Chapitre n°14

Graphes et chaînes de Markov

I. Graphes probabilistes - Matrice(s) de transition

Définition 1.1

Un graphe probabiliste (d'ordre n) est un graphe orienté et pondéré dans lequel :

- Les n sommets du graphe s'appellent les états du système et sont numérotés de 1 à n ;
- Les poids des arcs indiquent les probabilités de passage d'un état à l'autre, il est notamment parfois possible de rester sur le même état ;
- La somme des poids des arcs issus d'un même sommet est égale à 1 .

Définition 1.2

La matrice de transition d'un graphe probabiliste d'ordre n est la matrice $A = (p_{i,j}) \in \mathcal{M}_n(\mathbb{R})$ où $p_{i,j}$ représente le poids de l'arc allant du sommet i au sommet j, c'est à dire la probabilité de passer de l'état i à l'état j.

Proposition 1.3

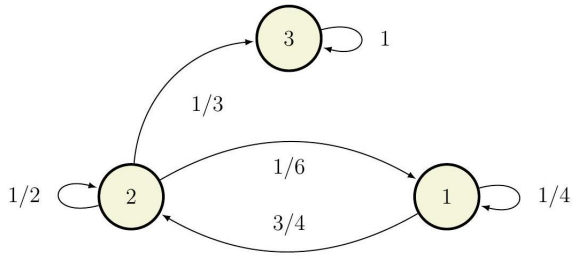
La somme de chaque ligne de la matrice de transition d'un graphe probabiliste est égale à 1 :

$$\forall i \in [\![1, n]\!], \quad \sum_{j=1}^{n} p_{i,j} = 1$$

Une telle matrice est dite **stochastique.**

Exemple :

On représente un exemple de graphe probabiliste à trois sommets.



Sa matrice de transition est la matrice

$$A = \begin{pmatrix} 1/4 & 3/4 & 0 \\ 1/6 & 1/2 & 1/3 \\ 0 & 0 & 1 \end{pmatrix}$$

On observe notamment qu'une fois sur l'état 3, on y reste (avec probabilité 1). Un tel état est dit **absorbant.**

II. Chaînes de Markov

Définition 2.1 — Chaînes de markov

Soit G un graphe probabiliste d'ordre r.
Une chaîne de Markov associée au graphe G est une suite de variables aléatoires $(X_n)_{n\in\mathbb{N}}$ toutes à valeurs dans l'ensemble des sommets de G et qui vérifie pour tout $n \in \mathbb{N}$, pour tout i et j de $[\![1,\, r]\!]$

$$\mathrm{P}_{[X_n=i]}(X_{n+1} = j) = p_{i,j} \text{ poids de l'arête allant de } i \text{ à } j$$

Remarques :

R1 – Les probabilités conditionnelles $p_{i,j}$ sont aussi appelées probabilités de transition.

R2 – L'un des points les plus importants qui apparaît dans cette définition c'est que la variable aléatoire X_{n+1} ne dépend que de la variable aléatoire X_n ce qui peut s'écrire

$$\mathrm{P}_{\bigcap_{j=0}^{n}[X_j=i_j]}(X_{n+1} = i_{n+1}) = \mathrm{P}_{[X_n=i_n]}(X_{n+1} = i_{n+1})$$

pour tout entiers i_0,..., i_n dans $[\![1,\, r]\!]$. Ceci peut se traduire par

> le futur (i.e. X_{k+1}) ne dépend que du présent (i.e. X_k) et pas du passé (i.e. $X_{k-1}, .., X_0$).

R3 – On se limite au cas où la matrice de transition est indépendante de n. En particulier, la loi d'évolution ne change pas avec le temps, la chaîne est dite **homogène.**

Définition 2.2

Soit (X_k) une chaîne de Markov associée à un graphe probabiliste d'ordre r et de matrice de transition $A = (p_{i,j}) \in \mathcal{M}_r(\mathbb{R})$.

- On dit que la chaîne est dans l'état $i \in [\![1, r]\!]$ à l'instant k si et seulement si $[X_k = i]$ est réalisé.
- On appelle k-ième état probabiliste de la chaîne le vecteur ligne $V_k \in \mathcal{M}_{1,r}(\mathbb{R})$ définie par

$$V_k = \begin{pmatrix} P(X_k = 1) & P(X_k = 2) & \dots & P(X_k = r) \end{pmatrix}$$

En particulier, $V_0 = \begin{pmatrix} P(X_0 = 1) & P(X_0 = 2) & \dots & P(X_0 = r) \end{pmatrix}$ est appelé état initial de la chaîne.

Retenir

Le k-ième état probabiliste de la chaîne caractérise la loi de X_k ; il s'agit d'une représentation de cette loi sous forme de vecteur ligne.

Remarque :

Si on pense une chaîne de Markov comme un mobile qui se déplace aléatoirement sur le graphe, connaître l'état n c'est connaître les probabilités que le mobile se retrouve sur chacun des sommets à l'instant n.

Exercice 1

Soient p, q deux réels de $]0\,;\,1\,[$ fixés. Un zappeur compulsif commence à regarder la télévision au moment 0 et hésite entre deux chaînes (la 1 et la 2). Au moment $k = 0$, il choisit la chaîne au hasard (de manière équiprobable) puis à chaque minute

- s'il est sur la chaîne 1, il change de chaîne avec probabilité p
- s'il est sur la chaîne 2, il change de chaîne avec probabilité q

On note X_k la v.a. correspondant au numéro de la chaîne que regarde le téléspectateur après k minutes.

1. Dessiner le graphe probabiliste auquel la chaîne (X_k) est associée et préciser la matrice de transition.
2. Quel est la loi de X_0 ? Quel est alors l'état probabiliste initial ?
3. Recopier et compléter la fonction Python afin qu'elle renvoie une simulation de X_k.

```
def simul_X(k,p,q):
        x=...
        for i in range(k):
                if x== ... :
                        if rd.rand() <= p :
                                ...
                else:
                        if rd.rand() <= q :
                                ...
        return(x)
```

Exercice 2

On considère la matrice

$$A = \begin{pmatrix} 0 & 0,2 & 0,8 \\ 0,3 & 0,3 & 0,4 \\ 0,8 & 0,1 & 0,1 \end{pmatrix}$$

1. Dessiner le graphe probabiliste dont A représente la matrice de transition.
2. Que fait le programme suivant :

```
def mystere(A,k,i0):
        x=i0-1
        n=len(A)
        for m in range(k):
                r=rd.random()
                i=x
                p=0
                for j in range(n):
                        if p<=r and r<p+A[i,j]:
                                x=j
                        p=p+A[i,j]
        return(x+1)
A=np.array([[0,0.2,0.8],[0.3,0.3,0.4],[0.8,0.1,0.1]])
k=100
i0=2
S=np.zeros(3)
for m in range(1000):
        j=mystere(A,k,i0)
        S[j-1]+=1
S=[S[j]/1000 for i in range(3)]
plt.grid()
plt.bar([1,2,3],S)
plt.show()
```

III. Calcul du k-ième état probabiliste

Proposition 3.1

Soit (X_k) une chaîne de Markov de matrice de transition $A = (p_{i,j}) \in \mathcal{M}_n(\mathbb{R})$.
Il découle de la formule des probabilités totales que, pour tout $k \in \mathbb{N}$, pour tout $j \in [\![1, n]\!]$,

$$\sum_{i=1}^{n} p_{i,j} P(X_k = i) = P(X_{k+1} = j)$$

Remarques :

R1 – La propriété précédente se réécrit sous forme matricielle, en faisant intervenir les états probabilistes de la chaîne. En effet, pour tout $k \in \mathbb{N}$, on a

$$V_{k+1} = V_k A$$

où $V_k = \begin{pmatrix} P(X_k = 1) & P(X_k = 2) & \dots & P(X_k = n) \end{pmatrix}$.

R2 – Si (X_k) est une chaine de Markov de matrice de transition A et de k-ième état probabiliste V_k (et d'état probabiliste initial V_0), une récurrence combinée aux résultats précédents donne, pour tout $k \in \mathbb{N}$,

$$V_k = V_0 A^k$$

Il faudra savoir redémontrer toutes les étapes menant à cette formule.

Exercice 3

On reprend l'exercice du zappeur compulsif avec $p = 1/2$ et $q = 2/3$.

1. Écrire la matrice de transition A.
2. Recopier le programme suivant et l'exécuter avec différentes valeurs initiales. Qu'observe-t-on ?

```
def etat_prob(A,k,V0):
        return(np.dot(V0,al.matrix_power(A,k))
A=np.array([[1/2,1/2],[2/3,1/3]])
V0=np.array([[1/2,1/2]])

for k in range(20):
        print(etat_prob(A,k,V0))
```

3. Déterminer les valeurs propres de A. Pour chaque valeur propre, déterminer une base du sous-espace propre associé. En déduire l'existence d'une matrice inversible P et d'une matrice diagonale D telles que

$$A = PDP^{-1}$$

(On choisira D de sorte que les coefficients diagonaux soient rangés dans l'ordre croissant.)

4. Expliciter la matrice A^k, pour tout $k \in \mathbb{N}^*$.
5. En déduire, pour tout $k \in \mathbb{N}^*$, l'expression du k-ième état probabiliste de la chaîne puis la loi de X_k.
6. Montrer que (X_k) converge en loi vers une certaine loi Z que l'on précisera.
7. Montrer que Z ne dépend pas de la loi de X_0.
8. On note $\Pi = (P(Z = 1) \quad P(Z = 2))$. Que dire que ${}^t\Pi$ pour ${}^tA$?

IV. États stables

Définition 4.1

Soit (X_k) une chaine de Markov à n états, de matrice de transition $A = (p_{i,j}) \in \mathcal{M}_n(\mathbb{R})$.
Soit $\Pi \in \mathcal{M}_{1,n}(\mathbb{R})$ un vecteur ligne définissant une loi de probabilité sur $[\![1, n]\!]$ (i.e. dont tous les coefficients sont des éléments de $[0; 1]$ avec une somme égale à 1).
On dit que Π est un état stable de la chaîne de Markov (X_k) si

$$\Pi A = \Pi$$

Auquel cas, on dit aussi que la loi de probabilité associée à Π est la loi stationnaire de la chaîne.

Remarque :

Si il existe un k_0 pour lequel X_{k_0} suit la soit de Π (c'est à dire pour lequel $V_{k_0} = \Pi$), alors, une récurrence immédiate permet de voir que, pour tout $k \geq k_0$, on a $V_k = \Pi$. La loi stationne bien.

Proposition 4.2

Soit (X_k) une chaine de Markov à n états, de matrice de transition $A = (p_{i,j}) \in \mathcal{M}_n(\mathbb{R})$.
Si $\Pi = \begin{pmatrix} \pi_1 & \pi_2 & \dots & \pi_n \end{pmatrix} \in \mathcal{M}_{1,n}(\mathbb{R})$, alors

$$\Pi \text{ est un état stable de la chaîne } \iff \begin{cases} {}^t\Pi \text{ est vecteur propre de } {}^tA \text{ associé à la valeur propre 1} \\ \Pi \text{ définit une loi de probabilité sur } [\![1, n]\!] \end{cases}$$

$$\iff \begin{cases} {}^tA \cdot {}^t\Pi = {}^t\Pi \\ \forall i \in [\![1, n]\!], \pi_i \geq 0 \\ \displaystyle\sum_{i=1}^{n} \pi_i = 1 \end{cases}$$

Attention:

Bien que 1 soit toujours valeur propre de ${}^tA$, il n'est pas clair qu'on puisse toujours trouver un vecteur propre associé dont toutes les composantes soient positives (qu'on normalise pour obtenir une loi de probabilité).

Théorème 4.3

Soit (X_k) une chaine de Markov à n états, de matrice de transition $A = (p_{i,j}) \in \mathcal{M}_n(\mathbb{R})$. Si (X_k) converge en loi vers une variable aléatoire Z, alors, nécessairement, le vecteur ligne

$$\Pi = \begin{pmatrix} P(Z = 1) & P(Z = 2) & \dots & P(Z = n) \end{pmatrix} \in \mathcal{M}_{1,n}(\mathbb{R})$$

est un état stable de la chaîne.

Attention:

La chaîne peut admettre un état stable sans qu'il y ait convergence en loi, mais s'il y a convergence c'est vers l'état stable.

Remarque :

Un résultat, complètement hors programme, affirme qu'une chaîne de Markov admet toujours un état stable. La preuve est délicate. Dans les exercices, on montrera, au cas par cas, que cet état existe.

V. Chaînes de Markov et Python

Soit (X_n) une chaîne de Markov à n états et $A = (p_{i,j})_{(i,j)\in[\![1,\,n]\!]^2}$ sa matrice de transition, c'est à dire :

$$\forall (i,j) \in [\![1,\,n]\!]^2, p_{i,j} = \mathbb{P}_{(X_n=i)}\left(X_{n+1}=j\right)$$

On écrit alors un programme qui simule la trajectoire des k premiers pas de la chaîne (k est à choisir par l'utilisateur) avec la position initiale x_0 (à choisir aussi par l'utilisateur).

```
import numpy as np

def markov(k,A,x0):
    X=np.zeros(k+1)
    X[0]=x0
    E=[ i  for i in range(1,len(A)+1)]
    for i in range(k):
        X[i+1]=np.random.choice(E,p=A[int(X[i]-1),:])
    return(X)
```

Retenir

La fonction `np.random.choice(E, p=...)` simule une variable aléatoire avec support dans E (liste des états) et dont les probabilités sont contenues dans `p`. Attention à bien faire correspondre le `E` et avec la taille de la matrice M.

Dans l'exemple précédent,

- On simule une variable à support dans $[\![1,n]\!]$ (n est la taille de la matrice de transition). Sur Python l'ensemble des états de la chaîne à $[\![0,n-1]\!]$. On rappelle que si A est une matrice carrée de taille $n\times n$, alors la commande `len(A)` renvoie l'entier n.
- Dans le i-ième tour de boucle, la chaîne se trouve donc dans l'état $X[i]$ et la loi de $X[i+1]$ s'obtient donc en conditionnant par rapport à X[i]. Les valeurs de la loi de X_{i+1} conditionnée à $X_k = $ X[k] sont sur la ligne $X[i]$ de la matrice, que l'on extrait donc avec la commande A$[int(\text{X[i]}-1), :]$.

VI. Déplacement d'un mobile

Un mobile se déplace entre les quatre sommets d'un carré numérotés 1, 2, 3 et 4 (dans le sens horaire). On note X_n la variable aléatoire donnant le numéro du sommet sur lequel se trouve le mobile à l'étape n et on note

$$U_n = \begin{pmatrix}\mathrm{P}(X_n=1) & \mathrm{P}(X_n=2) & \mathrm{P}(X_n=3) & \mathrm{P}(X_n=4)\end{pmatrix}$$

À chaque étape la probabilité de passer d'un sommet a un sommet adjacent est de $\frac{1}{4}$ et celle de passer à un sommet opposé est de $\frac{1}{2}$. À l'instant initial $n=0$ le mobile se trouve en position 2

1. Représenter la situation à l'aide d'un graphe. Les arêtes seront pondérées avec les probabilités de transition.
2. Déterminer une matrice M telle que $U_{n+1} = U_n M$.
3. Démontrer que $\forall n \in \mathbb{N} \qquad U_n = U_0 M^n$.
4. Que fait la fonction suivante ?

```
def markov(A,x0,n):
    E=[ i  for i in range(1,len(A)+1)]
    position=x0
    R=[x0]
    for i in range(1,n+1):
        position=rd.choice(E,p=A[position-1])
        R.append(position)
    return R
```

5. Créer une matrice correspondant à la matrice M et utiliser la fonction précédente pour simuler 10 déplacements avec les règles précédentes . Comment interpréter le résultat ?
6. Le mobile se trouve maintenant en position 4 quand $n = 0$. Modifier le programme précédent.
7. On note $N = {}^tM$. Montrer que 1 est valeur propre de N et calculer le sous espace propre associé. Trouver un vecteur ligne V tel que la somme des coefficients de V soit égale à 1, tous les coefficients de V soit positif et $VM = V$.
8. On aimerait savoir où se trouve "le mobile en moyenne". Recopier et compléter la fonction suivantes

```
def freq(P,x):
    '''P liste de positions successives
    x une position données fixé
    renvoie la fréquence d'apparition de x dans P'''

    c=....
    for ..........:
        if x==..... :
            c=.....;
    return .........
```

9. Utiliser la fonction précédente pour calculer la fréquence de chacune des positions lors d'un très grand nombre de simulations. Que constate t on ?

VII. Evolution d'une maladie

On modélise l'évolution d'une maladie au sein d'une population au cours du temps en classant les individus en trois groupes :

- Le groupe S des individus sains et non immunisés ;
- Le groupe M des individus malades ;
- Le groupe I des individus immunisés.

On appelle $E = \{S, M, I\}$ l'ensemble des états possibles. On donne la loi d'évolution suivante :

- À l'instant 0 , tous les individus sont dans l'état S.
- La moitié des individus dans l'état S à l'instant n restent dans le même état à l'instant $n+1$ et l'autre moitié tombe malade ; un dixième des individus immunisés à l'instant n passent dans le groupe S à l'instant $n+1$, les autres restent immunisés. Un malade sur cinq le reste, les autres guérissent et deviennent immunisés.

1. Représenter le modèle précédemment décrit par un graphe ;
2. On s'intéresse à la suite (X_n) de variables aléatoires telle que X_n désigne l'état du système l'instant n, c'est à dire que $X_n(\Omega) = \{1, 2, 3\}$, où $(X_n = 1)$ (resp. 2,3) correspond à l'état S(resp .M, I) à l'instant n.) La suite (X_n) est donc une chaîne de Markov.
3. Recopier et compléter le programme ci-dessous pour qu'il simule la suite des états $X_0, X_1, \ldots, X_n$ (c'est-à-dire une trajectoire de la chaîne).

```
def Exemple(n):
    X=np.zeros(n)
    X[0]=1
    for k in range(n-1):
        if X[k]==1:
            X[k+1]=np.random.choice(np.array([1,2,3]),...)
        elif X[k]==2:
            X[k+1]=...
        else:
            X[k+1]=...
    return(X)
```

4. Que fait le programme suivant ?

```
y=Exemple(1000)
plt.plot(y,'+')
plt.show()
```

5. Que fait le programme suivant ?

```
U=np.zeros(100)
for k in range(100):
    U[k]=Exemple(1000)[999]

frequ=np.zeros(3)
for j in range(3):
    for k in range(len(U)):
        if U[k]==j+1:
            frequ[j]=frequ[j]+1
frequ=frequ/len(U)
plt.bar([1,2,3],frequ)
plt.show()
```

6. Pour $n \in \mathbb{N}$, on note

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

   (a) Déterminer la matrice de transition de la chaîne de Markov (elle vérifie donc $U_{n+1} = U_n A$).

   (b) Montrer que, pour tout $n \in \mathbb{N}$, on a $U_n = U_0 A^n$. Préciser U_0, (c'est la loi initiale).

   (c) Définir, dans la console, la matrice de transition A et le vecteur de la loi initiale u0, et calculer $U_0 A^{1000}$. Comparer la cohérence des résultats avec ceux de la Question 2c.

VIII. Algorithme du Page Rank de Google.

Nous allons illustrer la méthode utilisée par Google pour classer les pages web par "indice de popularité". L'idée est d'étudier l'évolution des déplacements d'un individu sur internet, et de regarder la probabilité p_i d'être sur un site i au bout d'une très grand nombre de déplacements. On dit alors que p_i est alors appelée "indice de popularité de la page i ". Nous commencerons par regarder un cas très simple où il n'existerait sur le Web que 3 pages internet puis on étudiera le cas général.

Un exemple avec 3 sites. À l'issue de sa requête, un internaute est susceptible d'aller sur trois sites A, B, et C. On suppose que l'internaute se déplace forcément vers un lien de la page Web où il se trouve.

- le site A contient un lien vers lui-même, un lien vers B, et un lien vers C ;
- le site B contient 5 liens vers lui-même, un vers A et un vers C ;
- le site C comporte un lien vers A, 7 liens vers B et 4 vers C.

À l'instant 0, l'internaute choisit au hasard l'un des trois sites. Par la suite, la probabilité de passer d'un site (à l'instant n) vers un autre (à l'instant $n+1$) est proportionnelle au nombre de liens du premier site vers le deuxième.

Pour $n \in \mathbb{N}$, on désigne par a_n, b_n et c_n les probabilités des évènements A_n (resp. B_n, C_n) définis respectivement par le fait de se trouver sur le site A (resp. B, C) à l'instant n.

Enfin, on note X_n la variable aléatoire définie par

$$X_n(\omega) = \begin{cases} 1, & \text{si } \omega \in A_n \\ 2, & \text{si } \omega \in B_n \\ 3, & \text{si } \omega \in C_n \end{cases}$$

1. En justifiant rigoureusement au moins l'une des égalités, exprimer a_{n+1}, (resp. b_{n+1}, c_{n+1}) en fonction des trois réels a_n, b_n et c_n.
2. Écrire la matrice de transition S.
3. Définir la matrice S dans un langage Python puis compléter le programme suivant qui permet d'afficher la trajectoire $(X_0, X_1, \ldots, X_n)$ de la chaîne de Markov (X_n).

```
def navigation(n,M):
        X=np.zeros(n)
        X[0]=...
        for k in range(n-1):
                X[k+1]=np.random.choice(a=list(range(1,4)),p=...)
        return(X)
```

4. Compléter le programme suivant afin de définir une matrice E de taille 3×11 dont la i-ème colonne contient le vecteur U_{i-1}. Afficher les lois de $U_0, U_1, \ldots, U_{10}$.

```
E=np.zeros([3,11])
E[:,0]=(1/3)*np.ones([3])
for k in range(10):
        E[:,k+1]=...
```

6. Tracer sur un même graphique les courbes des suites (a_n), (b_n) et (c_n). Que remarque-t-on ?
7. On appelle "indice de notoriété" d'une page A la valeur $a = \lim a_n$. Déterminer la valeur exacte des indices de notoriété en calculant la valeur exacte de l'état stable. On admettra que la probabilité invariante est unique.

Etude du cas général

Le cas général. On modélise l'ensemble des pages du Web par un ensemble fini d'états $E = \{1, 2, \ldots, N\}$ (le nombre N est bien très grand, de l'ordre de 10^{13}, mais reste fini).

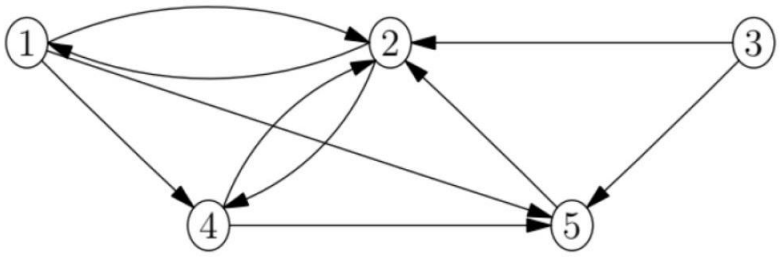
On modélise les déplacements de l'internaute par une chaînes de Markov (X_n) où la variable X_n est le numéro de la page où se situe l'internaute après n déplacements. Après avoir fait des études statistiques des comportements sur le Web, Google part du principe que :

- À l'instant 0 , l'internaute choisit une page internet au hasard parmi les N pages ;
- Lorsqu'un internaute est sur une page i :
- dans 85% des cas, il se déplacera au hasard vers l'une des ℓ_i pages pointées par i (ici, une page a au plus un lien vers une autre).
- dans 15% des cas, il se déplacera au hasard vers n'importe laquelle des pages du Web.

La matrice de transition $M = (P(i,j))$ est alors définie par

$$P(i,j) = \begin{cases} 0,85 \times \dfrac{1}{\ell_i} + 0,15 \times \dfrac{1}{N}, & \text{si } i \text{ pointe vers } j \\ 0,15 \times \dfrac{1}{N}, & \text{sinon} \end{cases}$$

Nous allons illustrer l'algorithme de PageRank sur un groupe autonome de $N = 5$ pages différentes, ayant des liens pointant les pages les unes vers les autres, selon le diagramme suivant :



8. La matrice de transition est donnée par

```
A=0.15*(1/5)*np.ones([5,5])+0.85*np.array([[0,1/3,0,1/3,1/3],[0.5,0,0,0.5,0],[0,0.
```

9. Déterminer avec la commande `al.eig` une valeur approchée de la loi stationnaire.
10. Recopier et exécuter les commandes suivantes. La chaîne converge-t-elle vers l'état stationnaire ?

```
N=100
E=np.zeros([5,N+1])
E[:,0]=(1/5)*np.ones([5])

for k in range(N):
    E[:,k+1]=np.transpose(E[:,k].dot(A))

for k in range(5):
    plt.subplot(1,5,k+1)
    plt.ylim(0,1)
```

```
    plt.plot([k for k in range(N+1)],E[k,:],'+')
plt.show()
```

11. Classer les pages par indice de popularité. Les résultats vous semblent-ils cohérents ?