Chaînes de Markov

Merwan Achibet
Université du Havre

1 Concept

1.1 Généralités

Une chaîne de Markov est un processus aléatoire à états pour lequel la propriété de Markov est vérifiée.

1.1.1 Processus aléatoire à états

Un processus aléatoire à états prend la forme d'un vecteur X représentant une liste d'états parcourus, chaque état possible appartenant à un ensemble d'états $S = \{x_0, \ldots, x_n\}$. À intervalle régulier (on parle alors de processus temporellement homogène), on peut soit rester dans le même état soit passer à un nouvel état.

Un exemple classique pour illustrer le principe des chaînes de Markov est celui de la marche aléatoire [Oco06] : on veut modéliser discrètement le déplacement d'un individu dans un environnement à une dimension, typiquement de gauche à droite. L'ensemble des états peut ici correspondre à l'indice de la position courante, ainsi $S = \{1, 2, 3, 4, 5\}$ pour 5 positions différentes. Prenons arbitrairement $X_0 = 2$; l'individu se trouve donc au départ sur la seconde position à partir de la gauche. À chaque étape, l'individu doit se déplacer vers une position voisine, ce qui revient à lui proposer deux choix (gauche ou droite), sauf lorsqu'il est à l'une des extrêmités de l'environnement, cas dans lequel il n'a qu'un seul choix (il va à droite s'il est à l'extrêmité gauche, et *vice versa*). Lorsque les deux choix sont possibles, la probabilité que chacun d'entre eux soit choisi est égale. La figure 1 illustre cet exemple et les probabilités de transition entre états voisins.

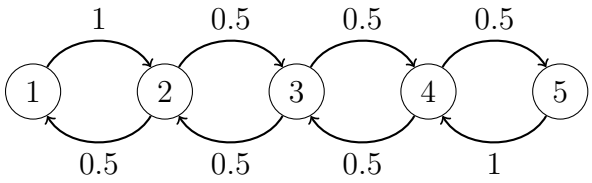

FIGURE 1 – Représentation du problème de la marche aléatoire.

Usuellement, on se base sur une matrice de transition T pour décrire les probabilités de passer d'un état précis à un autre. Ici, la matrice de transition est la suivante :

$$T = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0.5 & 0 & 0.5 & 0 & 0 \\ 0 & 0.5 & 0 & 0.5 & 0 \\ 0 & 0 & 0.5 & 0 & 0.5 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

On remarque que tous les T_{ii} avec $i \in \{1, \ldots, 5\}$ valent 0, car l'individu ne peut pas rester à la même position pendant deux étapes successives ; la probabilité de rester dans le même état est donc nulle. De plus, les probabilités de passer à un état représentant une position non voisine de la position actuelle sont elles aussi nulles, car l'individu est limité à des déplacements vers les positions adjacentes. Bien sûr, $\forall i \in \{1, \ldots, 5\}$, $\sum_{j=1}^{5} T_{ij} = 1$

À partir de T, et si l'on veut prévoir la position de l'individu à la deuxième étape (on cherche donc X_1), on remarque que seules les transitions $2 \to 1$ et $2 \to 3$ sont envisageables. Si l'on dispose d'un générateur de valeurs aléatoires, on peut alors simuler la décision de l'individu et choisir en conséquence son état à l'étape suivante. Puisque $P(X_1 = 1 | X_0 = 2) = 0.5$ et que $P(X_1 = 3 | X_0 = 2) = 0.5$, l'individu a une chance sur deux d'aller vers la position 1 et une chance sur deux d'aller vers la position 3.

1.1.2 Propriété de Markov

Maintenant que la notion de processus aléatoire à états est présentée, il faut préciser la propriété de Markov à laquelle une chaîne de Markov se doit d'obéir.

Concrètement, cette propriété énonce que, dans le cadre d'un processus aléatoire à états, l'état futur dépend uniquement de l'état présent et non de tous les états précédents.

Plus formellement, si le processus en question a atteint sa $t^{\text{ème}}$ étape :

$$P(X_{t+1}|X_t, X_{t-1}, \ldots, X_0) = P(X_{t+1}|X_t)$$

Autrement dit, seul l'état courant du système importe. Cette propriété est intéressante, car dans l'exemple de la marche aléatoire, un processus aléatoire à états basique peut se servir de la matrice de transition pour déterminer X_1 à partir de X_0 alors qu'une chaîne de Markov peut s'en servir pour déterminer X_{t+1} à partir de n'importe quel X_t.

1.2 Interêts pour la modélisation

Un système peut être modélisé par un large choix de formalismes. Parmi les modélisations envisageables, on pense notamment à la modélisation individu-centrée, ou bien à l'application de lois logistiques ou à l'utilisation de systèmes d'équations. Un avantage des chaînes de Markov pour réprésenter un problème est l'éventail de possibilités de quantification et d'analyse qu'elles offrent [IIGS09].

Contrairement à une modélisation individu-centrée, de laquelle on ne peut extraire des analyses qu'à partir des résultats opérationnels, plusieurs aspects d'un modèle de type chaîne de Markov peuvent être évalués avant toute éxecution à partir de la matrice de transition. On peut évidemment prévoir l'état suivant d'une chaîne mais aussi faire des prévisions sur l'état qui sera atteint au bout de n étapes. Il est de plus possible d'étudier la périodicité d'une chaîne (si le modèle admet des cycles dans sa progression) et sa récurrence (le nombre de fois qu'un état particulier est pris). Nous présentons par la suite les méthodes de prévision ainsi qu'une classe particulière de chaînes de Markov, les chaînes absorbantes.

1.2.1 Prévision

Comme énoncé précédemment, la propriété de Markov permet de prévoir l'état suivant d'une chaîne, quel que soit le numéro de l'étape actuelle. Autrement dit, que l'on en soit à l'étape initiale ou cent étapes plus tard, on peut évaluer la possibilité de prendre un certain état à l'étape suivante.

En conséquence, il est aussi possible de calculer la possibilité d'atteindre un état j à partir d'un état i au bout d'un certain nombre d'étapes. Reprenons l'exemple de la marche aléatoire, pour lequel $X_0 = 2$. On veut déterminer la probabilité que l'individu soit encore à la position 2 à la $3^{\text{ème}}$ étape, soit deux étapes plus tard. Autrement dit, on veut calculer $P(X_2 = 2|X_0 = 2)$. On note cette probabilité $p_{2,2}^{(2)}$ [GRL02].

On peut représenter conceptuellement toutes les issues possibles de cette situation par un arbre dans lequel chaque niveau correpond à une étape (figure 2). On remarque que dans deux cas sur trois, on atteint l'état 2 à partir de l'état 2.

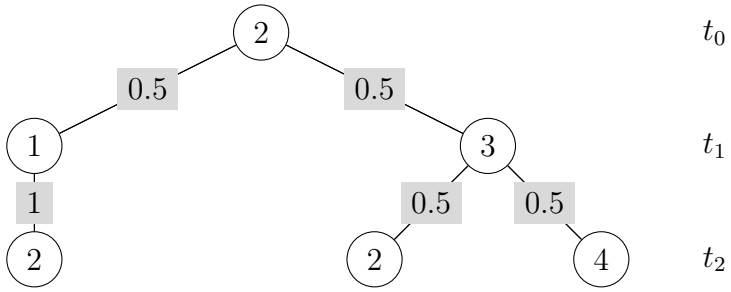

FIGURE 2 – Arbre représentant les états atteignables après deux étapes et en partant de la position 2.

Pour calculer manuellement $p_{2,2}^{(2)}$, on évalue la probabilité de chaque branche valide par rapport à toutes les issues possibles :

$$p_{2,2}^{(2)} = \frac{p_{2,1}p_{1,2} + p_{2,3}p_{3,2}}{p_{2,1}p_{1,2} + p_{2,3}p_{3,2} + p_{2,3}p_{3,4}} = \frac{0.5 \times 1 + 0.5 \times 0.5}{0.5 \times 1 + 0.5 \times 0.5 + 0.5 \times 0.5} = 0.75$$

Ce calcul, ici simple, se révélera plus fastidieux lorsque la différence temporelle entre les deux étapes considérées s'agrandira, car le nombre de feuilles de l'arbre augmentera exponentiellement.

Heureusement, ce même résultat peut être obtenu en exploitant la matrice de transition. Plus généralement, la probabilité d'être en l'état j à partir de l'état i après n étape est $p_{ij}^{(n)} = T_{ij}^n$ [GS06].

$$T^2 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{pmatrix} 0.5 & 0 & 0.5 & 0 & 0 \\ 0 & 0.75 & 0 & 0.25 & 0 \\ 0.25 & 0 & 0.5 & 0 & 0.25 \\ 0 & 0.25 & 0 & 0.75 & 0 \\ 0 & 0 & 0.5 & 0 & 0.5 \end{pmatrix} \end{array}$$

Dans ce cas, on retrouve bien $T^2_{2,2} = 0.75$. On note que cette opération fournit les probabilités de transition projetées vers tous les états et à partir de tous les états, contrairement au calcul manuel utilisé plus tôt.

1.2.2 Absorption

Parmi les différentes classes de chaînes de Markov, on trouve les chaînes absorbantes. Cette variante permet des facilités d'analyse si le modèle associé est adéquat.

Un état absorbant est un état duquel on ne peut pas sortir une fois atteint. Une chaîne de Markov absorbante est une chaîne de Markov possédant au moins un état absorbant.

[GS06] utilise une variante de la marche aléatoire pour illustrer le principe des chaînes de Markov absorbantes : l'individu a un peu trop bu et la position 1 correspond à son domicile tandis que la position 5 correspond à un bar. S'il atteint un de ces états, il n'en sort plus, ce sont les états absorbants. Lorsqu'un état absorbant est atteint, on dit que la chaîne est absorbée.

La nouvelle matrice de transition est :

$$T = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0.5 & 0 & 0.5 & 0 & 0 \\ 0 & 0.5 & 0 & 0.5 & 0 \\ 0 & 0 & 0.5 & 0 & 0.5 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

Grâce au principe des chaînes de Markov et à des opérations de base sur la matrice de transition T, si l'on dispose d'une chaîne absorbante, on peut répondre à des questions telles que :

1. Quelle est la probabilité pour que l'individu finisse par être bloqué ?
2. Au bout de combien de déplacements l'individu sera-t-il bloqué ?
3. Combien de fois l'individu va-t-il passer par chaque position ?

Pour répondre à ces questions, il est nécessaire de passer la matrice de transition sous forme canonique. Ce passage s'opère par permutation des lignes et des colonnes de la matrice de transition de façon à ce que les entrées des états absorbants valant 1 forment une matrice identité de taille $n \times n$ avec n le nombre d'états absorbants.

$$T = \begin{array}{c} \\ 2 \\ 3 \\ 4 \\ 1 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 2 & 3 & 4 & 1 & 5 \end{array} \\ \begin{pmatrix} 0 & 0.5 & 0 & 0.5 & 0 \\ 0.5 & 0 & 0.5 & 0 & 0 \\ 0 & 0.5 & 0 & 0 & 0.5 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array} = \begin{pmatrix} Q & * \\ 0 & I \end{pmatrix}$$

On décompose donc T en :

$$Q = \begin{pmatrix} 0 & 0.5 & 0 \\ 0.5 & 0 & 0.5 \\ 0 & 0.5 & 0 \end{pmatrix} \qquad I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Si l'on souhaite par exemple estimer le nombre moyen de passages par chaque état avant l'absorption de la chaîne, on calcule la matrice fondamentale N telle que :

$$N = (I - Q)^{-1}$$

Chaque ligne de N correspond au nombre de passage attendu par chaque autre état en ayant pris l'état de la ligne comme point de départ. Dans le cas de la marche aléatoire alcoolisée, on prévoit que l'individu commençant son trajet à la position 3 passera en moyenne une fois sur la position 2 avant absorption.

$$N = \begin{array}{c} \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{ccc} 2 & 3 & 4 \end{array} \\ \begin{pmatrix} 1.5 & 1 & 0.5 \\ 1 & 2 & 1 \\ 0.5 & 1 & 1.5 \end{pmatrix} \end{array}$$

2 Application : D'Artémis à Zeus

Les chaînes de Markov sont couramment utilisées pour générer du texte de façon aléatoire [Sha48], ou même de la musique [Con03]. En considérant qu'un état est un symbole (lettre, syllabe, mot ou phrase) et que la succession d'états parcourus par la chaîne forme une composition de symboles, on peut obtenir des textes donnant une impression d'authenticité. Pour obtenir un résultat satisfaisant, il sera bien sûr nécessaire d'utiliser une matrice de transition adaptée. Typiquement, elle sera calculée à partir de l'étude statistique d'un corpus contenant les textes à imiter.

Nous allons implémenter un générateur de noms inspiré de ceux des protagonistes de la mythologie grecque. Il prendra en entrée un corpus de noms grecs et générera un nouveau nom adoptant les mêmes caractéristique structurelles.

2.1 Modèle

Dans le cadre de cet exemple, la chaîne de Markov correspond à un nom, soit un mot. Un mot est composé de lettres, chacune correspondant à un état. Ici, l'ensemble des états est donc $S = \{a, b, c, \ldots, z, \xi\}$, avec ξ le symbole de fin de mot.

L'état initial X_0 sera tiré au hasard parmi cet alphabet, en ignorant ξ. On passe ensuite d'état en état pour construire progressivement le mot, jusqu'à ce que l'on atteigne ξ. Cet état fait ici office d'état absorbant puisqu'on ne peut plus en sortir une fois qu'il est atteint. Ce symbole de fin de mot est nécessaire à la génération de noms convaincants puisque que c'est le passage vers l'état le représentant qui déterminera l'arrêt de la construction et donc la longueur du mot.

L'opération se déroulera en deux phases principales. Tout d'abord, la matrice de transition T, qui nous permet de déterminer quel parcours suivre, doit être calculée. Pour cela, on se basera sur un corpus de noms de divinités

grecques que nous allons parcourir afin de relever les relations de succession entre les différentes lettres de notre alphabet. Une fois T obtenue, il reste à construire la chaîne de Markov de façon classique. La succession d'états parcourus formera le nouveau nom.

2.2 Implémentation

Cette implémentation a été réalisée en Javascript [1]. Le corpus utilisé est une sélection de noms issus de `http://en.wikipedia.org/wiki/List_of_Greek_mythological_figures`

Le corpus, contenant 45 noms grecs, est fourni à notre programme. Le processus d'apprentissage comptabilise les relations de succession entre chaque symbole et permet par la suite la génération de noms structurellement équivalents. Plus la taille du corpus est élevée et plus les noms produits seront convaincants car le poids des relations entre les symboles formant des syllabes caractéristiques en seront renforcés.

```
var corpus = [
  'aergia',
  'aether',
  'agrius',
  'alcyoneus',
  'aphrodite',
  'apollo',
  'ares',
  'artemis',
  'asteria',
  'athena',
  'chronos',
  'clymene',
  'cronus',
  'demether',
  ...
]
```

On note que le symbole de fin de mot n'est pas explicitement précisé dans le corpus mais *demether* est bien égal à *demether*ξ.

1. Sources consultables sur `http://github.com/merwaaan/papers/blob/master/markov/markov.js`

On commence par parcourir chacune des entrées de ce dictionnaire pour compter les relations de succession entre lettres voisines. On dispose au départ d'une matrice de dimensions 27×27 (26 lettres et le symbole de fin de mot) dont toutes les cases sont nulles. À chaque succession de la lettre i par la lettre j, on incrémente T_{ij}. On obtient à la fin de ce traitement une matrice équivalente à :

$$T = \begin{array}{c} \\ a \\ b \\ c \\ \vdots \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & \dots \end{array} \\ \begin{pmatrix} 0 & 1 & 3 & \dots \\ 1 & 0 & 0 & \dots \\ 4 & 0 & 0 & \dots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} \end{array}$$

Il est ensuite nécessaire d'en normaliser les valeurs afin que la somme des éléments de chaque ligne donne 1. On note cette matrice normalisée T' et on la calcule telle que :

$$\forall i, j \in S, \qquad {T'}_{ij} = \frac{T_{ij}}{\sum_{k \in S} T_{ik}}$$

On obtient au final une matrice T' équivalente à :

$$T' = \begin{array}{c} \\ a \\ b \\ c \\ \vdots \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & \dots \end{array} \\ \begin{pmatrix} 0 & 0.0153 & 0.054 & \dots \\ 0.12 & 0 & 0 & \dots \\ 0.32 & 0 & 0 & \dots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} \end{array}$$

Une fois la matrice de transition calculée, la génération se fait de façon classique. Un état initial est choisi aléatoirement dans S et on passe d'état en état jusqu'à atteindre l'état absorbant ξ. La sélection de chaque lettre est effectuée par une fonction utilisant un algorithme de roue de la fortune.

2.3 Résultats

La table 1 contient une sélection de noms générés par notre programme utilisant les chaînes de Markov. On note que les majuscules ont été ajoutées en fin de traitement à des fins de présentation car nos états représentent tous des lettres en minuscule.

Les résultats sont satisfaisants et on s'imagine facilement les épopées de Tallonyx, Erapithes et Mitheron.

Diaetheron	Therisethe	Temes
Omeneus	Iophemion	Usthypios
Crymiarene	Phyostalal	Erapithes
Arito	Zethetus	Anurysthes
Nepadion	Tallonyx	Bolemeprio
Hopheleus	Onthonophe	Phaleria
Bonus	Stionys	Mitheron

TABLE 1 – Liste de noms générés et convaincants.

Cependant d'autres résultats semblent moins réalistes et on se rend compte des faiblesses des chaînes de markov en observant le tableau 2. On y retrouve des noms trop courts, trop longs, répétitifs ou bien difficilement prononçables.

R	La	Chrthellyx
Pen	Mememememese	Bonysteidianataeronetisus
Ymiodiollclllchete	Crme	X

TABLE 2 – Liste de noms générés moins convaincants.

Les longueurs innapropriées de certains noms sont inévitables à moins d'ajouter des limites minimale et maximale dures à notre programme. Ce problème est en fait lié à la propriété de Markov : puisqu'une chaîne ne se soucie que de l'état présent, elle n'a aucune connaissance des états précédents. En conséquence, nous ne disposons pas d'une vue d'ensemble du nom pendant sa construction et il peut finir soit très tôt soit très tard.

Les noms composées d'une unique lettre peuvent découler du problème décrit précedemment mais aussi d'un corpus trop réduit. On obtient par exemple le nom X à cause du fait que le symbole x est uniquement présent dans l'entrée *styx* du corpus. Ainsi $T'_{x\xi} = 1$ et si le premier état choisi

aléatoirement est x, le mot est déjà terminé ! pour réduire les occurrences de ce problème, il faudra agrandir le corpus afin d'éviter les caractères disposant de peu de successeurs.

Autre soucis, certains noms sont imprononçables. En effet, la propriété de Markov pose encore une limite puisque la chaîne ne considère que les lettres voisines et non les syllabes dans leur ensemble. Pour résoudre ce problème, on pourrait imaginer utiliser des syllabes comme état plutôt que des lettres. L'isolation de syllabes est une tâche complexe mais on pourrait, plus généralement, utiliser des blocs de lettres de longueur n comme état. De cette façon, on réduirait le nombre de syllabes incohérentes. Un corpus beaucoup plus important serait néanmoins nécessaire pour contrebalancer le fait que le nombre d'états augmenterait considérablement.

Références

[Con03] Darrell Conklin. Music generation from statistical models. City University London, 2003.

[GRL02] Raymond N. Greenwell, Nathan P. Ritchey, and Margaret L. Lial. *Calculus with Applications for the Life Sciences*. 2002.

[GS06] C. Grinstead and J. Snell. *Introduction to Probability*. American Mathematical Society, 2006.

[IIGS09] Luis R. Izquierdo, Segismundo S. Izquierdo, José Manuel Galán, and José Ignacio Santos. Techniques to understand computer simulations : Markov chain analysis. `http://jasss.soc.surrey.ac.uk/12/1/6.html`, 2009.

[Oco06] Daniel Ocone. *Discrete and Probabilistic Models in Biology*. University of New Brunswick, 2006.

[Sha48] C. E. Shannon. A mathematical model of communication. *The Bell System Technical Journal*, 27, 1948.