Vecteurs et Matrices en Scilab

1 Vecteurs

L'objet mathématique qu'est le vecteur est représenté par un tableau monodimensionnel.

1.1 Définitions

a) Plusieurs façons de définir un vecteur :

1. En définissant chacun de ses éléments :

```
--> x1 = [1 5 1 89 2];
--> x2 = [1, 5 ,1, 89, 2];
-->  y = [1; 5 ;1; 89; 2];
```

   Les deux premières instructions produisent des vecteurs lignes, la troisième produit un vecteur colonne.

2. Si on souhaite construire un vecteur dont les composantes varient d'un pas constant, on peut utiliser l'instruction **a : p : b** qui désigne le vecteur :

$$(a, a+p, a+2p, \cdots, a+np) \text{ avec } a+np \leq b,$$

   ou bien par la commande **linspace(a,b,n)** qui désigne le vecteur à n composantes :

$$(a, a+\frac{b-a}{n-1}, a+2\frac{b-a}{n-1}, \cdots, a+(n-2)\frac{b-a}{n-1}, b).$$

   Exemple :

```
--> x3 = 0:10;                     // le pas vaut par defaut 1
--> x4 = 10:-1:0;                  // le pas peut etre negatif
--> x5 = linspace(0,10,101);
```

   La variable `x3` contient le vecteur $(0, 1, 2, \cdots, 9, 10)$, `x4` contient $(10, 9, \cdots, 0)$ et le vecteur `x5` est le vecteur $(0, 1/100, 2/100, \cdots, 1)$.

b) Accès aux éléments :

```
--> x1  = [0:2:25];            // Construction du vecteur
--> x1(5)                      // Acc\`es au 5eme \'el\'ement
--> y = x1(5:12);              // Acc\'es \`a un sous-vecteur
```

`x1(5)` désigne le 5eme élément de `x1`, *i.e.* ici 8. `y` est un vecteur qui contient les éléments 5, 6, ..., 12 du vecteur `x1`, *i.e.* ici le vecteur $(8\,,10\,,12\,,14\,,16\,,18\,,20\,,22)$.

Note : les indices des tableaux commencent à 1.

1.2 Opérations sur les vecteurs

a) Transposition

Pour transposer un vecteur, on utilise $\cdot'$

```
--> y3 = x3.'; y4 = x4.'; y5 = x5.';
```

b) Conjugaison

Pour conjuguer un vecteur, on utilise '

```
-->x=[1+%i 2 3]
 x  =

    1. + i      2.    3.
-->x'
ans  =

   1. - i
   2.
   3.
```

c) Opérations élémentaires

Les opérations élémentaires comme l'addition, la soustraction et la multiplication sont applicables à des vecteurs. Evidemment les dimensions des deux vecteurs doivent concorder. Attention plus particulièrement il n'y a pas de sens à multiplier avec le signe $*$ deux vecteurs de même taille, la multiplication se fait au sens matriciel du terme.

```
--> x1 = [1 2 3 4 5];              // Vecteur ligne
--> y = [6;7;8;9;10];              // Vecteur colonne
--> x1 + 2*y';                     // Addition de deux vecteurs 1X5
--> x1*x1';                        // vec. 1X5 fois vec. 5x1 a un sens
--> x1 + 5;                        // Ajoute 5 \`a chaque composante
```

Cette dernière instruction qui ne parait mathématiquement pas très correcte (addition d'un vecteur et d'un scalaire ?), est acceptée par Scilab, il ajoute 5 à chacun des éléments de **x1**.

d) Autres opérations

Des fonctions prédéfinies sont à votre disposition pour manipuler les vecteurs :

length(v)	calcule la taille du vecteur v.
sum(v)	additionne tous les éléments du vecteur v.
prod(v)	multiplie tous les éléments du vecteur v.
max(v)	calcule le plus grand élément du vecteur v.
min(v)	calcule le plus petit élément du vecteur v.

2 Matrices

2.1 Définitions

a) Définition terme à terme

De façon similaire aux vecteurs (un vecteur n'est en fait qu'une matrice à une colonne ou une ligne) les matrices se définissent comme suit :

```
--> A = [1,2,3 ; 4,5,6 ; 7,8,9];
--> B = [1,2,3 ; 4:2:8 ; 7,8,9];
```

et même à partir de vecteurs :

```
--> v1 = [1 2 3 4 5 6];
--> v2 = [7 8 9 1 2 3];
--> A = [v1;v2]

A =

   1.     2.     3.     4.     5.     6.
   7.     8.     9.     1.     2.     3.
```

b) Quelques matrices particulières

Quelques matrices sont déjà prédéfinies, il suffit d'indiquer leur taille.

eye(n,n)	matrice identité de taille n.
zeros(n,m)	matrices de taille n×n ou n×m remplie de zeros.
ones(n,m)	matrices de taille n×n ou n×m remplie de uns.
rand(n,m)	matrices de taille n×n ou n×m remplie de nombres aléatoires.
diag(v)	matrice dont la diagonale est le vecteur v.
testmatrix(name,n)	matrice de Hilbert ou magique (name='hilb' ou 'magi').

c) Accès aux éléments

On peut accéder à un élément ou à une partie des éléments d'une matrice :

- On accède à l'élément qui se trouve sur la $i^{\text{ème}}$ ligne et la $j^{\text{ème}}$ colonne par l'instruction $A(i, j)$.
- On accède à la $n^{\text{ème}}$ ligne de A grâce à l'instruction : $A(n, :)$. Le résultat est donc un vecteur ligne.

- On accède à la n$^{\text{ème}}$ colonne grâce à l'instruction $A(:,n)$. Le résultat est donc un vecteur colonne.
  Exemple :

```
   --> A = [0:2:10 ; 1:2:11 ; 20:2:30];
   --> A(2,:)
ans =

     1.     3.     5.     7.     9.    11.

   --> A(1,2:5)
ans =

     2.
     4.
     6.
     8.
```

- Le symbole $ permet d'accéder au dernier élément d'un vecteur :

```
-->A=[1 2 3 ;4 5 6]
 A  =

    1.    2.    3.
    4.    5.    6.

-->A(:,$-1)
 ans  =

    2.
    5.
```

2.2 Opérations sur les matrices

a) Opérations élémentaires

Les opérations élémentaires comme l'addition, la soustraction et la multiplication sont applicables à des matrices sous condition de compatibilité des dimensions.

A'	Conjuguée de A
A.'	Transposée de A
inv(A)	Inverse de A
A + B	Addition de A et B
A - B	Soustraction de A et B
X*Y	Multiplication
A\ B	Equivalent à inv(A)*B
B/A	Equivalent à B*inv(A)

Evidemment on peut multiplier une matrice et un vecteur si les dimensions concordent :

```
   --> A = ones(5,2);
   --> b = ones(2,1)*4;
   --> A*b

ans =

     8.
     8.
     8.
     8.
     8.
```

b) Opérations de l'algèbre linéaire

L'ensemble des fonctions de l'algèbre linéaire élémentaires peuvent être effectuées par Scilab :

rank(A)	rang de A
det(A)	déterminant de A
trace(A)	trace de A
expm(A), **logm**(A), **sqrtm**(A)	exponentielle, logarithme et racine carrée de A.

2.3 Manipulation des lignes et colonnes des matrices

a) Suppression de ligne ou colonne

La suppression d'une colonne (resp. ligne) se fait par substitution avec une colonne (resp. ligne) vide.

```
   -->  A=[1 1 3; 4 0 6; 2,5,-1]

A =

     1.     1.     3.
     4.     0.     6.
     2.     5.    -1.

   -->  A(2,:)=[] //suppression de la 2eme ligne

A =

     1.     1.     3.
     2.     5.    -1.

   -->  A(:,3)=[] //suppression de la 3eme colonne

A =
```

```
     1.     1.
     2.     5.
```

Les instructions suivantes permettent d'ajouter une ligne et une colonne

```
   -->  A=[A;3 4] //ajout d'une 3eme ligne

A =

     1.     1.
     2.     5.
     3.     4.

   -->  A(:,3)=[7 9 12]' //ajout d'une 3eme colonne

A =

     1.     1.     7.
     2.     5.     9.
     3.     4.    12.
```

d) Commande matrix.

La commande **matrix**(x, m, n) construit une matrice de taille m×n à partir du vecteur x (x doit avoir m*n éléments) :

```
   --> x=[1 2 3 4 5 6]; matrix(x,2,3)

ans =

     1.     3.     5.
     2.     4.     6.
```

e) Commande repmat

Une autre fonction utile est la fonction **repmat** qui (comme son nom l'indique) permet de construire une matrice à partir de la répétition en ligne et en colonne d'une autre matrice. Exemple :

```
   --> C=[1 2;3 4]

C =

     1.     2.
     3.     4.

   --> repmat(C,2,3)
```

```
ans =

     1.     2.     1.     2.     1.     2.
     3.     4.     3.     4.     3.     4.
     1.     2.     1.     2.     1.     2.
     3.     4.     3.     4.     3.     4.
```

Toutes les opérations s'effectuant sur des matrices ne sont évidemment pas citées ici. Nous n'en avons indiqué que les principales pour que le lecteur comprenne le mode de fonctionnement de Scilab.

2.4 Autres fonctions

min, max	valeurs minimale et maximale des éléments
sum	somme des éléments
prod	produit des éléments
mean	valeur moyenne des éléments

Voici un exemple d'utilisation de la fonction **max**.

```
   --> A=[1 3 2 ; 10 4 6; 7 4 9];
A  =

    1.     3.    2.
    10.    4.    6.
    7.     4.    9.

-->max(A)
 ans  =

    10.

--> max(A,'r')

ans =
    10.    4.    9.

   --> max(A,'c')
 ans  =
    3.
    10.
    9.
```

2.5 Opérations terme à terme

Les opérations terme à terme constituent une particularité de Scilab bien utile à certains calculs. Comme on va le voir, il faut toutefois être très prudent avec ces opérations.

1. On peut appliquer des fonctions à chacun des termes d'une matrice :

```
   --> A=[%pi/2 %pi %pi/3 ; %pi 0 %pi/2 ; %pi 0 %pi/4]
 A  =

    1.5707963    3.1415927    1.0471976
    3.1415927    0.           1.5707963
    3.1415927    0.           0.7853982

   --> sin(A)
ans  =

    1.           1.225D-16    0.8660254
    1.225D-16    0.           1.
    1.225D-16    0.           0.7071068
```

   On peut ainsi appliquer n'importe quelle fonction à chacun des termes de A. Attention notamment aux fonctions **exp**, **log sqrt** par exemple qu'il ne faut pas confondre aux fonctions **expm**, **logm** et **sqrtm** (voir section (2.2)).

2. On peut également multiplier terme à terme deux matrices (ces matrices doivent avoir la même taille) :

```
   --> A = [1 2 3 ; 4 5 6 ; 7 8 9];
   --> B = [2 2 2 ; 2 2 2 ; 2 2 2];
   --> A.*B

ans =

     2.     4.     6.
     8.    10.    12.
    14.    16.    18.
```

   Il existe de façon similaire l'opérateur puissance terme à terme ($\cdot\hat{}$) et la division terme à terme ($\cdot\backslash$).
   Encore une fois ces opérateurs ne sont que des outils proposés par Scilab et ne correspondent pas à des opérateurs mathématiques. Il faudra donc bien réfléchir avant de les utiliser.

3. On peut également comparer terme à terme deux matrices. Avec les deux matrices de l'exemple précédent on obtient :

```
 -->A>B
 ans  =

  F F T
  T T T
  T T T
```

```
 -->sum(A>B)
ans  =

   7.         // Il y a 7 elements Aij>Bij
```

Index