Corrigé Code de partage avec Capytale : ded2-1732010

Préambule - manipuler des matrices

Voici quelques commandes essentielles pour manipuler des matrices, à tester :

```
import numpy as np
A=np.array([[0,1],[1,0]]); B=np.array([[1,2],[3,4]]) ; # définition de matrices
    2*2
C=np.array([[1,2,8],[3,4,9]]) # définition de matrice 2*3
A+B; A+C; # addition de matrices, A+C n'est pas défini (message d'erreur)
np.dot(A,B); np.dot(A,C); np.dot(C,A); # multiplication de matrices (CA n'est pas
     défini donc renvoie un message d'erreur)
A[1,:];C[:,2] # extrait la ligne 1 de A (i.e. la deuxième), la colonne 2 de C
np.sum(C[1,:]) # renvoie le résultat de la somme des coefficients de la première
    ligne de C
import numpy.linalg as al # librairie pour utiliser la commande ci-dessous
al.matrix_power(A,2), al.matrix_power(A,3) # renvoie A puissance 2 (on peut
    vérifier avec np.dot(A,A) ; A puissance 3
```

Exercice 1 - identités remarquables et puissances

On considère les matrices $A = \begin{pmatrix} 0 & 0,5 & 0 \\ 1 & 0 & 1 \\ 0 & 0,5 & 0 \end{pmatrix}$ et $B = \begin{pmatrix} 0 & 1 & 5 \\ -1 & -2 & 3 \\ 1 & 5 & 3 \end{pmatrix}$

1. Définir ces matrices avec Python

```
A=array([[0,0.5,0],[1,0,1],[0,0.5,0]])
B=array([[0,1,5],[-1,-2,3],[1,5,3]])
```

2. (a) Avec Python, calculer ensuite $(A+B)^2$ puis $A^2+2AB+B^2$ et commenter le résultat.

   On peut faire directement le calcul de $(A+B)^2$

```
np.dot(A+B,A+B) # ou al.matrix_power(A+B,2)
```

   ou avec une matrice auxiliaire : $C = A+B$

```
C=A+B
np.dot(C,C) # ou al.matrix_power(C,2)
```

   puis en comparant avec $A^2+2AB+B^2$, on voit que l'identité remarquable n'est pas vérifiée, cas le plus fréquent pour les matrices (puisque A et B ne commutent pas).

```
# Calcul de A**2+2AB+B**2
np.dot(A,A)+2*np.dot(A,B)+np.dot(B,B)
```

   Nota bene : si on a importé la bibliothèque `linalg` on peut utiliser indifféremment `al.matrix_ power(A,2)` à la place de `np.dot(A,A)` (de même pour B).
   On peut aussi tester l'égalité avec `np.dot(A+B,A+B) ==np.dot(A,A)+2*np.dot(A,B)+np.dot(B,B`

   (b) Les matrices A et B vérifient-elles d'autres identités remarquables ?

   Non comme on pouvait si attendre, on peut tester directement les égalités avec Python (et un `==`) qui renverra pour chaque coefficient de la matrice `true` si l'égalité est vérifiée

et `false` sinon. On voit même qu'il n'y a aucun coefficient identique (pour chacune des identités).

```
np.dot(A-B,A-B)==np.dot(A,A)-2*np.dot(A,B)+np.dot(B,B)
np.dot(A+B,A-B)==np.dot(A,A)-np.dot(B,B)
```

3. Avec Python, calculer A^2, A^3 et A^4 puis émettre une conjecture sur A^n

   On peut utiliser `al.matrix_ power(A,...)` (après avoir importé `numpy.linalg`), on constate que $A^2 = A^4$ et que $A^3 = A$ et on comprend rapidement que $\forall n \in \mathbb{N}, A^{2n} = A^2$ et $A^{2n+1} = A$

```
al.matrix_power(A,2)
al.matrix_power(A,3)
al.matrix_power(A,4)
al.matrix_power(A,2)==al.matrix_power(A,4)
al.matrix_power(A,3)==A
```

Quelques compléments avec linalg

Exercice 2 - échauffement

On pose $A = \begin{pmatrix} 1 & 2 \\ 0 & -1 \end{pmatrix}$ et $B = \begin{pmatrix} 3 \\ 2 \end{pmatrix}$

1. Définir les matrices A et B avec Python.

   Comme plus haut :

```
A=np.array([[1,2],[0,-1]]); B=np.array([[3],[2]]) ;
```

2. Tester la commande `al.inv(A)` et commenter le résultat renvoyé.

```
al.inv(A)
```

   Cette commande donne la matrice inverse de A si elle existe, on peut vérifier en faisant `np.dot(A,al.inv(A))`

3. Tester la commande `al.solve(A,B)` et commenter le résultat renvoyé.

```
al.solve(A,B)
```

   Cette commande résoud le système $AX = B$ ou X est la matrice des inconnues, on trouve ici $x = 7$ et $y = -2$, on peut le retrouver en faisant `np.dot(al.inv(A),B)` car on sait que quand la matrice du système est inversible, l'unique solution est $A^{-1}B$

Exercice 3 - Déterminer une équation de parabole

- On considère une parabole d'équation $y = ax^2 + bx + c$ où a, b et c sont trois réels avec $a \neq 0$
- On suppose que la parabole passe par les point $A(1;4), B(-2;-5)$ et $C(3;0)$
- On cherche á déterminer les trois réels a, b et c

1. Justifier que a, b et c sont solutions du système : $\begin{cases} a+b+c &= 4 \\ 4a-2b+c &= -5 \\ 9a+3b+c &= 0 \end{cases}$

   a, b et c sont solutions du système car les point $A(1;4), B(-2;-5)$ et $C(3;0)$ sont sur la courbe, autrement dit $f(1) = 4$; $f(-2) = -5$ et $f(3) = 0$ où $f(x) = ax^2 + bx + c$

   la matrice du système est $A = \begin{pmatrix} 1 & 1 & 1 \\ 4 & -2 & 1 \\ 9 & 3 & 1 \end{pmatrix}$ et le second membre est la matric $B = \begin{pmatrix} 4 \\ -5 \\ 0 \end{pmatrix}$

2. Avec Python, définir la matrice A associée au système et la matrice B second membre du système.

   Donc avec Python, on définit :

   ```
   A=np.array([[1,1,1],[4,-2,1],[9,3,1]]); B=np.array([[4],[-5],[0]]) ;
   ```

3. Avec Python, montrer que A est inversible et trouver son inverse.

   On exécute `al.inv(A)` et on trouve
   `np.array([[-1/6,1/15,1/10],[1/6,-4/15,1/10],[1,1/5,-1/5]])`

4. Avec Python, résoudre le système en utilisant A^{-1}

   On sait que quand la matrice du système est inversible, l'unique solution est $A^{-1}B$ d'où

   ```
   np.dot(al.inv(A),B)
   ```

   et on trouve que la matrice solution est $B = \begin{pmatrix} -1 \\ 2 \\ 3 \end{pmatrix}$ autrement dit $a = -1; b = 2 etc = 3$

5. Avec Python, résoudre le système en utilisant la fonction `solve`

   Avec `al.solve(A,B)` on retrouve bien la même solution