# TP1 : Révisions sur la création de matrices en **Scilab**

**Pré-requis** : l'objectif des premières séances de TP est de faire le point sur des fonctionnalités importantes de **Scilab** qui ont été vues en première année. Je vous invite à consulter les chapitres de cours correspondants sur ma page : support informatique.
On pourra en particulier se reporter au « CH 5 : Manipulation de matrices en **Scilab** ».

## I. Mise en place de votre environnement de travail

Afin de pouvoir conserver une semaine après l'autre le travail effectué, nous commençons tout d'abord par créer un répertoire de sauvegarde.

- Placez-vous dans le dossier `Mes documents`. Créez alors un nouveau dossier `Info_2a` (pour les cubes : créez un dossier `Info_3a`). Ce dossier contiendra l'ensemble de vos TP de l'année.
- Placez-vous dans le dossier `Info_2a` (resp. `Info_3a`). Créez alors un nouveau dossier `TP_1`.

À chaque TP, il faudra créer un nouveau dossier `TP_...` à l'intérieur du dossier `Info_2a` (resp. `Info_3a`).

## II. Création de matrices en Scilab : les « mille et une » façons

Dans la suite, on considère les matrices ci-dessous.

$$A = \begin{pmatrix} 1 & 3 & 5 & 7 \end{pmatrix} \qquad B = \begin{pmatrix} 4 & 2 & -1 \end{pmatrix} \qquad C = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad D = \begin{pmatrix} 0 \\ -2 \\ 0 \\ 0 \end{pmatrix}$$

### II.1. Création d'une matrice par valeurs (ligne par ligne)

- Écrire en **Scilab** les matrices précédentes en les stockant dans des variables `A`, `B`, `C`, `D`.

```
A=[1, 3, 5, 7]; B=[4, 2, -1]; C=[1, 0, 1; 0, 1, 0; 0, 0, 1]; D=[0; -2; 0; 0]
```

*(du fait de la présence du séparateur « ; », seule la dernière matrice sera affichée à l'écran)*

- Rappeler le rôle des délimiteurs « , » et « ; ».

> - Le symbole « , » permet de délimiter les colonnes.
> - Le symbole « ; » permet de délimiter les lignes.

### II.2. Création d'une matrice par blocs

Dans la question précédente, on a créé les matrices en entrant les valeurs ligne par ligne. Il est aussi possible de créer une matrice en juxtaposant les différentes colonnes qui la composent.

- Écrire la matrice $C$ comme la succession des 3 matrices colonnes $\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$, $\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$ et $\begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$.
  Stocker le résultat dans la variable `C` et noter ci-dessous l'appel correspondant.

```
C = [[1;0;0], [0;1;0], [1;0;1]]
```

L'exemple précédent est un cas particulier de la création de matrices par blocs. On considère par la suite les matrices ci-dessous.

$$M = \begin{pmatrix} 1 & 3 & 5 & 7 \\ 4 & 2 & -1 & 0 \\ 1 & 0 & 1 & -2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \qquad N = \begin{pmatrix} 0 & -2 & 0 & 0 \\ 1 & 0 & 1 & 4 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & -1 \end{pmatrix}$$

► Écrire en **Scilab** la matrice $M$. Pour cela, utiliser les variables `A, B, C, D`.
Stocker le résultat dans la variable `M` et noter l'appel correspondant.

> `M = [ A; [ [B; C], D ] ]` ou tout simplement `M = [ A; [B; C], D ]`.
> *(il n'y a pas besoin de parenthèses pour regrouper les éléments appartenant à la même ligne)*

► Rappeler l'opérateur permettant d'obtenir la transposée d'une matrice.

> L'opérateur de transposition se note à l'aide d'une apostrophe « ' » en **Scilab**.

► Écrire en **Scilab** la matrice $N$. Pour cela, utiliser les variables `B, C, D`.
Stocker le résultat dans la variable `N` et noter l'appel correspondant.

> `N = [ D'; [C, B'] ]` ou tout simplement `N = [ D'; C, B' ]`.

**Remarque**

- Les crochets `[ ]` permettent de regrouper différents éléments au sein d'un même bloc.
- Ce faisant, on veillera à ce que les tailles de ces matrices le permettent. Par exemple :
  - × l'appel `N = [ D'; [C, B'] ]` est licite,
  - × mais l'appel `N = [ [D'; C], B' ]` ne l'est pas. Notamment car on en peut créer un bloc avec la transposée de `D` (de taille $1 \times 4$) en haut et la matrice `C` (qui n'a que 3 colonnes) en dessous.

## II.3. Création de matrices à l'aide d'opérateurs prédéfinis en Scilab

► La matrice `A` est un vecteur ligne composé de valeurs régulièrement espacées.
Deux méthodes en **Scilab** permettent d'obtenir ce type de vecteurs.
Illustrer ces procédés en fournissant les appels pour créer `A`.

> On peut opérer de deux manières différentes :
> × `A = 1:2:7` : on insiste ici sur l'écart (de 2) entre chaque valeur,
> × `A = linspace(1,7,4)` : on insiste ici sur le nombre de valeurs (4) que contient le vecteur.

► Noter l'appel permettant d'obtenir une matrice de taille `3` × `4` remplie de 1.

```
ones(3,4)
```

► Noter l'appel permettant d'obtenir une matrice de taille `4` × `2` remplie de 0.

```
zeros(4,2)
```

- Noter l'appel permettant d'obtenir une matrice de taille $3 \times 8$ contenant des cœfficients aléatoirement choisis dans $[0, 1]$.

```
rand(3,8)
```

- Noter l'appel permettant d'obtenir une matrice de taille $2 \times 5$ dont les éléments diagonaux sont des 1 et les autres cœfficients sont nuls.

```
eye(2,5)
```

- Stocker dans une variable `I` la matrice identité de taille $3 \times 3$.

```
I = eye(3,3)
```

## II.4. Accès et modification des éléments d'une matrice

- Noter l'appel permettant d'obtenir la matrice $\begin{pmatrix} 1 & 3 & 5 \\ 4 & 2 & -1 \end{pmatrix}$, sous-matrice de la matrice `M`.

```
M(1:2,1:3)
```

- Même question avec la matrice $\begin{pmatrix} -2 \\ 0 \\ 1 \\ 0 \end{pmatrix}$, sous-matrice de la matrice `N`.

```
N(:,2)
```

- Même question avec la matrice $\begin{pmatrix} 1 & 5 & 7 \\ 4 & -1 & 0 \\ 1 & 1 & -2 \end{pmatrix}$, sous-matrice de la matrice `M`.

```
M(1:3,[1,3,4])
```

- Modifier un cœfficient de la matrice `I` de sorte à obtenir la matrice `C`.

```
I(1,3) = 1
```

## II.5. Taillle d'une matrice

- Comment récupère-t-on en **Scilab** la **taille** d'une matrice `A` ?
  Comment récupère-t-on le nombre de lignes de `A` ? Et le nombre de colonnes ?

L'appel `size(A)` permet de récupérer un vecteur ligne de taille $1 \times 2$ contenant le nombre de lignes de `A` (premier cœfficient) et le nombre de colonnes de `A` (deuxième cœfficient).

Il y a deux manières de récupérer ce nombre de lignes et de colonnes.

*1)* Soit on réalise : `s = size(A)`. Le nombre de lignes est obtenu par l'appel `s(1)` et le nombre de colonnes est obtenu par l'appel `s(2)`.

*2)* Soit on réalise : `[nbl, nbc] = size(A)`. Cela permet de stocker le nombre de lignes de `A` dans `nbl` et le nombre de colonnes de `A` dans `nbc`.

# III. Création de matrices aux concours

## III.1. EDHEC 2015

L'épreuve `EDHEC 2015` commençait par l'étude d'un endomorphisme dont on donnait la matrice $C$ dans la base canonique de $\mathbb{R}^5$.

$$C = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

▶ *Compléter, à l'aide de matrices de type* `zeros` *et* `ones`*, les deux espaces laissés libres dans la commande* ***Scilab*** *suivante pour qu'elle permette de construire la matrice $C$.*

```
C = [ones(1,5);------,------;ones(1,5)]
```

```
C = [ones(1,5); ones(4,1), zeros(3,4); ones(1,5)]
```

## III.2. EDHEC 2017

L'épreuve `EDHEC 2017` contenait un exercice de réduction. On demandait d'écrire la matrice $A$ d'une application linéaire $\varphi$ dans une certaine base. On trouvait :

$$A = \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

▶ *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(------)
```

```
2 A = [1, 1/2, 1/3; 0, 1, 1; 0, 0, 1]
3 disp(A ^ n)
```

**Remarque**

- À l'`EDHEC`, il faudra vous attendre à des questions d'informatique du type :

  *« Compléter les commandes* ***Scilab*** *suivantes afin que ... »*

  C'est à double tranchant :

  × à première vue, cela paraît simple puisqu'on dispose d'un morceau de code sur lequel se reposer.

  × cela est quelque peu restrictif. Ici, on n'évalue pas votre compétence à créer des matrices de manière générale mais votre compétence à créer une matrice de la manière souhaitée par le sujet.

  Dans ce cas précis et au-delà de la création de matrices, on teste ici la bonne connaissance des commandes `ones` et `zeros` qui sont exigibles aux concours.

- Si vous n'arrivez pas à répondre de la manière souhaitée par l'énoncé (dans les espaces vides laissés) mais que vous savez le faire autrement, **il faut absolument présenter votre version** sur votre copie. Elle sera prise en compte par les correcteurs.

On répondra **toujours** aux questions **Scilab** de l'EDHEC. Deux bonnes raisons à cela :
× elles sont abordables.
× les instructions **Scilab** à utiliser sont généralement rappelées dans l'énoncé.
Ce sont donc des points qu'il faut s'obliger à prendre.

## III.3. HEC 2015

Les questions **Scilab** de l'épreuve HEC se trouvaient en fin d'énoncé (question ***11***).
On reporte ici le début de cette question.

*On considère ici le programme* ***Scilab*** *suivant :*

```
x=linspace(-2,2,400); y=(exp(-exp(-x))); plot(x,y), plot(y,x)
```

► *Le réel* 0 *fait-il partie des nombres renvoyés par la commande* `x=linspace(-2,2,400)` *?*

La commande `linspace(-2, 2, 400)` crée un vecteur de `400` éléments régulièrement espacés entre `-2` et `2`. Tout élément de ce vecteur s'écrit sous la forme :

$$-2 + k \times pas$$

où $k \in [\![0, 399]\!]$ et $pas = \dfrac{2-(-2)}{399} = \dfrac{4}{399}$.

En effet, le vecteur contient 400 points régulièrements espacés.
Cela correspond à découper l'intervalle $[-2, 2]$ en 399 sous-intervalles.
La question consiste donc à savoir si l'entier 0 peut s'écrire sous cette forme. Or :

$$-2 + k \times pas \;=\; 0 \quad\Leftrightarrow\quad k \times pas \;=\; 2 \quad\Leftrightarrow\quad k \;=\; 2 \times \frac{399}{4} \quad\Leftrightarrow\quad k \;=\; \frac{399}{2}$$

Le réel $\dfrac{399}{2}$ n'étant pas un entier, 0 n'est pas un élément du vecteur créé par la commande `linspace(-2, 2, 400)`.

► *Quel est le résultat de l'exécution de ce programme ?*

- La commande `plot(x,y)` permet d'effectuer le tracé de la courbe d'équation $y = \exp(-\mathrm{e}^{-x})$. Ce tracé s'effectue point à point.
  Le vecteur `x` contient les abscisses des points tracés.
  Le vecteur `y` contient les ordonnées correspondantes.
- La commande `plot(y,x)` permet aussi d'effectuer un tracé.
  Par rapport à la commande précédente, les rôles de `x` et de `y` ont été permutés.
  Graphiquement, cela correspond à effectuer une symétrie d'axe $y = x$.
  Mathématiquement, si la courbe initiale est celle d'une fonction bijective $f$, la symétrique représente alors la courbe de la bijection réciproque $f^{-1}$.

Les questions suivantes ne faisaient pas partie de l'épreuve.

► Commenter l'utilisation des parenthèses dans l'expression de la variable `y`.

> Les parenthèses les plus à l'extérieur sont inutiles. On pouvait tout aussi bien écrire :
>
> `y=exp(-exp(-x))`

► Commenter l'utilisation du délimiteur « ; » dans ce programme.

> Les deux premières instructions du programme sont suivies du délimiteur « ; ». Le résultat de ces instructions ne sera donc pas visible à l'écran.

## III.4. HEC 2017

L'épreuve `HEC 2017` commençait par un exercice théorique de réduction. On introduisait la matrice :

$$B = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

*On considère les instructions et la sortie **Scilab** suivantes :*

```
-->  B = [0,1,0; 1,0,0; 0,0,1];
-->  P = [1,1,0; 1,-1,0; 0,0,1];
-->  inv(P)*B*P
    ans  =
        1.    0.    0.
        0.   -1.    0.
        0.    0.    1.
```

► *Déduire les valeurs propres de $B$ de la séquence **Scilab** précédente.*

> La matrice $B$ s'exprime dans une certaine base, sous forme diagonale.
> Les valeurs propres de $B$ se lisent sur cette diagonale : 1 et $-1$.

## III.5. 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 $\ell$ 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`.

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.

- On sait : $\forall k \in \mathbb{N}^*,\ q(1, k) = 1$.

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

- On sait : $\forall \ell \geqslant k,\ q(\ell, k) = q(k, k)$. 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 $\ell$ 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 : $\forall k<\ell$ : $q(\ell,k)=q(k,k)$. 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 $(\ell,k)$ 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 $\ell\in[\![1,n]\!]$ : $q(\ell,\ell)=1+q(\ell-1,\ell)$.

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

  *(le coefficient en position $(\ell,\ell)$ 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 : $\forall k>\ell$, $q(\ell,k)=q(\ell-1,k)+q(\ell,k-\ell)$.

```
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.

Par définition, $p(n)$ est le coefficient en position $(n,n)$ de la matrice $\left(q(\ell,k)\right)_{\substack{1\leqslant\ell\leqslant n\\1\leqslant k\leqslant n}}$.

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`.

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