# Correction TP3 python

Exercice 1

1)programme :

```
import numpy as np
A=np.array(([1,-3],[2,-7]))
print("A=",A)
B=np.array(([1,1],[-1,0]))
print("B=",B)
print("AB=",np.dot(A,B))
C=np.transpose(np.dot(A,B))
print("C=",C)
print("---------------------------------")
print("transposée de A=",np.transpose(A))
print("transposée de B=",np.transpose(B))
D=np.dot(np.transpose(B),np.transpose(A))
print("D=",D)
```

Le programme renvoie :

$AB = \begin{pmatrix} 4 & 1 \\ 9 & 2 \end{pmatrix}$, $C = \begin{pmatrix} 4 & 9 \\ 1 & 2 \end{pmatrix}$, ${}^tA = \begin{pmatrix} 1 & 2 \\ -3 & -7 \end{pmatrix}$, ${}^tB = \begin{pmatrix} 1 & -1 \\ 1 & 0 \end{pmatrix}$ et $D = \begin{pmatrix} 4 & 9 \\ 1 & 2 \end{pmatrix}$.

2)On remarque que $C = D$, c'est-à-dire que ${}^t(AB) = ({}^tB)({}^tA)$, égalité bien connue, vraie pour toutes matrices $A$ et $B$.

Exercice 2

1)$(S)$ s'écrit alors matriciellement sous la forme $AX = B$ avec :

$A = \begin{pmatrix} 1 & -2 & -4 \\ 2 & 0 & 3 \\ -3 & 5 & 9 \end{pmatrix}$, $X = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$ et $B = \begin{pmatrix} 2 \\ 1 \\ 3 \end{pmatrix}$.

2)programme :

```
import numpy as np
import numpy.linalg as al
A=np.array(([1,-2,-4],[2,0,3],[-3,5,9]))
print("A=",A)
B=np.array(([2],[1],[3]))
print("B=",B)
print("inverse de A=",al.inv(A))
print("La solution est",np.dot(al.inv(A),B))
```

Le programme renvoie : $A^{-1} = \begin{pmatrix} 15 & 2 & 6 \\ 27 & 3 & 11 \\ -10 & -1 & -4 \end{pmatrix}$.

La solution du système $(S)$ est alors donnée par $X = A^{-1}B = \begin{pmatrix} 50 \\ 90 \\ -33 \end{pmatrix}$.

3)La commande al.solve(A,B) donne directement la solution de $(S)$.

Exercice 3

1)$A = \begin{pmatrix} \frac{1}{6} & \frac{1}{6} \\ 1 & \frac{1}{3} \end{pmatrix}$.

2)Soit $\mathscr{P}(n)$ la proposition : « $C_n = A^n C_0$ ».
$\mathscr{P}(0)$ s'écrit : « $C_0 = A^0 C_0$ ». Elle est vraie car $A^0 = I$.
Soit $n \in \mathbf{N}$. Supposons $\mathscr{P}(n)$ vraie, montrons que $\mathscr{P}(n+1)$ est vraie.
$C_{n+1} = AC_n = A(A^n C_0) = A^{n+1} C_0$. Donc $\mathscr{P}(n+1)$ est vraie.
On conclut que $\forall n \in \mathbf{N}, C_n = A^n C_0$.

3)programme :

```
import numpy as np
import numpy.linalg as al
A=np.array(([1/6,1/6],[1,1/3]))
n=int(input("entrer n"))
C_0=np.array(([1],[2]))
C_n=np.dot(al.matrix_power(A,n),C_0)
print("u_",n,"=",C_n[0,0])
print("v_",n,"=",C_n[1,0])
```

Le programme renvoie par exemple : $u_{10} = 0.01387323$ et $v_{10} = 0.04161967$.
On peut conjecturer que $\lim\limits_{n \to +\infty} u_n = 0$ et $\lim\limits_{n \to +\infty} v_n = 0$.

Exercice 4

1)programmes :

```
#programme 1
import numpy as np
suite=[1]
for n in range(9):
    suite.append(1-np.exp(-suite[n]))
print('u_9=',suite[9])
```

```
#programme 2
import numpy as np
u=1
for n in range(9):
    u=1-np.exp(-u)
print('u_9 =',u)
```

2)C'est le programme 1 le mieux adapté car il a construit une liste contenant les termes $u_0$, $u_1$, ..., $u_9$. Il a donc $u_5$ en mémoire, on le récupère par la commande suite[5], ce qui donne $u_5$ = 0.2680768156757.
Pour avoir la somme des termes, on tape la commande sum(suite), ce qui donne $\sum_{n=0}^{9} u_n$ = 3.860899569371.

Exercice 5

1)$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}, B = \begin{pmatrix} 5 \\ 6 \end{pmatrix}, C = \begin{pmatrix} 7 & 8 \end{pmatrix}, V = \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 7 & 8 \end{pmatrix}$ et $H = \begin{pmatrix} 1 & 2 & 5 \\ 3 & 4 & 6 \end{pmatrix}$.

2)$A = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$, $L = \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix}$ et $C = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$.

Par concaténations, $B = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$ et $D = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix}$.

3)programme :

```
import numpy as np
C=np.ones(shape=(3,1))
Z=np.zeros(shape=(3,4))
H=np.hstack((C,Z))
L=np.ones(shape=(1,5))
D=np.vstack((L,H,L))
```

Exercice 6

programme :

```
import numpy as np
def f(a,b,n):
    A=b*np.ones(shape=(n,n))+(a-b)*np.eye(n)
    return A
```

Exercice 7

1)$u$ est une matrice ligne de $\mathscr{M}_{1,n}(\mathbf{R})$.
Par transposition, ${}^t u$ est une matrice colonne de $\mathscr{M}_{n,1}(\mathbf{R})$.
$\left({}^t u\right) v$ est donc la matrice carrée de $\mathscr{M}_n(\mathbf{R})$ définie par :

$\left({}^t u\right) v = \begin{pmatrix} u_1 & u_1 & . & . & u_1 \\ u_2 & u_2 & . & . & u_2 \\ . & . & . & . & . \\ . & . & . & . & . \\ u_n & u_n & . & . & u_n \end{pmatrix}$.

2)Le programme construit de proche en proche les composantes de la matrice ligne u, initialement égale à la matrice nulle.
Le passage dans la boucle donne : $u[0,0] = 1$, $u[0,1] = 2$, ..., $u[0,n-1] = n$.
Le programme affiche donc finalement $u = (1, 2, ..., n)$.

3)On applique la question 1) en prenant $u_i = i$ pour tout $i \in [\![1, n]\!]$.
On a alors $\left({}^t u\right) v = A$. D'où le programme :

```
import numpy as np
n=int(input("entrer n"))
u=np.zeros(shape=(1,n))
for k in range(n):
    u[0,k]=k+1
v=np.ones(shape=(1,n))
c=np.dot(np.transpose(u),v)
print(c)
```

4)programme :

```
import numpy as np
n=int(input("entrer n"))
L=np.ones(shape=(1,n))
A=L
#concaténation verticale de L, 2L, ..., nL
for k in range(2,n+1):
    A=np.vstack((A,k*L))
print(A)
```

Exercice 8

1)Exemple de matrice stochastique : $A = \begin{pmatrix} 0.2 & 0.8 \\ 0.3 & 0.7 \end{pmatrix}$.

2)programme :

```
def stochastique(A):
    for i in range(len(A)):
        for j in range(len(A)):
            if A[i,j]<0:
                return False
        if sum(A[i,:])!=1:
            return False
    return True
```