`Python` td 10 - Retour sur les matrices

I. Les commandes au programme sur les matrices

On importe les librairies suivantes : `import numpy as np` et `import numpy.linalg as al`

```
np.array, np.zeros, np.ones, np.eye, np.linspace,np.arange,
np.dot, np.transpose,
np.sum, np.min, np.max,np.cumsum
```

La commande `np.mean(A)` donne la moyenne de tous les coefficients de A. La commande `np.mean(A,0)` renvoie une matrice ligne contenant les moyennes de chaque colonne de A. La commande `np.mean(A,1)` est une matrice ligne contenant les moyennes de chaque ligne de A.
On peut utiliser de façon similaire `np.sum(A,0)`, `np.max(A,0)` etc...

La commande `np.shape(M)` renvoie la taille de la matrice M sous la forme `[nb-lignes, nb-colonnes]`
Si l'on tape
`a,b=np.shape(A)`
alors `a` est le nombre de lignes et `b` est le nombre de colonnes.
On peut aussi taper `a=np.shape(A)[0]` et `b=np.shape(A)[1]`.

```
al.inv, al.matrix_rank, al.matrix_power
```

La commande `al.solve(A,Y)` permet de résoudre un système linéaire du type $AX = Y$ dans le cas où A est une matrice carrée inversible.
La commande `al.eig(A)` renvoie un couple (L, P) de matrices où L est la matrice ligne des valeurs propres (= eigen values en anglais) et P est une matrice carrée dont les colonnes sont des vecteurs propres.
Si on tape `L,P=al.eig(A)`, alors `L` sera la matrice ligne des valeurs propres.

Si M est une matrice, la commande `M>=1` renvoie une matrice de booléens (True ou False) de même taille que M indiquant si chaque coefficient est ≥ 1 ou non. Rappelons qu'en Python `True=1` et `False=0`
Si M et N sont deux matrices de même taille, la commande `M==N` renvoie une matrice de booléens de même taille que M indiquant à chaque fois si $m_{i,j} = n_{i,j}$.

II. Exercices

Exercice 1
Ecrire une fonction intitulée `def Positif(A)`, dont la variable est une matrice A, et qui renvoie le nombre de coefficients de A qui sont positifs ou nuls.

Exercice 2
Soit $n \in \mathbb{N}$, $n \geq 2$ et $U_n = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} \in \mathcal{M}_{n,1}(\mathbb{R})$. On considère l'application Φ qui va de $\mathcal{M}_n(\mathbb{R})$ dans $\mathbb{R}$ et qui est définie par :

$$\Phi : A \mapsto {}^tU_n.A.U_n$$

1. Ecrire une fonction Python intitulée `def Phi(A):` fonction d'une matrice carrée A, qui renvoie $\Phi(A)$. *Cette fonction devra tout d'abord déterminer n puis définir U_n*

2. Calculer à la main $\Phi(A)$ en fonction des coefficients de A. En déduire une autre méthode plus rapide pour calculer $\Phi(A)$.

Exercice 3
1. Soit

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

   (a) Déterminer à l'aide de Python le rang de la matrice $A + I$.

   (b) Calculer avec Python le spectre de A.

2. Ecrire une fonction intitulée `def sp(n):` qui définit la matrice J de taille $n \times n$ remplie de 1 puis qui retourne son spectre. Tester pour $n = 3$. Que constate-t-on ?

Exercice 4
. Soit $H = \begin{pmatrix} 1 & 1 & 2 \\ 1 & 2 & 6 \\ 2 & 6 & 24 \end{pmatrix}$ et $Y = \begin{pmatrix} 6 \\ 24 \\ 120 \end{pmatrix}$. A l'aide de Python montrer que H est inversible et déterminer X tel que $HX = Y$.

Exercice 5
1. Ecrire une fonction Python intitulée `def Spectre(A) :` qui a pour variable une matrice carrée $A \in \mathcal{M}_n(\mathbb{R})$ et qui renvoie le spectre de A

2. S'en servir pour calculer le spectre de $A = \begin{pmatrix} 2 & -2 & -5 \\ -2 & 5 & 10 \\ 1 & -2 & -4 \end{pmatrix}$. A est-elle diagonalisable ?

Exercice 6
1. Ecrire une fonction intitulée `def Sym(A) :` qui a pour variable une matrice carrée A et qui renvoie `True` si cette matrice est symétrique et `False` sinon.

2. On dit qu'une matrice symétrique est positive ssi ses valeurs propres sont toutes positives.
   Ecrire une fonction Python intitulée `def SymPos(A) :` qui a pour variable une matrice carrée A et qui renvoie `True` si cette matrice est symétrique et positive et `False` sinon.

Exercice 7
Une matrice carrée A est dite **stochastique** si elle est à coefficients positifs et si la somme des coefficients de chacune de ses lignes vaut 1.
Ecrire une fonction intitulée `Stochas(A) :` qui prend pour paramètre une matrice carrée A, renvoie `True` si cette matrice est stochastique et `False` sinon.