TP13 : Chaînes de Markov

I. Notion de chaîne de Markov : illustration sur un exemple

I.1. Énoncé du problème

Doudou le hamster passe son temps entre ses trois activités favorites : dormir, manger et faire de la roue. Au début de la journée, il mange, et à chaque heure, il change d'activité selon les critères suivants.

1) Si, à l'heure n, il est en train de manger, alors il va dormir l'heure suivante avec probabilité 0.7 et faire de l'exercice avec probabilité 0.3.

2) Si, à l'heure n, il est en train de dormir, alors il continue à dormir l'heure $n+1$ avec probabilité 0.4, il va manger avec probabilité 0.3 et il va faire de l'exercice avec probabilité 0.3.

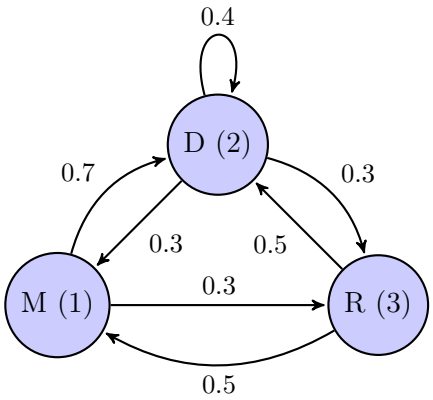
3) Si, à l'heure n, il est en train de faire de la roue, il va manger l'heure suivante avec probabilité 0.5 et il va dormir avec probabilité 0.5.

On s'intéresse ici à l'évolution du comportement de Doudou. On souhaite notamment déterminer si l'une de ses activités prendra, à terme, le dessus sur les autres.

I.2. Modélisation mathématique

On modélise ce problème comme suit.

- On commence par numéroter les activités par un entier entre 1 et 3. Plus précisément,
  - × Manger sera l'activité 1
  - × Dormir sera l'activité 2
  - × Faire de la roue sera l'activité 3
- On note X_n la v.a.r. égale à l'état du hamster à l'heure n.
  Ainsi, la suite de v.a.r. (X_n) représente l'évolution des activités du hamster.
- Cette évolution peut être modélisée par le graphe suivant.



- On définit enfin la `matrice de transition` A associée au problème. Il s'agit de la matrice :

$$A = (a_{i,j})_{\substack{1\leqslant i\leqslant 3\\ 1\leqslant j\leqslant 3}} \quad \text{où} \quad a_{i,j} = \mathbb{P}_{[X_n=i]}([X_{n+1}=j])$$

($a_{i,j}$ représente la probabilité de passage de l'état i à l'état j)

I.3. Étude de la matrice de transition

▶ Soit $(i_0, \dots, i_n, i_{n+1}) \in [\![1, 3]\!]^{n+2}$. Que vaut $\mathbb{P}_{[X_0=i_0]\cap\dots\cap[X_n=i_n]}([X_{n+1} = i_{n+1}])$?

On a :
$$\mathbb{P}_{[X_0=i_0]\cap\dots\cap[X_n=i_n]}([X_{n+1} = i_{n+1}]) = \mathbb{P}_{[X_n=i_n]}([X_{n+1} = i_{n+1}])$$
Cette propriété est appelée `propriété de Markov`.
On la retient souvent par la phrase :
Le futur (la position X_{n+1} à l'instant $n+1$) ne dépend du passé (positions X_0, …, X_n) que par le présent (la position X_n à l'instant n).

▶ Donner une expression de $\mathbb{P}_{[X_n=i]}([X_{n+1} = j])$ qui ne dépende pas de n.

$$\mathbb{P}_{[X_n=i]}([X_{n+1} = j]) = \mathbb{P}_{[X_0=i]}([X_1 = j])$$

▶ Déterminer la matrice de transition A du problème précédent.
Écrire l'appel permettant de la stocker dans une variable `A`.

$$A = \begin{pmatrix} 0 & 0.7 & 0.3 \\ 0.3 & 0.4 & 0.3 \\ 0.5 & 0.5 & 0 \end{pmatrix}$$
En **Python** : `A = np.array([[0, 0.7, 0.3], [0.3, 0.4, 0.3], [0.5, 0.5, 0 ]])`

▶ Les coefficients de la matrice A ne dépendent pas de n. Comment appelle-t-on une telle chaîne de Markov ?

Une telle chaîne de Markov est dites homogène.

Remarque

- Il est classique de faire l'étude d'une grandeur aléatoire qui varie dans le temps discret. Par exemple, on peut étudier :
  - × l'évolution du prix d'une action jour après jour,
  - × l'évolution du gain d'un joueur après chaque partie d'un jeu,
  - × le déplacement d'un mobile (d'une puce) sur les sommets d'un triangle / carré (ou sur un axe gradué),
  - × …

  Lorsque l'évolution se fait de sorte que l'état à un instant ne dépend que de l'état à l'instant précédent, on est dans le cadre des chaînes de Markov.
- Il est fréquent de voir ce type d'étude aux concours. Comme on va le voir, cela permet de faire un sujet qui mêle les résultats des chapitres Probabilités (FPT, sce) et Algèbre Linéaire (étude d'une matrice et des ses puissances).

I.4. Lien entre la matrice de transition et la loi de X_n

► Rappeler $X_n(\Omega)$.

$X_n(\Omega) = \{1, 2, 3\}$.

Dans la suite, on considère le vecteur ligne V_n qui définit la loi de X_n (n^{e} état de la chaîne) :

$$V_n \;=\; \begin{pmatrix} \mathbb{P}([X_n = 1]) & \mathbb{P}([X_n = 2]) & \mathbb{P}([X_n = 3]) \end{pmatrix}$$

► Déterminer la probabilité $\mathbb{P}([X_{n+1} = 1])$ en fonction de $\mathbb{P}([X_n = 1])$, $\mathbb{P}([X_n = 2])$ et $\mathbb{P}([X_n = 3])$ et des cœfficients de la matrice A.

La famille $([X_n = 1], [X_n = 2], [X_n = 3])$ est un système complet d'événements.
Ainsi, d'après la formule des probabilités totales :

$$\begin{aligned}
\mathbb{P}([X_{n+1} = 1]) &= \sum_{k=1}^{3} \mathbb{P}([X_n = k] \cap [X_{n+1} = 1]) \\
&= \sum_{k=1}^{3} \mathbb{P}([X_n = k]) \times \mathbb{P}_{[X_n=k]}([X_{n+1} = 1]) \\
&= \mathbb{P}([X_n = 1]) \times \mathbb{P}_{[X_n=1]}([X_{n+1} = 1]) \\
&\quad + \mathbb{P}([X_n = 2]) \times \mathbb{P}_{[X_n=2]}([X_{n+1} = 1]) \\
&\quad + \mathbb{P}([X_n = 3]) \times \mathbb{P}_{[X_n=3]}([X_{n+1} = 1]) \\
&= a_{1,1} \times \mathbb{P}([X_n = 1]) + a_{2,1} \times \mathbb{P}([X_n = 2]) + a_{3,1} \times \mathbb{P}([X_n = 3])
\end{aligned}$$

► Déterminer de même $\mathbb{P}([X_{n+1} = 2])$ et $\mathbb{P}([X_{n+1} = 3])$.

En procédant de la même manière, on obtient :

- $\mathbb{P}([X_{n+1} = 2]) \;=\; a_{1,2} \times \mathbb{P}([X_n = 1]) + a_{2,2} \times \mathbb{P}([X_n = 2]) + a_{3,2} \times \mathbb{P}([X_n = 3])$
- $\mathbb{P}([X_{n+1} = 3]) \;=\; a_{1,3} \times \mathbb{P}([X_n = 1]) + a_{2,3} \times \mathbb{P}([X_n = 2]) + a_{3,3} \times \mathbb{P}([X_n = 3])$

► En déduire que pour tout $n \in \mathbb{N}$, $V_{n+1} = V_n A$. Exprimer enfin V_n en fonction de A et de V_0.

On reconnaît dans les formules précédentes le produit matriciel $V_n A$. On en déduit que :
$\forall n \in \mathbb{N},\ V_{n+1} = V_n A$.
Par une récurrence immédiate, on obtient : $\forall n \in \mathbb{N},\ V_n = V_0 A^n$.

► Comment modéliser le fait que le hamster est initialement en train de manger ? Comment obtient-on V_n dans ce cas ?

Si le hamster est initialement dans l'état 1 (*i.e.* il mange), on a alors $\mathbb{P}([X_0 = 1]) = 1$.
D'où $V_0 = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}$.
On obtient V_n en utilisant la formule $V_n = V_0 A^n$ et en remarquant que le produit matriciel $V_0 A^n$ renvoie la première ligne de la matrice A^n.

► Rappeler la commande **Python** qui permet de calculer A^n.

`import numpy.linalg as al` puis `al.matrix_power(A,n)`

► Déterminer A^5, A^{10} et A^{20} à l'aide de **Python**. Que remarque-t-on ?

On remarque que la suite des puissances itérées (A^n) semble converger vers une matrice proche de :

$$\begin{pmatrix} 0.27 & 0.5 & 0.23 \\ 0.27 & 0.5 & 0.23 \\ 0.27 & 0.5 & 0.23 \end{pmatrix}$$

II. Simulation des trajectoires en Python

II.1. Évolution de l'état du hamster sur une heure partant d'un état x_0 fixé

Dans ce qui suit, et sauf mention du contraire, on fait l'hypothèse que le hamster se trouve initialement dans l'état $x_0 = 2$ (c'est-à-dire qu'il dort). Autrement dit : $\mathbb{P}([X_0 = 2]) = 1$.

► Que vaut V_0 dans ce cas ? Et V_1 ?

Dans ce cas, on a $V_0 = \begin{pmatrix} 0 & 1 & 0 \end{pmatrix}$ et $V_1 = V_0 A = \begin{pmatrix} 0.3 & 0.4 & 0.3 \end{pmatrix}$

► Faire le lien entre V_1 et A. En **Python**, quel appel sur A permet de récupérer V_1 ?

V_1 est la deuxième ligne de A. On récupère cette ligne à l'aide de l'appel : `A[1]` (attention, on numérote à partir de 0 en **Python**).

► Décrire le comportement que doit avoir une fonction **Python** simulant X_1.

$X_1(\Omega) = [\![1, 3]\!]$. Une fonction simulant X_1 doit renvoyer :

- 1 avec probabilité 0.3,
- 2 avec probabilité 0.4,
- 3 avec probabilité 0.3.

► Écrire, à l'aide de la fonction `rd.random()`, une fonction `SimuX1` (sans paramètre d'entrée) permettant de simuler la v.a.r. X_1 dans le cas où $x_0 = 2$.

```
def SimuX1():
    r = rd.random()
    if r < 0.3:
        return 1
    elif r < 0.7:
        return 2
    else:
        return 3
```

II.2. Évolution de l'état du hamster sur une heure partant d'un état x_0 quelconque

▶ Modifier la fonction précédente afin de l'adapter au cas où x_0 est quelconque. On notera cette nouvelle fonction `SimuMarkovEtape`. On prendra la valeur `x0` ainsi que la matrice de transition `A` en paramètres d'entrée.

```
def SimuMarkovEtape(x0, A):
    r = rd.random()
    if r < A[x0-1,0]:
        return 1
    elif r < A[x0-1,0] + A[x0-1,1]:
        return 2
    else:
        return 3
```

II.3. Évolution de l'état du hamster sur n heures

▶ Écrire une fonction `SimuMarkov(x0,A,n)` qui prend en paramètres l'état initial `x0`, la matrice de transition `A` et un entier n et renvoie une simulation de la v.a.r. X_n.

Il s'agit simplement d'itérer la fonction précédente.

```
def SimuMarkov(x0, A, n):
    x = x0
    for i in range(n):
        x = SimuMarkovEtape(x, A)
    return x
```

II.4. Simulation d'une trajectoire

- Pour $\omega \in \Omega$, on appelle **trajectoire** de taille n la suite finie $(X_0(\omega), X_1(\omega), \ldots, X_n(\omega))$.
- Simuler une trajectoire de taille n c'est donc obtenir une suite $(x_0, x_1, \ldots, x_n)$ où x_i est le résultat de la simulation de X_i.

▶ Écrire une fonction `SimuTrajectoire(x0, A, n)` qui simule une trajectoire de taille `n`. Cette fonction renverra un tableau contenant la suite finie $(x_0, x_1, \ldots, x_n)$.

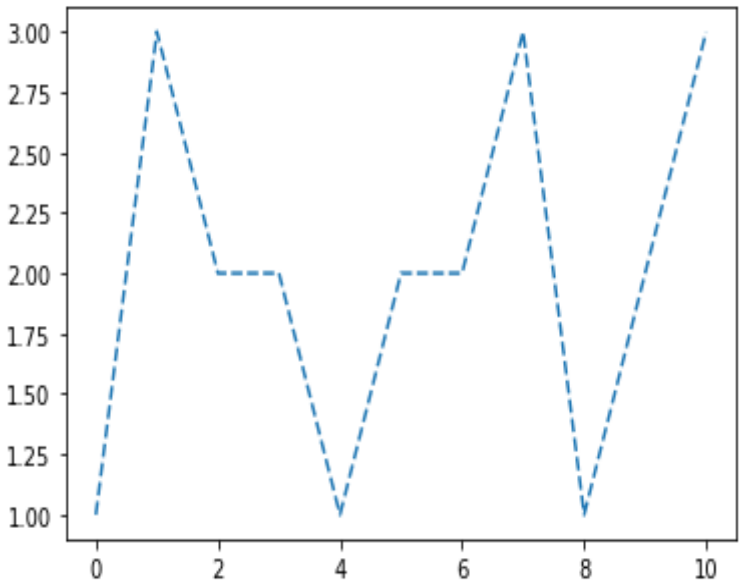
```
def SimuTrajectoire(x0, A, n):
    tab = np.zeros(n+1)
    x = x0
    tab[0] = x
    for i in range(n):
        x = SimuMarkovEtape(x, A)
        tab[i+1] = x
    return tab
```

II.5. Représentation de trajectoires

► Étant donné un état initial `x0` et une matrice de transition `A`, quel appel permet de représenter une trajectoire simulée de taille `n` ? On utilisera la fonction `plt.plot`.

```
plt.plot(range(n+1), SimuTrajectoire(x0, A, n), '--')
```

► On représente ci-dessous le tracé obtenu avec le code précédent et le choix `x0 = 1` et `n = 10`.



III. Comportement asymptotique du hamster

On souhaite conjecturer le comportement de la loi de X_n lorsque n tend vers $+\infty$. On aimerait en particulier savoir si (X_n) converge en loi vers une v.a.r. X. On se propose alors, pour un x_0 donné, de tracer les trois suites $(\mathbb{P}([X_n = k]))_{n\in\mathbb{N}}$ pour $k \in \{1, 2, 3\}$.

► Partant de l'état initial `x0`, par quel appel obtient-on V_n en supposant que la matrice A^n est stockée dans une variable `M` ?

On sélectionne la ligne correspondant à l'état `x0` : `M[x0-1]`

► Compléter le programme suivant pour qu'il renvoie la matrice

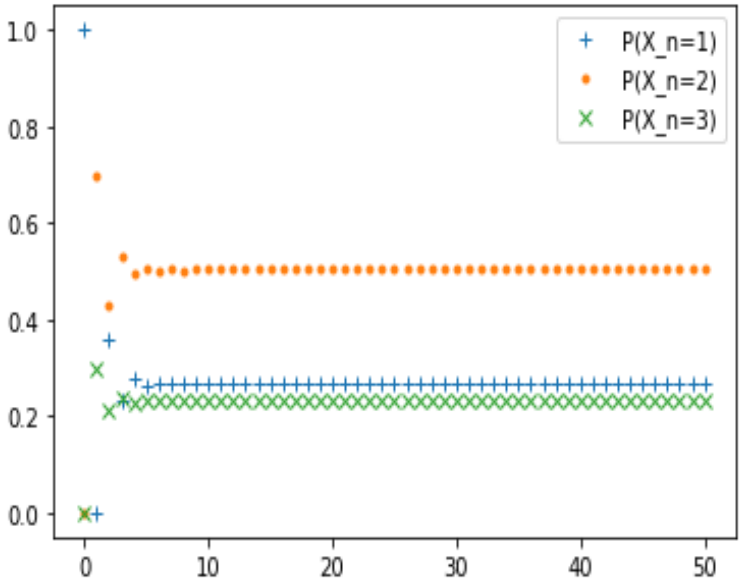
$$V = \begin{pmatrix} \mathbb{P}([X_0 = 1]) & \mathbb{P}([X_0 = 2]) & \mathbb{P}([X_0 = 3]) \\ \mathbb{P}([X_1 = 1]) & \mathbb{P}([X_1 = 2]) & \mathbb{P}([X_1 = 3]) \\ \vdots & \vdots & \vdots \\ \mathbb{P}([X_n = 1]) & \mathbb{P}([X_n = 2]) & \mathbb{P}([X_n = 3]) \end{pmatrix} \in \mathscr{M}_{n+1,3}(\mathbb{R})$$

```
1  def PremTermesLoiXn(x0, A, n):
2      V = np.zeros([n+1,3])
3      M = np.eye(3)
4      for k in range(n+1):
5          V[k] =  M[x0-1]
6          M =  np.dot(M,A)
7      return V
```

- On compile le script suivant :

```
n = 50
x0 = 1
Y = range(n+1)
V = PremTermesLoiXn(x0, A, n)
plt.plot(Y, V[:,0], '+', label='P(X_n=1)')
plt.plot(Y, V[:,1], '.', label='P(X_n=2)')
plt.plot(Y, V[:,2], 'x', label='P(X_n=3)')
plt.legend(loc='best')
```

et on obtient le tracé :



Que peut-on conjecturer ?

> On conjecture que les suites $(\mathbb{P}([X_n = k]))_{n\in\mathbb{N}}$ convergent toutes et donc que la suite (X_n) converge en loi vers une v.a.r. X.

- En admettant que (X_n) converge en loi vers une v.a.r. X qui ne dépend pas de x_0 et en notant

$$\pi = \begin{pmatrix}\mathbb{P}([X=1]) & \mathbb{P}([X=2]) & \mathbb{P}([X=3])\end{pmatrix}$$

Que peut-on dire de π ?

> Le vecteur ligne π est nécessairement un état stable de la chaîne de Markov : $\pi = \pi A$.

- On rappelle que lorsque n est grand, la matrice A^n est approximativement égale à la matrice

$$\begin{pmatrix}0.27 & 0.5 & 0.23\\ 0.27 & 0.5 & 0.23\\ 0.27 & 0.5 & 0.23\end{pmatrix}$$

Donner alors une approximation de π.

> $\pi = \begin{pmatrix}0.27 & 0.5 & 0.23\end{pmatrix}$

▶ Que peut-on en conclure sur le comportement à terme du hamster ?

> À terme, le hamster mange avec une probabilité d'environ 0.27, dort avec une probabilité d'environ 0.5 et tourne dans sa roue avec une probabilité d'environ 0.23.