T.P. VII - Matrices

Code Capytale : aa20-2467643

I - Ce qu'il faut savoir

Solution de l'exercice 1.

```
import numpy as np

n = 10
I = np.eye(3) # Contient la matrice identité : A^0
A = np.array([[1,2,1],[1/2,1,2],[1,1/2,1]]) # Contient la matrice
B = np.dot(A, A) # Contient la matrice A^2

for k in range(3, n+1): # k varie de 3 à n
    C = 3 * B + 9/4 * I # Contient la matrice A^k
    I = A # Contient la matrice A^(k-2)
    A = B # Contient la matrice A^(k-1)
    B = C # Contient la matrice A^k

print(B) # Affiche A^n = A^(10)
```

□

Solution de l'exercice 2.

1.

```
import numpy as np

def g(x): # Définition de la fonction g
    return 2 * x - 1 + np.log(x/(x+1))

a = 0.5 # initialisation de la valeur de a
b = 1 # initialisation de la valeur de g

while b - a > 10**(-2) : # s'arrête dès que b - a <= 10^(-2)
    m = (a + b)/2 # calcule le milieu de [a, b]
    if g(a) * g(m) <= 0: # si g change de signe entre a et m
        b = m
    else: # si g change de signe entre m et b
        a = m

print(a, b)
```

2.

```
import numpy as np
import matplotlib.pyplot as plt

U = np.zeros((51, 1)) # Crée un vecteur contenant uniquement des 0

for n in range(1, 51):
    U[n] = (2 * n - 1) - g(n) # U[n] contient la valeur de u_n

X = np.arange(0, 51) # Liste des abscisses : entiers de 0 à
50
# U contient [u_0, u_1, u_2, u_3, ..., u_50]
S = np.cumsum(U)
# S contient [u_0, u_0 + u_1, u_0 + u_1 + u_2, ..., u_0 + ... + u_

plt.figure()
plt.plot(X, S, '+')
T = np.arange(0.1, 51, 0.1)
plt.plot(T, np.log(T), 'r')
plt.show()
```

Le graphique représente la suite de points de coordonnées $(n, S_n)_{0 \leqslant n \leqslant 50}$, où $S_n = \sum_{i=0}^{n} u_i$.
On peut conjecturer que la suite (S_n) tend vers $+\infty$ et donc que la série de terme général u_n diverge. □

II - Suites récurrentes

Solution de l'exercice 3.

```
import numpy as np

n = 20
A = np.array([[-2, 1], [3, 0]]) # A contient A
C = np.array([[0], [1]]) # C contient (u_0, v_0)

for k in range(1, n+1): # k varie de 1 à 20
    C = np.dot(A, C) # C contient C_k

print(C[0, 0], C[0, 1]) # C contient C_20
```

□

Solution de l'exercice 4. On note $U_n = \begin{pmatrix} u_n \\ v_n \end{pmatrix}$.

```
import numpy as np

n = 12
A = np.array([[1, 1], [2, 0]]) # A contient A
C = np.array([[1.], [0.]]) # C contient (u_0, v_0) = (1, 0)

for k in range(1, n+1): # k varie de 1 à 12
    C = np.dot(A, C) # C contient C_n

print(C[0, 0], C[1, 0]) # C contient C_12 = (u_12, v_12)
```

□

Solution de l'exercice 5.

```
import numpy as np

A = np.array([[4, -6, 2], [2, -4, 2], [-2, 2, 0]]) # A contient A
U = [[0.], [1.], [2.]] # U contient U_0 = (a_0, b_0, c_0)
for i in range(1, 11): # i varie de 1 à 10
    U = np.dot(A, U) # U contient U_i

print("a_10", U[0,0]) # U contient U_10 = (a_10, b_10, c_10)
print("b_10", U[1,0]) # U contient U_10 = (a_10, b_10, c_10)
print("c_10", U[2,0]) # U contient U_10 = (a_10, b_10, c_10)
```

□

Solution de l'exercice 6.

1.

```
n = 20
v = -1 # Contient la valeur de u_0
u = 1 # Contient la valeur de u_1

for i in range(2, n+1): # i varie de 2 à 20
    a = u # a contient la valeur de u_(i-1)
    u = 3 * u + 2 * v - 4 # u contient la valeur de u_i
    v = a # v contient la valeur de u_(i-1)

print(u) # u contient la valeur de u_20
```

2.

```
n = 20
X = np.array([[1.], [-1.]]) # contient X_0
A = np.array([[3., 2.], [1., 0.]]) # crée la matrice A
B = np.array([[4.], [0.]]) # crée la matrice B

for i in range(1, n+1): # i varie de 1 à 20
    X = np.dot(A, X) + B # contient la valeur de X_i

print(X[1, 0]) # affiche la seconde composante de X_20, soit a_20
```

□

Solution de l'exercice 7.

```
import numpy as np
import matplotlib.pyplot as plt

A = np.array([[1,0,0],[0,1,2],[2,0,1]]) # A contient A
u = np.zeros((11, 1)) # u a 10 lignes et 1 colonnes, que des zéros
v = np.zeros((11, 1)) # v a 10 lignes et 1 colonnes, que des zéros
w = np.zeros((11, 1)) # w a 10 lignes et 1 colonnes, que des zéros

u[0] = 1 # la première composante de u contient u_0
v[0] = 0 # la première composante de u contient v_0
w[0] = 2 # la première composante de u contient w_0
X = np.array([1, 0, 2])
```

```
for i in range(1, 11):
    X = np.dot(A, X)
    u[i] = 1
    v[i] = X[1]
    w[i] = X[2]

T = np.arange(0, 11)

plt.figure()
plt.plot(T, u, 'r.') # Trace la suite u avec des points rouges
plt.plot(T, v, 'go') # Trace la suite v avec des points verts
plt.plot(T, w, 'b+') # Trace la suite w avec des + bleus
plt.show()
```

□