Corrigé Code de partage avec Capytale : c0d2-8228870

On utilisera les bibliothèques suivantes :

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

Exercice 1 - illustration de l'instabilité et des conditions initiales

On considère le système différentiel suivant :

$$(\mathscr{S}) \ : \qquad \forall t \in \mathbb{R} \quad \left\{ \begin{array}{ccccccc} x'(t) & = & 3x(t) & - & y(t) & + & z(t) \\ y'(t) & = & 3x(t) & - & 2y(t) & + & 2z(t) \\ z'(t) & = & 3x(t) & + & y(t) & - & z(t) \end{array} \right.$$

On notera $X(t) = \begin{pmatrix} x(t) \\ y(t) \\ z(t) \end{pmatrix}$ pour tout $t \in \mathbb{R}$

1. Définir la matrice A sous Python.

   `A=np.array([[3, -1, 1], [3, -2, 2], [3, 1, -1]])`

2. Déterminer les valeurs propres de A (utiliser Python, on rappelle que `al.eig` renvoie les valeurs propres et des vecteurs propres associés).

   On comprend que les valeurs propres de A sont $-3, 0$ et 3

3. Déterminer une matrice P inversible et une matrice D diagonale telles que : la dernière ligne de P est $\begin{pmatrix} 1 & 1 & 1 \end{pmatrix}$ et les éléments diagonaux de D sont rangés dans l'ordre croissant.
   On pourra utiliser la matrice Q définie avec la commande suivante et la commande `np.diag` pour D

   ```
   spectre, Q=al.eig(A)
   for i in range(3):
       Q[:, i]=Q[:, i]/Q[2, i]
   print (spectre, Q)
   ```

   La commande nous permet d'avoir des vecteurs propres « normalisés » (le coefficient de la dernière ligne vaut 1) : $\begin{pmatrix} -1/3 \\ -1 \\ 1 \end{pmatrix}$, $\begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}$ et $\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$ respectivement.
   On définit D avec `D=np.diag([-3, 0, 3])` et pour définir P, on peut le faire explicitement avec `P=np.array([[-1/3, 0, 1], [-1, 1, 1], [1, 1, 1]])`, ou on peut réutiliser la matrice Q avec la commande suivante (pour remettre dans le bon ordre les vecteurs propres) :

   ```
   D=np.diag([-3, 0, 3])
   P=np.zeros([3, 3])
   P[:, 0]=Q[:, 2]; P[:, 1]=Q[:, 0]; P[:, 2]=Q[:, 1]
   print(P)
   ```

4. Résoudre $(\mathscr{S})$
   On écrira $X(t)$ sous la forme :
$$X(t)=C_0e^{r_0t}U+C_1e^{r_1t}V+C_2e^{r_2t}W$$
   U,V,W étant trois vecteurs colonnes à déterminer, r_0,r_1,r_2 trois réels à déterminer, C_0,C_1,C_2 étant des paramètres (constantes quelconques).

   Pour information (vérification et pour la suite du problème)
   La résolution de $(\mathscr{S})$ donne :
$$\exists(C_0,C_1,C_2)\in\mathbb{R}^3,\ \forall t\in\mathbb{R},\ X(t)=C_0e^{-3t}U+C_1V+C_2e^{3t}W$$
   où : U,V,W sont les colonnes de P (dans cet ordre), c'est-à-dire :
$$U=\begin{pmatrix}-1/3\\-1\\1\end{pmatrix}\qquad V=\begin{pmatrix}0\\1\\1\end{pmatrix}\qquad W=\begin{pmatrix}1\\1\\1\end{pmatrix}$$
   $(\mathscr{S})\Leftrightarrow X'(t)=AX(t)$ où $X'(t)=\begin{pmatrix}x'(t)\\y'(t)\\z'(t)\end{pmatrix}$ et $A=\begin{pmatrix}3&-1&1\\3&-2&2\\3&1&-1\end{pmatrix}$ donc $(\mathscr{S})\Leftrightarrow X'(t)=PDP^{-1}X(t)\Leftrightarrow P^{-1}X'(t)=P^{-1}PDP^{-1}X(t)\Leftrightarrow P^{-1}X'(t)=DP^{-1}X(t)\Leftrightarrow Y'(t)=DY(t)$ où $Y(t)=P^{-1}X(t)=\begin{pmatrix}a(t)\\b(t)\\c(t)\end{pmatrix}$ et alors $Y(t)=P^{-1}X'(t)=\begin{pmatrix}a'(t)\\b'(t)\\c'(t)\end{pmatrix}$
   or $DY(t)=\begin{pmatrix}3&0&0\\0&0&0\\0&0&-3\end{pmatrix}\begin{pmatrix}a(t)\\b(t)\\c(t)\end{pmatrix}=\begin{pmatrix}-3a(t)\\0\\3c(t)\end{pmatrix}$ donc
   $Y'(t)=DY(t)\Leftrightarrow\begin{pmatrix}a'(t)\\b'(t)\\c'(t)\end{pmatrix}=\begin{pmatrix}-3a(t)\\0\\3c(t)\end{pmatrix}\Leftrightarrow\left\{\begin{array}{lcl}a'(t)&=&-3a(t)\\b'(t)&=&0\\c'(t)&=&3c(t)\end{array}\right.\Leftrightarrow\left\{\begin{array}{ll}a(t)=C_0e^{-3t}&\text{où } C_0\in\mathbb{R}\\b(t)=C_1&\text{où } C_1\in\mathbb{R}\\c(t)=C_2e^{3t}&\text{où } C_2\in\mathbb{R}\end{array}\right.$
   alors $Y=P^{-1}X(t)$ donne $X(t)=PY(t)=\begin{pmatrix}-1/3&0&1\\-1&1&1\\1&1&1\end{pmatrix}\begin{pmatrix}a(t)\\b(t)\\c(t)\end{pmatrix}=\begin{pmatrix}-1/3a(t)+0+c(t)\\-a(t)+b(t)+c(t)\\a(t)+b(t)+c(t)\end{pmatrix}$
   donc $X(t)=a(t)U+b(t)V+c(t)V=C_0e^{-3t}U+C_1V+C_2e^{3t}W$

5. La fonction Python suivante définit la fonction `X(t, condition)` renvoyant un triplet `[x(t), y(t), z(t)]` avec $t\mapsto X(t)$ solution de $(\mathscr{S})$ sous la condition initiale $X(0)={}^t(a\quad b\quad c)$. `condition` sera la liste `[a, b, c]`.
   Expliquer les lignes 4 et 7.

```python
def SolX(t, condition):
    P=np.array([[-1/3, 0, 1], [-1, 1, 1], [1, 1, 1]])
    r=np.array([-3, 0, 3])
    C=np.dot(al.inv(P),np.transpose(condition)) # Définition de C[0], C[1],
   C[2]
    X=np.zeros(3)
    for k in range(3):
          X=X+C[k]*np.exp(r[k]*t)*P[:, k]     # Calcul de X(t)
    return X
```

La condition $X(0)=\begin{pmatrix}a\\b\\c\end{pmatrix}$ donne : $C_0U+C_1V+C_2W=\begin{pmatrix}a\\b\\c\end{pmatrix}$ soit : $P\begin{pmatrix}C_0\\C_1\\C_2\end{pmatrix}=\begin{pmatrix}a\\b\\c\end{pmatrix}$

et donc : $\begin{pmatrix}C_0\\C_1\\C_2\end{pmatrix}=P^{-1}\begin{pmatrix}a\\b\\c\end{pmatrix}$ c'est ce qu'on calcule en ligne 4

et en ligne 7, on calcule la somme avec la formule de la question 4.
Finalement pour n'importe quelles conditions initiales (a,b,c) et $t\in\mathbf{R}$, le programme renvoie la valeur de $X(t)$ pour l'unique solution correspondante (sous la forme d'une matrice, en utilisant l'écriture de la question précédente, à l'aide des colonnes de P).

6. Le code suivant trace les courbes x, y, z solution du problème de Cauchy avec $x(0)=-1/3$, $y(0)=-1$, $z(0)=1$ sur trois graphiques différents.
Commentez le résultat.

```python
T=np.linspace(0, 5, 100)
X=[]
Cauchy=[-1/3, -1, 1]
fonction=['x', 'y', 'z']
for t in T:
    X.append(SolX(t, Cauchy))

for k in range(3):
    plt.close(k)
    plt.figure(k)
    x=[X[i][k] for i in range(100)]
    plt.plot(T, x, label=fonction[k])
    plt.legend()
    plt.show()
```

Pour préciser un peu le contenu du programme, il commence par créer, pour les conditions initiales données en énoncé, les 100 valeurs de $X(t)$ (ces valeurs étant issues du `linspace`, donc équitablement réparties sur $[0,5]$).
Ensuite, il crée trois graphiques, celui du $x(t)$ avec les valeurs des premières lignes de $X(t)$, ...
On remarque que, d'après le graphique, $\lim_{x\to+\infty}x(t)=\lim_{x\to+\infty}y(t)=\lim_{x\to+\infty}z(t)=0$, ce qui est le cas, car alors puisque $X(0)=U$ alors $X(t)$ est ici l'unique solution telle que $X(0)=U$, ce qui implique $C_1=C_2=0$
$X(t)$ ne possède donc qu'une « composante » selon U (et pas selon V et W) et donc le seul terme variable est e^{-3t} qui tend vers 0 quand t tend vers $+\infty$.

7. Le code suivant trace les courbes x, y, z solution du problème de Cauchy avec $x(0)=-0.333333$, $y(0)=-1$, $z(0)=1$ sur trois graphiques différents.
Commentez le résultat.

```
T=np.linspace(0, 5, 100)
X=[]
Cauchy=[-0.333333, -1, 1]
fonction=['x', 'y', 'z']
for t in T:
    X.append(SolX(t, Cauchy))

for k in range(3):
    plt.close(k)
    plt.figure(k)
    x=[X[i][k] for i in range(100)]
    plt.plot(T, x, label=fonction[k])
    plt.legend()
    plt.show()
```

Le seul changement au sein de ce programme est la condition initiale, et on remarque que cette fois, $x(t), y(t)$ et $z(t)$ semblent diverger. Ce qui est le cas car, $X(0)$ a beau être très proche de U, il n'y a pas pour autant égalité, donc on trouvera $C_2 \neq 0$ et la solution divergera.