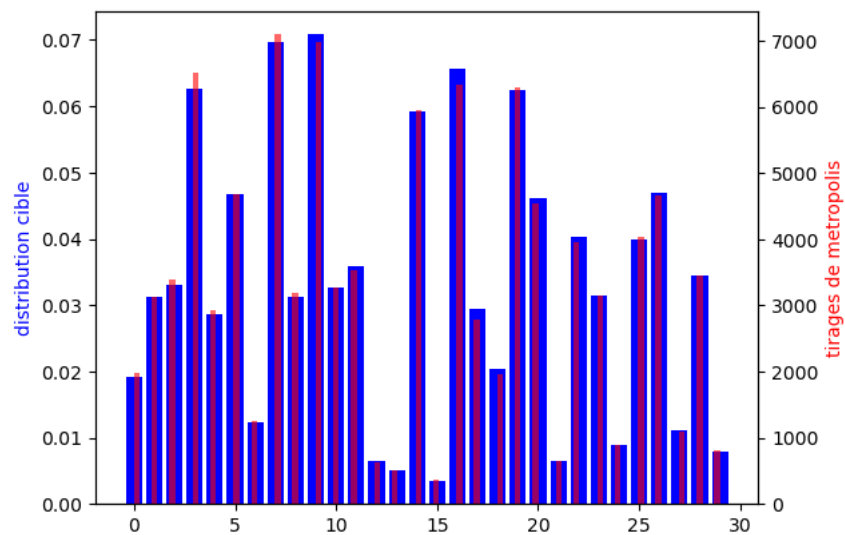
distribution_cible = lambda k : distribution[k] if k < N and 0<=k
                        else 0

#matrice de transition aleatoire
proposition = np.array([ ligne / ligne.sum()
                        for ligne in [np.array([abs(np.random.normal(i,1)) for _ in
                                                range(N)]) for i in range(N)]
])

tirage = hasting_metropolis_tirage(0,distribution_cible,proposition
                                ,iterations=100_000)
mk_c = markov_chain_from_tirage(tirage,N)

```

résultats :



5.2 Sources

Documents textuels :

- https://fr.wikipedia.org/wiki/Chaîne_de_Markov
- https://dms.umontreal.ca/~bedard/BergeronL_rapport_final.pdf
- https://www.math.univ-paris13.fr/~tourner/fichiers/agreg/2014/cours_markov.pdf
- <https://www.math.u-bordeaux.fr/~mchabano/Agreg/ProbaAgreg1213-COURS5-CM.pdf>
- <https://www.imo.universite-paris-saclay.fr/~pierre-loic.meliot/agreg/markov.pdf>
- <https://www.college-de-france.fr/fr/agenda/cours/apprentissage-et-generation-par-echantillonnage-aleatoire/algorithmes-de-metropolis-hasting>
- <https://www.math.u-bordeaux.fr/~mibonnef/mimse-markov/recurrence-transience.pdf>
- <https://perso.univ-rennes1.fr/jean-christophe.breton/agreg/AGREG/COURS/ch-mark2.pdf>
- https://fr.wikipedia.org/wiki/Processus_stochastique
- https://fr.wikipedia.org/wiki/Méthode_de_Monte-Carlo_par_chaînes_de_Markov
- <https://www.math.u-bordeaux.fr/~mchabano/Agreg/ProbaAgreg1314-COURS5-CM.pdf>
- https://fr.wikipedia.org/wiki/Algorithme_de_Metropolis-Hastings#cite_note-2
- <https://www.radcliffe.harvard.edu/news-and-ideas/flash-of-genius>
- https://fr.wikipedia.org/wiki/Problème_du_voyageur_de_commerce
- <https://www.cambridge.org/core/journals/mathematical-proceedings-of-the-cambridge-philosophical-society/article/abs/shortest-path-through-many-points/F1C28B5730B94887F4659FCBF8A1F2BB>

Documents vidéos :

- <https://www.youtube.com/watch?v=yCv2N7wGDCw>
- https://www.youtube.com/watch?v=MxI78mpq_44
- <https://www.youtube.com/watch?v=e0ZHDk4DSEI&list=PLWoShwK0FEjovcc32x9LbpDTf8pquPimV>
- <https://www.youtube.com/playlist?list=PLWoShwK0FEjovcc32x9LbpDTf8pquPimV>