DM 2

A rendre le Vendredi 27 Septembre

Exercice 1

On note A la matrice

$$A = \begin{pmatrix} 2 & -1 & -1 & -1 \\ -1 & 2 & -1 & -1 \\ -1 & -1 & 2 & -1 \\ -1 & -1 & -1 & 2 \end{pmatrix}$$

et I la matrice unité de taille 4.

1. Donner les valeurs de a et b telles que la commande `Python` suivante construise la matrice A :

```
A = a*np.ones((4,4))+b*np.eye(4,4)
```

2. Calculer A^2.
3. Montrer qu'il existe deux réels α et β tels que $A^2 = \alpha A + \beta I$.
4. En déduire (sans faire de pivot) que A est inversible et exprimer son inverse en fonction de A et de I.
5. Établir par récurrence que pour tout entier naturel n il existe des réels α_n et β_n tels que :

$$A^n = \alpha_n A + \beta_n I.$$

   On exprimera α_{n+1} et β_{n+1} en fonction de α_n et β_n.

6. Compléter la fonction `Python` suivante qui prend n en entrée et renvoie en sortie les valeurs de α_n et β_n :

```
def suites(n):
    alpha = 0
    beta = 1
    for k in range(1,n+1):
        mem = .....
        alpha = .....
        beta = .....
    return alpha, beta
```

7. (a) Montrer que la suite $(\alpha_n)_{n\in\mathbb{N}}$ est récurrente linéaire d'ordre 2 et déterminer l'expression de α_n en fonction de n.
   (b) En déduire celle de β_n.
8. (a) Les expressions de α_n et β_n sont-elles encore valables pour $n = -1$?
   (b) Montrer que pour tout $n \in \mathbb{N}^*$, $A^{-n} = \alpha_{-n} A + \beta_{-n} I$.

Exercice 2

On dit qu'une matrice A carrée d'ordre n est une matrice nilpotente s'il existe un entier naturel k (appelé indice de nilpotence de A) tel que :

$$A^{k-1} \neq 0_n \quad \text{et} \quad A^k = 0_n$$

où 0_n représente la matrice carrée nulle d'ordre n.
Soit A une matrice carrée d'ordre n. On dit que le couple (Δ, N) est une décomposition de Dunford de A lorsque :

$$\begin{cases} \Delta \text{ est un matrice diagonalisable i.e. il existe } P \text{ inversible et } D \text{ diagonale telles que } \Delta = PDP^{-1} \\ N \text{ est une matrice nilpotente} \\ \Delta N = N\Delta \text{ et } A = N + \Delta \end{cases}$$

1. On pose :
$$A = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}, \quad \Delta = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \text{et} \quad N = \begin{pmatrix} 0 & 2 \\ 0 & 0 \end{pmatrix}$$
Vérifier que (Δ, N) est une décomposition de Dunford de A.
Dans toute la suite de l'exercice, on pose :
$$A = \begin{pmatrix} 3 & 1 & -1 \\ -2 & 0 & 2 \\ 0 & 0 & 1 \end{pmatrix}, \quad N = \begin{pmatrix} 0 & 0 & -1 \\ 0 & 0 & 2 \\ 0 & 0 & 0 \end{pmatrix}, \quad \Delta = \begin{pmatrix} 3 & 1 & 0 \\ -2 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad D = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

2. (a) Calculer $(A - I)^2(A - 2I)$.
   (b) En déduire les valeurs propres possibles de A à l'aide de la question précédente puis vérifier que ce sont bien des valeurs propres de A.
   (c) Utiliser la question 2.(a) pour prouver que A est inversible et exprimer son inverse A^{-1} en fonction des puissances de A et de I.

3. Retrouver les valeurs propres de A par la méthode du pivot.

4. On considère les matrices colonnes
$$X_1 = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}, \quad X_2 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \quad \text{et} \quad X_3 = \begin{pmatrix} 1 \\ -2 \\ 0 \end{pmatrix}$$
   (a) Prouver que la famille (X_1, X_2, X_3) est une base de $\mathscr{M}_{3,1}(\mathbb{R})$.
   (b) Montrer que X_1 est un vecteur propre de Δ; à quelle valeur propre est-il associé ?
   Même question pour X_2 et X_3.
   (c) En déduire une matrice P inversible et une matrice D diagonale telles que : $\Delta = PDP^{-1}$.
   (d) Montrer que $D = P^{-1}\Delta P$ (en transformant l'égalité ci-dessus, sans faire de réel produit de matrices 3×3).
   (e) Calculer P^{-1}.
   (f) On entre les instructions suivantes dans la console :

```
>>> Delta = np.array([[3,1,0],[-2,0,0],[0,0,1]])
>>> Sp, VP = al.eig(Delta)
>>> print(Sp)
array([2., 1., 1.])
>>> print(VP)
array([[ 0.70710678, - 0.4472136 , 0. ]
       [-0.70710678,   0.89442719, 0. ]
       [ 0.        , 0.          , 1. ]])
```

i. Que fait la fonction `al.eig` ?

ii. La matrice P obtenue en `Python` n'est pas la même que celle obtenue dans l'énoncé. Justifier que les colonnes de la matrice P donnée par `Python` forment aussi une base de vecteurs propres de Δ.

5. (a) Établir que N est une matrice nilpotente.
   (b) Vérifier que (Δ, N) est une décomposition de Dunford de la matrice A.
   (c) En utilisant la formule du binôme de Newton que l'on justifiera, donner l'expression de A^n en fonction des puissances de Δ, de N et de n.
   (d) Établir que : $\forall k \in \mathbb{N}$, $\Delta^k N = N$
   (e) Proposer une décomposition de Dunford de A^n.

Exercice 3

On désigne par n un entier naturel non nul, et l'on se propose d'étudier les racines de l'équation

$$(E_n) : \quad \ln(x) + x = n$$

A cet effet, on introduit la fonction f de la variable réelle x définie sur $\mathbb{R}_+^*$ par :

$$f(x) = \ln(x) + x.$$

1. Montrer que, pour tout $n \in \mathbb{N}^*$, l'équation (E_n) admet une et une seule solution x_n.
2. Montrer que la suite (x_n) est croissante.
3. Montrer que : $\forall x > 0$, $\ln(x) < x$.
4. Prouver que l'on a :
$$\forall n \in \mathbb{N}^*, \quad \frac{n}{2} \leq x_n \leq n.$$
5. Quelle est la limite de x_n quand n tend vers $+\infty$?
6. Montrer que $\lim\limits_{n\to+\infty} \dfrac{\ln(x_n)}{n} = 0$ et en déduire que $x_n \sim n$.
7. Calculer la limite de $x_{n+1} - x_n$ quand n tend vers $+\infty$.
8. On pose, pour tout $n \in \mathbb{N}^*$, $u_n = \dfrac{n - x_n}{\ln(n)}$
   (a) Montrer que :
   $$\forall n \in \mathbb{N}^*, \quad u_n - 1 = \frac{\ln\left(\dfrac{x_n}{n}\right)}{\ln(n)}.$$
   (b) Quelle est la limite de u_n quand n tend vers $+\infty$?
   (c) Prouver alors que : $1 - u_n \sim \frac{1}{n}$.
9. En déduire que : $x_n = n - \ln(n) + \dfrac{\ln(n)}{n} + o\left(\dfrac{\ln(n)}{n}\right)$.
10. Compléter le programme suivant qui, pour tout $n \in [\![1, 100]\!]$, détermine une valeur approchée à 10^{-6} près de x_n (i.e. de la solution de l'équation $f(x) = n$) par méthode de dichotomie (se faire un dessin si besoin pour illustrer la méthode) :

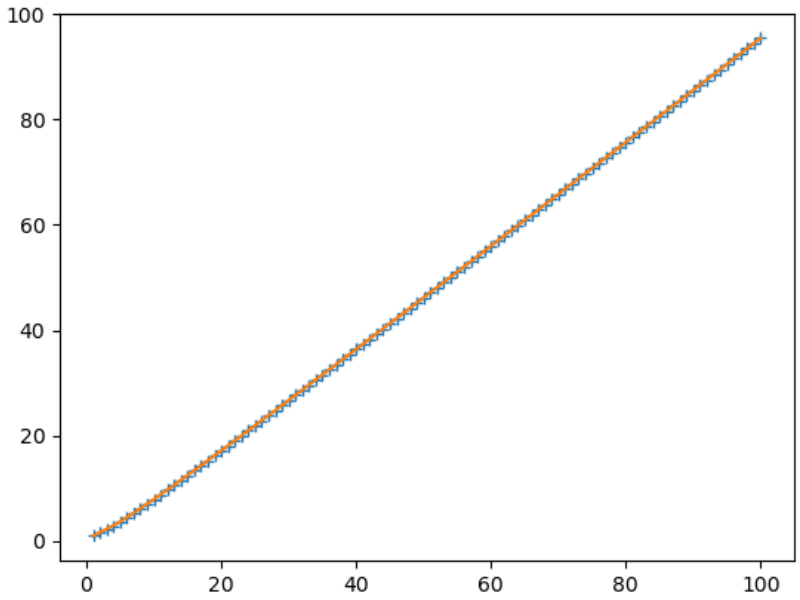
```
def f(x):
    return(.....)

x = np.arange(1,101)
y = np.zeros(101)
for n in range(1,101) :
    a = n/2
    b = n
    while np.abs(b-a) >10**(-6) :
        m = (a+b)/2
        if f(m) < n :
            .....
        else :
            .....
    y[k-1] = (a+b)/2

plt.plot(x,y,'+')

n = list(range(1,101))
plt.plot(n,n-np.log(n)+np.log(n)/n)
plt.show()
```

Après exécution, on obtient le graphique suivant :



Commenter les deux représentations graphiques obtenues.