Autour des matrices : Annales 2015 / 2016 / 2017 / 2018 / 2019

EDHEC – 2017

- On note E l'espace vectoriel des fonctions polynomiales de degré inférieur ou égal à 2 et on rappelle que la famille (e_0, e_1, e_2) est une base de E, les fonctions e_0, e_1 e_2 étant définies par :

$$\forall t \in \mathbb{R} \quad e_0(t) = 1, \quad e_1(t) = t, \quad e_2(t) = t^2$$

  On considère l'application φ qui, à toute fonction P de E, associe la fonction, notée $\varphi(P)$, définie par :

$$\forall x \in \mathbb{R},\ (\varphi(P))(x) = \int_0^1 P(x+t)\, dt$$

- On montre que :

$$A = \mathrm{Mat}_{(e_0,e_1,e_2)}(\varphi) = \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

a) Compléter les commandes **Scilab** suivantes pour que soit affichée la matrice A^n pour une valeur de n entrée par l'utilisateur :

```
1 n = input('entrez une valeur pour n : ')
2 A = [---]
3 disp(---)
```

Démonstration.

- On stocke la matrice A dans la variable `A`.

```
2 A = [1, 1/2, 1/3; 0, 1, 1; 0, 0, 1]
```

- Puis on demande l'affichage de A^n.

```
3 disp(A ^n)
```

□

HEC – 2017

- Pour tout $n \in \mathbb{N}^*$, on note $\mathscr{M}_n(\mathbb{R})$ l'ensemble des matrices carrés à n lignes et n colonnes à coefficients réels et B_n l'ensemble des matrices de $\mathscr{M}_n(\mathbb{R})$ dont tous les coefficients sont égaux à 0 ou à 1.
- *Exemple 2.* Soit B la matrice de B_3 définie par : $B = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$.

  On considère les instructions et la sortie (-→) **Scilab** suivantes :

```
1 B = [0,1,0;1,0,0;0,0,1]
2 P = [1,1,0;1,-1,0;0,0,1]
3 inv(P) ⋆ B ⋆ P
```

```
-->
    1.     0.     0.
    0.    -1.     0.
    0.     0.     1.
```

a) Déduire les valeurs propres de B de la séquence **Scilab** précédente.

Démonstration.

Notons $D = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ et $P = \begin{pmatrix} 1 & 1 & 0 \\ 1 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$.

D'après la séquence **Scilab**, $P^{-1}BP = D$, d'où $B = PDP^{-1}$.
Ainsi, B est semblable à une matrice diagonale.
Elle est donc diagonalisable et ses valeurs propres sont les coefficients diagonaux de D.

$$\boxed{\mathrm{Sp}(B) = \{-1, 1\}}$$

□

b) Déterminer une base de chacun des sous-espaces propres de B.

Démonstration.

- Le programme **Scilab** précédent nous fournit la diagonalisation de la matrice B :

$$B = PDP^{-1}$$

La famille $\mathcal{F} = \left(\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \right)$ constituée des colonnes de la matrice P est une base de vecteurs propres de B. Plus précisément, pour tout $i \in [\![1, 3]\!]$, le $i^{\text{ème}}$ vecteur de $\mathcal{F}$ est un vecteur propre associé au coefficient diagonal $d_{i,i}$.

- On en déduit :

$$E_{-1}(B) \supset \mathrm{Vect}\left(\begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix} \right) \qquad \text{et} \qquad E_1(B) \supset \mathrm{Vect}\left(\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \right)$$

- Ainsi, $\dim(E_{-1}(B)) \geqslant 1$ et $\dim(E_1(B)) \geqslant 2$.
Or, comme B est diagonalisable :

$$\dim(E_{-1}(B)) + \dim(E_1(B)) = \dim(\mathscr{M}_{3,1}(\mathbb{R})) = 3$$

On en déduit : $\dim(E_{-1}(B)) = 1$ et $\dim(E_1(B)) = 2$.

$$\boxed{\text{Ainsi : } E_{-1}(B) = \mathrm{Vect}\left(\begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix} \right) \quad \text{et} \quad E_1(B) = \mathrm{Vect}\left(\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \right).}$$

Commentaire

- On aurait aussi pu déterminer ces deux sous-espaces propres « à la main ».
- Détaillons la méthode. Soit $X = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \in \mathscr{M}_{3,1}(\mathbb{R})$.

$$\begin{aligned} X \in E_{-1}(B) &\Leftrightarrow (B + I_3)\, X = 0 \\ &\Leftrightarrow \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 2 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \\ &\Leftrightarrow \left\{ \begin{array}{rcrcl} x & + & y & = & 0 \\ x & + & y & = & 0 \\ & & 2\,z & = & 0 \end{array} \right. \\ &\Leftrightarrow \left\{ \begin{array}{rcl} x & = & -y \\ z & = & 0 \end{array} \right. \end{aligned}$$

Ainsi :

$$\begin{aligned} E_{-1}(B) &= \{X = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \in \mathscr{M}_{3,1}(\mathbb{R}) \mid X \in E_{-1}(B)\} \\ &= \{\begin{pmatrix} x \\ y \\ z \end{pmatrix} \mid x = -y \ \texttt{ET} \ z = 0\} \\ &= \{\begin{pmatrix} -y \\ y \\ 0 \end{pmatrix} \mid y \in \mathbb{R}\} \\ &= \{y \cdot \begin{pmatrix} -1 \\ 1 \\ 0 \end{pmatrix} \mid y \in \mathbb{R}\} = \text{Vect}\left(\begin{pmatrix} -1 \\ 1 \\ 0 \end{pmatrix}\right) \end{aligned}$$

La famille $\left(\begin{pmatrix} -1 \\ 1 \\ 0 \end{pmatrix}\right)$ est donc :

× génératrice de $E_{-1}(B)$,

× libre car constituée d'un unique vecteur non nul.

C'est donc une base de $E_{-1}(B)$.

- De même, on démontre : $E_1(B) = \text{Vect}\left(\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}\right)$.

□

`HEC – 2018`

À la fin de l'exercice 1 de l'épreuve `HEC 18`, on trouvait une question **Scilab** qui consistait à écrire une matrice $n \times n$ dont les coefficients $q(\ell, k)$ étaient donnés par la valeur d'une fonction $q : \mathbb{N}^2 \to \mathbb{N}$ étudiée dans le sujet. Plus pécisément, les questions précédant le **Scilab** permettait d'affirmer :

× $\forall k \in \mathbb{N}^*$, $q(1, k) = 1$.

× $\forall \ell \geqslant k$, $q(\ell, k) = p(k) = q(k, k)$.

× $\forall k < \ell$, $q(\ell, k) = q(k, k)$.

× $\forall \ell \in [\![1, n]\!]$, $q(\ell, \ell) = 1 + q(\ell - 1, \ell)$.

× $\forall k > \ell$, $q(\ell, k) = q(\ell - 1, k) + q(\ell, k - \ell)$.

- La fonction **Scilab** suivante dont le script est incomplet (lignes 5 et 6), calcule une matrice `qmatrix(n)` telle que pour chaque couple $(\ell, k) \in [\![1, n]\!]^2$, le coefficient situé à l'intersection de la ligne ℓ et de la colonne k est égal à $q(\ell, k)$.

```
1  function q = qmatrix(n)
2      q = ones(n, n)
3      for L = 2:n
4          for K = 2:n
5              if (K<L) then
6                  q(L,K) = ................
7              elseif (K==L) then
8                  q(L,K) = ..............
9              else
10                 q(L,K) = q(L-1,K) + q(L,K-L)
11             end
12         end
13     end
14 endfunction
```

L'application de la fonction `qmatrix` à l'entier $n = 9$ fournit la sortie suivante :

```
--> qmatrix(9)
    1.    1.    1.    1.    1.    1.    1.    1.    1.
    1.    2.    2.    3.    3.    4.    4.    5.    5.
    1.    2.    3.    4.    5.    7.    8.   10.   12.
    1.    2.    3.    5.    6.    9.   11.   15.   18.
    1.    2.    3.    5.    7.   10.   13.   18.   23.
    1.    2.    3.    5.    7.   11.   14.   20.   26.
    1.    2.    3.    5.    7.   11.   15.   21.   28.
    1.    2.    3.    5.    7.   11.   15.   22.   29.
    1.    2.    3.    5.    7.   11.   15.   22.   30.
```

a) Compléter les lignes 5 et 6 du script de la fonction `qmatrix`.

Démonstration.
Le but de cette question est d'obtenir la matrice $\left(q(\ell, k)\right)_{\substack{1 \leqslant \ell \leqslant n \\ 1 \leqslant k \leqslant n}}$.
Pour ce faire, il faut se servir des résultats précédents.

- D'après la question ***4.a)(i)***, pour tout $k \in \mathbb{N}^*$, $q(1, k) = 1$.

  On en déduit que la première ligne de la matrice recherchée ne contient que des 1.

- D'après la question ***4.a)(ii)***, pour tout entier $\ell \geqslant k$, $q(\ell, k) = p(k) = q(k, k)$. Cette propriété est notamment vérifiée pour $k = 1$. Ainsi :

$$\forall \ell \in [\![1, n]\!],\ q(\ell, 1) = q(1, 1) = 1$$

  On en déduit que la première colonne de la matrice recherchée ne contient que des 1.

- La stratégie du programme consiste à créer initialement une matrice carrée d'ordre `n` remplie de 1 et stockée dans une variable `q`.

```
2      q = ones(n, n)
```

  La première ligne et première colonne de cette matrice est, de fait, constituée de 1.
  Le reste du programme consiste à remplir cette matrice ligne par ligne (de la $2^{\text{ème}}$ à la $n^{\text{ème}}$).
  D'où la présence de la structure itérative suivante :

```
3       for L = 2:n
```

- La ligne ℓ de la matrice est mise à jour en procédant comme suit.

  (1) On met à jour les coefficients à gauche du coefficient diagonal. Autrement dit, les valeurs $q(\ell,k)$ pour $k<\ell$. Pour ce faire, on se sert de nouveau du résultat de la question ***4.a)(ii)***, qui stipule que pour $k<\ell$: $q(\ell,k)=p(k)=q(k,k)$. Ce qui se traduit comme suit :

```
4           for K = 2:n
5               if (K<L) then
6                   q(L,K) = q(K,K)
```

  (le coefficient en position (ℓ,k) est donné par la valeur du coefficient diagonal situé dans la même colonne)

  (2) On met alors à jour le coefficient diagonal. Pour ce faire, on se sert du résultat de la question ***4.c)(ii)***, qui stipule que pour tout $\ell\in[\![1,n]\!]$: $q(\ell,\ell)=1+q(\ell-1,\ell)$.
  Ce qui se traduit comme suit :

```
7               elseif (K==L) then
8                   q(L,K) = 1 + q(L-1,L)
```

  (le coefficient en position (ℓ,ℓ) est donné par la valeur du coefficient directement situé au-dessus auquel on ajoute 1)

  (3) On met alors à jour les coefficients à droite du coefficient diagonal.
  Pour ce faire, on se sert du résultat de la question ***4.c)(i)***, qui stipule que pour tout $k>\ell$: $q(\ell,k)=q(\ell-1,k)+q(\ell,k-\ell)$. Ce qui se traduit comme suit :

```
9               else
10                  q(L,K) = q(L-1,K) + q(L,K-L)
```

□

b) Donner un script **Scilab** permettant de calculer $p(n)=q(n,n)$ à partir d'une valeur de n entrée au clavier.

Démonstration.

- D'après la question ***4.a)(ii)***, on a, pour tout $\ell\geqslant k$: $Q(\ell,k)=P(k)$.
  On a notamment : $Q(n,n) = P(n)$. Et ainsi :

$$p(n) = q(n,n)$$

- Ainsi, $p(n)$ est le coefficient en position (n,n) de la matrice $\big(q(\ell,k)\big)_{\substack{1\leqslant\ell\leqslant n\\1\leqslant k\leqslant n}}$.
  Pour obtenir ce coefficient on écrit un programme :

  (1) qui demande à l'utilisateur d'entrer au clavier une valeur pour n et la stocke dans une variable `n`.

  (2) qui génère la matrice `q` à l'aide de la fonction de la question précédente.

  (3) qui affiche la valeur contenu dans la variable `n`.

  On obtient le programme **Scilab** suivant :

```
1  n = input('Entrez une valeur entière non nulle n')
2  q = qmatrix(n)
3  disp(q(n,n))
```

□

HEC – 2019

La fonction **Scilab** suivante permet de multiplier la $i^{\text{ème}}$ ligne L_i d'une matrice A par une réel sans modifier ses autres lignes, c'est-à-dire de lui appliquer l'opération élémentaire $L_i \leftarrow a\,L_i$ (où $a \neq 0$).

```
function B = multilig(a, i, A)
    [n, p] = size(A)
    B = A
    for j = 1:p
        B(i, j) = a * B(i,j)
    end
endfunction
```

a) Donner le code **Scilab** de deux fonctions `adlig` (d'arguments `b`, `i`, `j`, `A`) et `echlig` (d'arguments `i`, `j`, `A`) permettant d'effectuer respectivement les autres opérations sur les lignes d'une matrice :

$$L_i \leftarrow L_i + b\,L_j \ (i \neq j) \quad \text{et} \quad L_i \leftrightarrow L_j \ (i \neq j)$$

Démonstration.

- On s'inspire de la fonction donnée pour créer `adlig` :

```
function B = adlig(b, i, j, A)
    [n, p] = size(A)
    B = A
    for k = 1:p
        B(i, k) = B(i, k) + b * B(j, k)
    end
endfunction
```

Commentaire

On note que la variable `j` est ici une variable d'entrée du programme (on l'utilise pour désigner la ligne ajoutée dans l'opération élémentaire $Li \leftarrow L_i + b\,L_j$ $(i \neq j)$). Cela oblige à renommer la variable d'itération du programme `multilig`.

- La fonction `echlig` est créée suivant le même principe :

```
function B = echlig(i, j, A)
    [n, p] = size(A)
    B = A
    aux = 0
    for k = 1:p
        aux = B(i, k)
        B(i, k) = B(j, k)
        B(j, k) = aux
    end
endfunction
```

Commentaire

- On a introduit ici une variable auxiliaire appelée `aux`. Le but de cette variable est de ne pas perdre d'information lors de l'échange des valeurs des deux coefficients de la même colonne. Plus précisément :
  - × l'instruction de la ligne 6 permet de stocker la valeur de `B(i, k)`.
  - × en ligne 7, on écrase la valeur du coefficient $B(i,k)$ en lui affectant la valeur $B(j,k)$.
  - × enfin, en ligne 8, on affecte à `B(i, k)` la valeur de `aux`, c'est-à-dire la valeur **initiale** (et pas la nouvelle valeur) du coefficient $B(i,k)$.
- On pouvait aussi tirer parti du fait que l'on travaille sur une copie `B` de la matrice `A` d'entrée (jamais modifiée) pour ne pas introduire de variable auxiliaire `aux`.

```
1  function B = echlig(i, j, A)
2      [n, p] = size(A)
3      B = A
4      for k = 1:p
5          B(i, k) = A(j, k)
6          B(j, k) = A(i, k)
7      end
8  endfunction
```

□

b) Expliquer pourquoi la fonction `multligmat` suivante retourne le même résultat `B` que la fonction `multlig`.

```
1  function B = multiligmat(a, i, A)
2      [n, p] = size(A)
3      D = eye(n, n)
4      D(i, i) = a
5      B = D * A
6  endfunction
```

Démonstration.
Nommons A, B, i, a, n et p les éléments codés par les variables du programme correspondantes.

- L'instruction en ligne 3 : `D = eye(n, n)` permet de créer la matrice identité I_n.
  (le nom `eye` *provient d'un jeu sur les sonorités : on crée à l'aide de cette instruction la matrice identité qui en anglais se dit « identity matrix » qu'il faut lire* `eye`*-dentity matrix)*
- L'instruction en ligne 4 : `D(i, i) = a` permet de remplacer le coefficient $D_{i,i}$ par la valeur a. On crée ainsi une matrice D diagonale carrée d'ordre n dont :
  - × le $i^{\text{ème}}$ coefficient diagonal est la valeur a,
  - × les autres coefficients diagonaux ont tous la même valeur 1.
- L'instruction en ligne 5 : `B = D * A` permet de stocker, dans la variable `B`, le résultat de la multiplication $D \times A$. Détaillons ce calcul.

  Soit $(i', j) \in [\![1, n]\!] \times [\![1, p]\!]$. Par la formule de multiplication matricielle, on a :

$$B_{i',j} \quad = \quad \sum_{k=1}^{n} D_{i',k} \times A_{k,j} \quad = \quad D_{i',i'} \times A_{i',j} \qquad (\textit{car } D_{i',k} = 0 \textit{ si } k \neq i')$$

  Deux cas se présentent alors :
  - × si $i' = i$ alors $D_{i',i'} = D_{i,i} =$ a et ainsi : $B_{i,j} = a \times A_{i,j}$.

La $i^{\text{ème}}$ ligne de B est obtenue en multipliant la $i^{\text{ème}}$ ligne de A par a.

× si $i' \neq i$ alors $D_{i',i'} = 1$ et ainsi : $B_{i',j} = 1 \times A_{i',j} = A_{i',j}$.

Les autres lignes de B sont des copies des lignes correspondantes de la matrice A.

On en conclut que la fonction `multiligmat` permet de calculer la matrice obtenue en appliquant à A l'opération élémentaire $L_i \leftarrow a\,L_i$.
Cela correspond bien au calcul effectué par la fonction `multilig`.

Commentaire

- La matrice $D \in \mathscr{M}_n(\mathbb{R})$ décrite dans cette question est une matrice **de dilatation**. L'opération élémentaire $L_i \leftarrow a\,L_i$ (resp. $C_i \leftarrow a\,C_i$) se traduit par la multiplication matricielle à gauche (resp. à droite) de la matrice initiale A par la matrice D.
- Illustrons de point par un exemple simple. Considérons $D = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 3 \end{pmatrix}$. Alors :

$$D\,M = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 5 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 \\ 1 & 2 & -1 \\ 2 & 4 & -2 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 2 & -1 \\ 10 & 20 & -10 \end{pmatrix}$$

(on multiplie la $3^{ème}$ ligne par 5)

$$\text{et} \quad M\,D = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 2 & -1 \\ 2 & 4 & -2 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 5 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 2 & -5 \\ 2 & 4 & -10 \end{pmatrix}$$

(on multiplie la $3^{ème}$ colonne par 5)

□