Questions ouvertes / 4 Décembre 2024

Probabilités : Chaînes de MARKOV

Bilan

Question ouverte 1

A quoi reconnait-on qu'une matrice carrée M est une matrice de transition de chaine de MARKOV ?
Que peut-on dire d'assez évident quant au spectre d'une telle matrice ?
Justifier de l'existence d'un vecteur U tel que $M.U = U$.

La matrice carrée $M = (M_{ij})_{1\leq i,j,\leq n}$ est une matrice de transition de chaîne de MARKOV au sens du cours si et seulement si chacune de ses colonnes est un vecteur de probabilité, c'est à dire si et seulement si :

(i) Toutes ses entrées sont positives : $\forall (i, j) \in (\{1,\ldots,n\})^2, M_{ij} \geq 0$

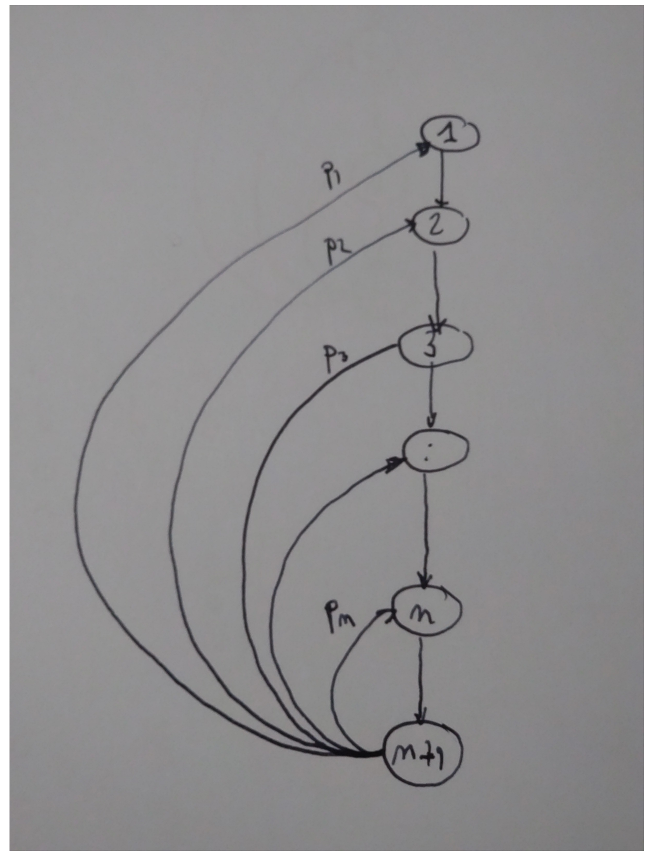
(ii) La somme de chacune de ses colonnes vaut 1 : $\forall j \in \{1,\ldots,n\}\,, \sum_{i=1}^{n} M_{ij} = 1$

A propos des valeurs propres, la condition (ii) montrée ci-dessus montre que $\mathbb{1} = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$ est vecteur propre de $M^\top$ associé à la v.p. 1, comme les valeurs propres d'une matrices et celles sont de sa transpoée sont identiques, il vient que 1 est valeur propre de M.

NB : Il existe donc un vecteur propre U tel que $M.U = U$. On peut montrer qu'un de ces vecteurs propres à des composantes toutes positives et on peut « normaliser » le vecteur de sorte à obtenir $P = (p_i)_{1\leq i\leq n}$ vérifiant $\sum_m ip_i = 1$ et $M.P = P$.
Un tel vecteur de probabilité P, vérifiant $M. = P$ est appelé une *distribution invariante* pour la chaîne définie par la matrice de transition M.
Si à un instant donné (par exemple l'initial), la configuration (un numéro entre 1 et n) est distribuée suivant la loi P, alors, à l'instant suivant, la configuration est distribuée suivant la même configuration.
On verra que dans beaucoup de cas, partant d'une distribution initiale quelconque P_0, la suite des distributions donnée par la chaîne, vérifiant la récurrence $\forall n$, $P_{n+1} = M.P_n$, converge vers une/la distribution invariante.
Cette convergence est assez rapide (exponentielle) est après une durée– dite de relaxation– le système est quasiment dans l'état stable décrit par la configuration invariante P. Ceci est du au fait que les autres valeurs propres de M sont en module < 1.)

Question ouverte 2

On considère une fontaine verticale (du genre de celle sur la photo) où le trajet d'une goutte suit le schéma (on a découpé la gamme de hauteur possibles en petits compartiments, la case $n+1$ représente la base de la fontaine, dès que la goutte y arrive, elle est renvoyée en hauteur).



Modéliser l'évolution de la position d'une goutte d'eau à l'aide d'une chaîne de MARKOV.

Est-ce que si la distribution de « renvoi de la goutte » $(p_1, \dots, p_n)$ est la distribution uniforme alors les gouttes d'eau sont uniformément réparties sur toute la hauteur ?

Peut on choisir une distribution « renvoi de la goutte » $(p_1, \dots, p_n)$ pour que les gouttes d'eau soient uniformément réparties sur toute la hauteur ?

En appliquant la technique de construction de matrice de transition vue en cours, on obtient que :

La matrice de transition, de taille $(n+1) \times (n+1)$ est

$$M = \begin{pmatrix} 0 & & \dots & 0 & p_1 \\ 1 & \ddots & & \vdots & \vdots \\ 0 & \ddots & \ddots & 0 & \vdots \\ \vdots & \ddots & \ddots & 0 & p_n \\ 0 & \dots & \ddots & 1 & 0 \end{pmatrix}$$

On suppose acquis le fait qu'il y ait une distribution invariante. V tq

$$M.V = V$$

La question qui suit, c'est que vaut V si $p_1 = \cdots = p_n = \frac{1}{n}$.

On voit que $p_1 v_{n+1} = v_1$, puis que $v_1 + p_2.v_{n+1} = v_2$,.., $v_{n-1} + p_n.v_{n+1} = v_n$ et enfin, $v_n = v_{n+1}$.

En regroupant tout cela, on voit $v_k = \frac{k}{n}.v_{n+1}$ pour $0 \leq k \leq n$ et que comme $\sum_{k=1}^{n+1} v_k = 1$, on doit avoir

$$(\frac{n(n+1)}{2n} + 1)v_{n+1} = 1$$

c'est à dire

$$v_{n+1} = \frac{2}{n+3}$$

et

$$\forall k,\ 1 \leq k \leq n \Rightarrow v_k = \frac{2k}{n(n+3)}$$

.

Pour avoir une distribution uniforme sur les hauteurs, il faudrait $v_1 = \cdots = v_n = p$ (et $v_{n+1} = v_n = p$) avce donc $p = \frac{1}{n+1}$. Les relations précédentes (c'est à dire $M.V = V$, forcent $p_1 = 1$ et $p_2 = \cdots = p_n = 0$. En clair, pour avoir une distribution uniforme, il faut remettre tout ce qui arrive en bas exactement en haut.

Question ouverte 3

A la roulette du casino, j'arrive avec 1000€en poche , à chaque tour de jeu, je mise 10€sur la « chance simple » : Rouge. Mes chances de gagner sont de 18/37. Si je perds, je perds ma mise, si je gagne, je récupère ma mise et gagne un montant égal à la mise.

Je sortirai du casino ruiné ou alors dès que j'aurai doublé mon capital inital.

Modéliser l'évolution du capital à l'aide d'une chaîne de MARKOV.

Combien de tours de jeu en moyenne ? Est-ce que ce nombre de tours de jeu est proportionnel à la mise initiale ?

Déjà, on peut changer d'unité monétaire et ramener la mise de base à 1.

On peut modéliser par une chaîne de MARKOV en prenant comme configurations les différentes valeurs possibles du capital. Si le capital est K, on construit la matrice $(2K+1) \times (2K+1)$. La ligne/colonne 0 correspond à l'état de sortie ruiné, la ligne colonne $2K$ à l'état de sortie vainqueur.

$$M = \begin{pmatrix} 1 & (1-p) & 0 & 0 & 0 & & 0 \\ 0 & 0 & (1-p) & 0 & 0 & & 0 \\ 0 & p & 0 & (1-p) & 0 & & 0 \\ 0 & 0 & p & 0 & \ddots & & \vdots \\ 0 & 0 & 0 & \ddots & \ddots & (1-p) & 0 \\ 0 & & & & p & 0 & 0 \\ 0 & & & & 0 & p & 1 \end{pmatrix}$$

A l'état initial, le joueur entre avec un capital de K, ce que l'on note $X_0 = K$. Plus précisément, si on note X_t le capital du joueur à l'instant t alors

$$U_t = (\mathbb{P}(X_t = k))_{0 \leq k \leq 2K}$$

Le vecteur de probabilité initial U_0 de longueur $2K+1$ comporte donc des zéros partout sauf un 1 en place K et l'évolution est donnée par la suite de vecteurs de probabilité $(U_t)_{t\in\mathbb{N}}$ où

$$\forall t \in \mathbb{N}, U_{t+1} = M.U_t.$$

Le script suivant simule la chaîne et permet le tracé du graphe du nombre moyen de tours en fonction du capital initial.

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

def RR(p = 18/37): #un tour de roulette, retourne 1 si rouge tiré, -1 sinon
    u = np.random.rand()
    return 2*int(u<= p) -1

def NbTours(cap = 10) :
    """ Simule la roulette avec capital initial cap jusque ruine ou doublement
    capital initial"""
    s = cap
    tour = 0
    while (s < 2*cap and s> 0) :
        tour += 1
        s += RR()
    return tour

def E_NbTours_sim(cap = 10, NS =  1000) :
    """ Evalue le nbre de tours à faire en moyenne
    en partant d'un capital de cap"""
    NT = 0
    for _ in range(NS):
        NT += NbTours(cap = cap)
    return NT/NS

#Tracé graphe NBTours en fonction cap
caps = np.arange(2, 10)
nbtours = []
for cap in caps :
    nbtours.append(E_NbTours_sim(cap = cap))
nbtours = np.array(nbtours)

plt.subplots()
plt.plot(caps,nbtours,'x')
plt.savefig('roulette-01.pdf', format = 'pdf')
plt.show()
```

On peut tenter un calcul théorique. Si T désigne l'iunstant de sortie, on a pour $t \in \mathbb{N}$, l'égalité d'événements

$$\{T > t\} = \{X_t \notin \{0, 2K\}\}$$

et donc

$$\mathbb{P}(T > t) = 1 - U_{t,0} - U_{t,2K}$$

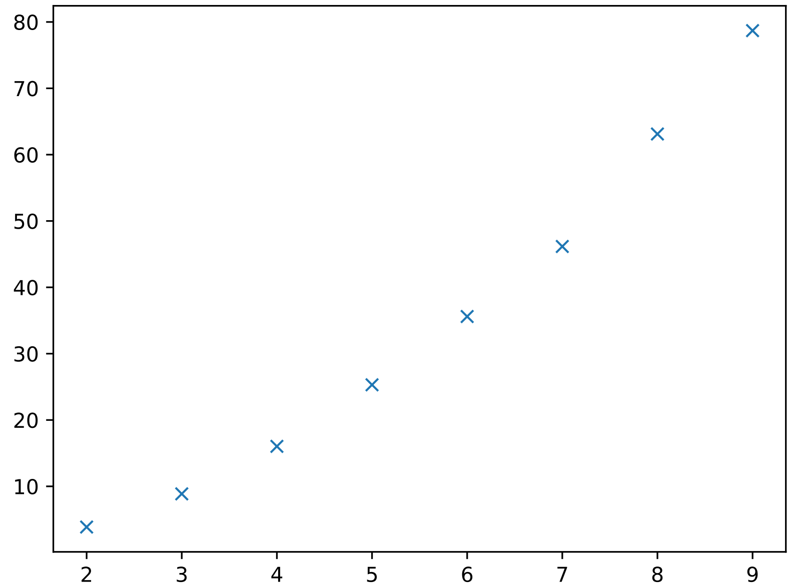


FIGURE 1

Comme (c'est une formule bien connue dans la théorie des variables aléatoires discrètes), on a

$$\mathbb{E}(T) = \sum_{t=0}^{+\infty} \mathbb{P}(T > t)$$

on dispose d'un moyen effectif de calculer l'espérance de T en fonction de K. C'est ce qui est fait dans l'extrait de script suivant (avec grpahique à la clé)

```
def Mat(K, p = 18/37):
    M = np.zeros((2*K+1,2*K+1))
    M[1:-1,1:-1] = np.diag([p]*(2*K-2),-1)
    M[1:-1,1:-1] += np.diag([1-p]*(2*K-2),1)
    M[0,0] = 1
    M[0,1] = 1 - M[2,1]
    M[-1,-2] = 1 - M[-3,-2]
    M[-1,-1] = 1
    return M

def ET(K, N = 50000) :
    M = Mat(K)
    X0 = np.zeros((2*K+1,)) ; X0[K] = 1.0
    S = 0
    for _ in range(N) :   #En fait, pour que ça ait du sens, il faut faire N = 2*K**2 tours
        S += 1- (X0[0] + X0[-1])
        X0 = M @ X0
    print("E(T) = ",S)
    return S
```

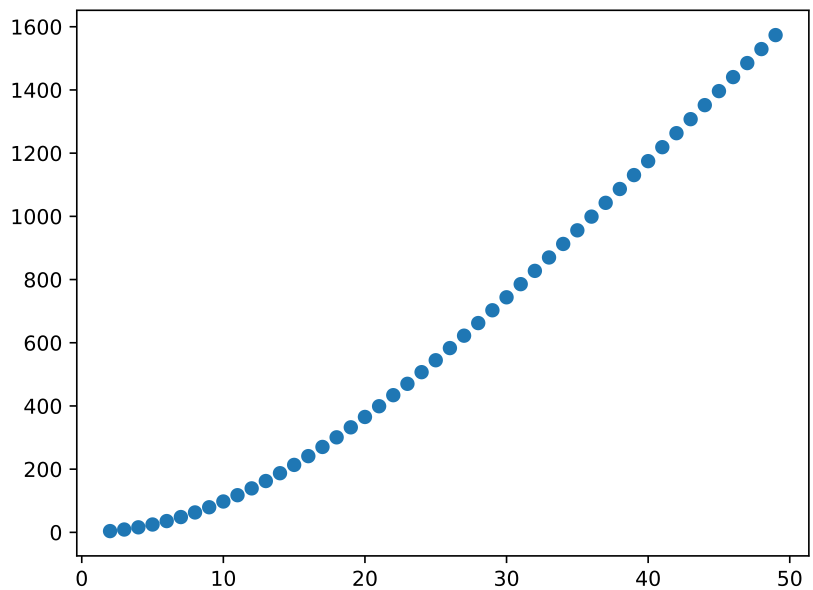


FIGURE 2

```
#Tracé graphe NBTours en fonction cap
caps = np.arange(2, 50)
nbtours = []
for cap in caps :
    nbtours.append(ET(K = cap))
nbtours = np.array(nbtours)

plt.subplots()
plt.plot(caps,nbtours,'o')
plt.savefig('roulette-02.pdf', format = 'pdf')
plt.show()
```