TP8 Python (chaînes de Markov et graphes probabilistes)

Exercice 1

Une puce se déplace sur un axe gradué en effectuant des sauts sur les points d'abscisses $\boxed{1}$, $\boxed{2}$ et $\boxed{3}$, selon les règles suivantes :

– A l'instant 0 (état initial), la puce est en $\boxed{2}$,

– Si à l'instant n, la puce est en $\boxed{1}$, alors à l'instant $n+1$, elle a une chance sur deux de rester en $\boxed{1}$ et une chance sur deux d'aller en $\boxed{2}$,

– Si à l'instant n, la puce est en $\boxed{2}$, alors à l'instant $n+1$, elle a une chance sur deux d'aller en $\boxed{1}$ et une chance sur deux d'aller en $\boxed{3}$,

– Si à l'instant n, la puce est en $\boxed{3}$, alors à l'instant $n+1$ elle va en $\boxed{2}$.

On considère la suite $(X_n)_{n\in\mathbf{N}}$ de variables aléatoires définies par :
X_n = abscisse de la puce à l'instant n. (On a alors $X_0 = 2$).

On dit que $(X_n)_{n\in\mathbf{N}}$ est une *chaîne de Markov* car la position de la puce à l'instant $n+1$ ne dépend que de sa position à l'instant n et non de sa position aux instants 0, 1, ..., $n-1$.

1)Préciser la loi de X_0 et de X_1. Reconnaître ces lois.

2)On appelle *matrice de transition* de la chaîne de Markov $(X_n)_{n\in\mathbf{N}}$, la matrice $A \in \mathscr{M}_3(\mathbf{R})$ dont le coefficient de la i-ème ligne et de la j-ème colonne vaut :

$$p_{ij} = P_{(X_n=i)}(X_{n+1} = j).$$

a)Former le *graphe probabiliste* montrant les différentes positions de la puce ainsi que les probabilités de passage d'une position à l'autre (c'est un graphe pondéré et orienté).

b)Ecrire la matrice de transition A.

c)Vérifier que ses coefficients sont positifs ou nuls et que sur chaque ligne, la somme des coefficients fait 1 (on dit que c'est une matrice *stochastique*).

3)En appliquant la formule des probabilités totales pour le système complet $(X_n = i)_{1\leq i\leq 3}$, établir que pour tout $n \in \mathbf{N}$:

$$\begin{cases} P(X_{n+1} = 1) = \dfrac{1}{2}P(X_n = 1) + \dfrac{1}{2}P(X_n = 2) \\[2ex] P(X_{n+1} = 2) = \dfrac{1}{2}P(X_n = 1) + P(X_n = 3) \\[2ex] P(X_{n+1} = 3) = \dfrac{1}{2}P(X_n = 2) \end{cases}$$

4)On appelle *n-ieme état probabiliste* de la chaîne de Markov $(X_n)_{n\in\mathbf{N}}$ la matrice ligne, notée U_n, représentant la loi de X_n et définie par :

$$U_n = \big(P(X_n = 1), P(X_n = 2), P(X_n = 3)\big).$$

a)Préciser U_0.

b)Vérifier à l'aide de 3) que $\forall n \in \mathbf{N},\ U_{n+1} = U_n A$.

5)a)Compléter le programme ci-dessous afin qu'il affiche les matrices lignes U_n pour $n \in [\![0, 30]\!]$, sous forme de tableau numpy, puis exécuter-le.

```
import numpy as np
A=np.array((................................))
U=np.array((...........))
print(U)
for k in range(1,31):
    U=.............
    print(U)
```

b)Observer les résultats obtenus.

6)Soit $U = (u_1, u_2, u_3)$ une matrice ligne définissant une loi de probabilité, c'est-à-dire vérifiant $\forall i \in [\![1, 3]\!],\ u_i \geq 0$ et $u_1 + u_2 + u_3 = 1$.
On dit que U est un *état stable* de la chaîne de Markov $(X_n)_{n\in\mathbf{N}}$ si $UA = U$.

Compte tenu des résultats obtenus dans la question 5)b), on peut conjecturer que $\lim_{n\to+\infty} U_n$ existe.

On dit alors que la suite $(X_n)_{n\in\mathbf{N}}$ *converge en loi.*

Notons $\Pi = \lim_{n\to+\infty} U_n = \left(\lim_{n\to+\infty} P(X_n = 1), \lim_{n\to+\infty} P(X_n = 2), \lim_{n\to+\infty} P(X_n = 3)\right)$.

a)Montrer que Π est une loi de probabilité.

b)Montrer que Π est un état stable de la chaîne.

c)Montrer que si $U_0 = \Pi$, alors $\forall n \in \mathbf{N},\ U_n = \Pi$.

d)En partant de l'égalité $\Pi A = \Pi$, établir que ${}^t\Pi$ est un vecteur propre de ${}^tA$ et préciser la valeur propre associée.

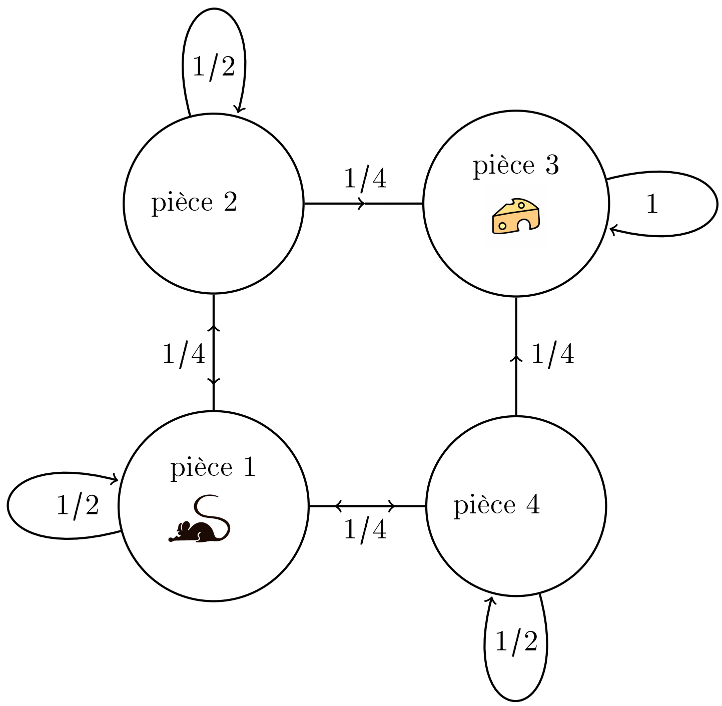
e)Déterminer le sous-espace propre de ${}^tA$ associé à 1, puis en déduire les composantes de Π[1]

1. Π s'appelle le vecteur de Perron-Frobénius. Il est utilisé par les algorithmes de Google pour calculer l'importance d'une page, appelée *PageRank*.
Pour approfondir, voir le sujet d'essec II 2008 ...

Exercice 2

Une maison comporte 4 pièces.
A l'instant 0, une souris se trouve dans la pièce 1.
A chaque minute, elle peut se déplacer d'une pièce à l'autre suivant le graphe probabiliste ci-dessous.
Dès que la souris atteint la pièce 3 où se trouve le fromage, elle y reste éternellement et manque tranquillement le fromage.

On considère la suite $(X_n)_{n\in\mathbf{N}}$ de variables aléatoires définies par :
X_n = position de la souris à la n-ième minute. (On a alors $X_0 = 1$).



1)Déterminer la matrice de transition A de la chaîne de Markov $(X_n)_{n\in\mathbf{N}}$ et vérifier qu'elle est bien stochastique.

2)En s'inspirant de la question 5)a) de l'exercice 1, écrire un programme Python qui affiche la loi de X_n pour $n \in [\![0, 100]\!]$.
Interpréter les résultats obtenus.

3)On rappelle que la fonction *rd.random()* du module *numpy.random* renvoie un réel aléatoire de $[0, 1]$.
La fonction ci-dessous prend comme paramètre une matrice ligne, intitulée « loi », formée de 4 probabilités dont la somme fait 1. Que renvoie t-elle ?

```
def choix(loi):
    a=rd.random()
    if a<loi[0]:
        return 1
    if a>loi[0] and a<loi[0]+loi[1]:
        return 2
    if a>loi[0]+loi[1] and a<loi[0]+loi[1]+loi[2]:
        return 3
    if a>1-loi[3]:
        return 4
```

4)La fonction ci-dessous prend comme paramètres :
n = nombre de minutes écoulées,
pos_depart = numéro de la pièce où se trouve la souris initialement.
Elle renvoie une liste X constituée des positions successives de la souris.

```
def markov(n,pos_depart):
    X=[0]*(n+1)
    X[0]=pos_depart
    for k in range(n):
        X[k+1]=choix(A[X[k]-1,:])
    return X
```

a)A l'aide de cette fonction, vérifier que quelle que soit sa position de départ, la souris met en général moins d'un quart d'heure à trouver le fromage.

b)Que renvoie la fonction Python ci-dessous ?

```
def fromage(pos_depart):
    X=[pos_depart]
    k=0
    while X[k]!=3:
        X.append(choix(A[X[k]-1,:]))
        k=k+1
    print(X)
    return(k)
```

c)Ecrire une fonction d'en-tête temps_moyen et de paramètre pos_depart, qui renvoie le temps moyen que met la souris à trouver le fromage lors de 1000 expériences aléatoires.
Vérifier alors qu'il faut en moyenne 6 minutes à la souris pour trouver le fromage.

Exercice 3

Thésée est enfermé dans un labyrinthe.
Pour en sortir et retrouver sa liberté, il doit recueillir 9 clés.
On suppose qu'initialement (jour 0), Thésée possède déjà 3 clés.
Chaque jour, Thésée fait un pari avec le minotaure qui lui fait gagner une clé avec la probabilité 0,6 et perdre une clé avec la probabilité 0,4.
Les paris s'enchaînent de jour en jour jusqu'à ce que Thésée possède 9 clés (il est libre et garde avec lui ses 9 clés eternellement) ou jusqu'à ce qu'il n'ait plus de clé (les paris s'arrêtent et Thésée reste alors sans clé et enfermé dans le labyrinthe éternellement).

On note $(X_n)_{n\in\mathbf{N}}$ la chaîne de Markov définie par :

$X_n =$ nombre de clés que possède Thésée au n-ième jour. (On a $X_0 = 3$).

1)Dessiner le graphe probabiliste de la chaîne de Markov $(X_n)_{n\in\mathbf{N}}$.

2)Soit $A = (p_{ij})_{0\leq i,j\leq 9}$ la matrice de transition où $p_{ij} = P_{(X_n=i)}(X_{n+1} = j)$.
(On convient que la première ligne de A est la ligne d'indice 0 et que la première colonne de A est la colonne d'indice 0).

a)Que valent $p_{0,0}$ et $p_{9,9}$?

b)Préciser $p_{i,i-1}$ et $p_{i,i+1}$ pour $i \in [\![1, 8]\!]$.

c)Préciser la valeur des nombres p_{ij} dans les autres cas.

d)Compléter le programme suivant de sorte qu'il affiche la matrice A.

```
import numpy as np
A=np.zeros(shape=(10,10))
A[0,0]=....
A[9,9]=....
for i in range(1,9):
    A[i,i-1]=....
    A[i,i+1]=....
print(A)
```

3)En s'inspirant de la question 5)a) de l'exercice 1, écrire un programme Python qui affiche la loi de X_n pour $n \in [\![0, 100]\!]$.
Interpréter les résultats obtenus.

4)a)Justifier que la suite $(X_n = 9)_{n\in\mathbf{N}}$ est une suite croissante d'événements.

b)A l'aide du théorème de la limite monotone, montrer que la probabilité que Thésée sorte du labyrinthe est égale à $\lim_{n\to+\infty} P(X_n = 9)$.

En utilisant la question 3), donner une valeur approchée de cette probabilité.