Chapitre 2

Les opérations sur les matrices

2.1 La transposition

Le caractère spécial ' (prime ou apostrophe) désigne l'opération de transposition. Les commandes suivantes

```
--> A=[1  2  3;  4  5  6;  7  8  0]
--> B=A'
```

donnent les résultats

```
A =

    1    2    3
    4    5    6
    7    8    0

B =

    1    4    7
    2    5    8
    3    6    0
```

et la commande

```
--> x=[-1  0  2]'

x =

    -1.
     0.
     2.
```

L'opération de transposition transpose les matrices au sens classique. Si `Z` est une matrice complexe, alors `Z'` désigne la transposée conjuguée de `Z`.

2.2 Addition et soustraction

Les opérateurs + et - agissent sur les matrices. Ces opérations sont valides dès que les dimensions des matrices sont les mêmes. Par exemple avec les matrices de l'exemple précédent l'addition

```
   --> A+x

Dimensions ligne/colonne incohérentes
```

n'est pas valide car A est 3×3 et x est 3×1. Par contre l'opération suivante est valide :

```
   --> C=A+B

   C =

         2.     6.    10.
         6.    10.    14.
        10.    14.     0.
```

L'addition et la soustraction sont aussi définies si l'un des opérandes est un scalaire, c'est à dire une matrice 1×1. Dans ce cas le scalaire est additionné ou soustrait de tous les éléments de l'autre opérande, par exemple :

```
   --> z=x-1

   z =

       -2.
       -1.
        1.
```

2.3 Multiplication

Le symbole * désigne l'opérateur de multiplication des matrices. Cette opération est valide dès que les dimensions des opérandes sont compatibles, à savoir, le nombre de colonnes de l'opérande de gauche doit être égal au nombre de lignes de l'opérande de droite. Par exemple l'opération suivante n'est pas valide

```
   --> x*z

Dimensions ligne/colonne incohérentes
```

Par contre la commande suivante

```
--> x'*z

ans =

    4.
```

donne le produit scalaire de x et z. Une autre commande valide est la suivante :

```
--> b=A*x

b =

      5.
      8.
     -7.
```

La multiplication d'une matrice par un scalaire est bien sûr toujours valide :

```
 --> A*2

ans =

      2.    4.    6.
      8.   10.   12.
     14.   16.    0.
```

2.4 Inversion d'une matrice et division

On obtient facilement l'inverse d'une matrice carrée avec la commande `inv` :

```
--> B=inv(A)

 B  =

 - 1.7777778    0.8888889  - 0.1111111
   1.5555556  - 0.7777778    0.2222222
 - 0.1111111    0.2222222  - 0.1111111
```

```
--> C=B*A

 C  =

    1.            4.441D-16    0.
  - 2.220D-16     1.           0.
    0             0            1.
```

On notera au passage les erreurs dues à la précision finie des calculs (la matrice C devrait être égale à la matrice identité).

La « division » de matrice est implantée dans Scilab, et porte la signification suivante : l'expression `A/B` donne le résultat de l'opération AB^{-1}. Elle est donc équivalente (mathématiquement) à la commande `A*inv(B)`. Pour la division à gauche, l'expression `A\B` donne le résultat de l'opération $A^{-1}B$. Il faut donc respecter la compatibilité des dimensions des deux matrices pour que cette division ait un sens.

La division à gauche est utilisée classiquement lorsque l'on a besoin de résoudre un système linéaire. Par exemple si l'on désire résoudre le système linéaire $Ay = x$, dont les équations sont

$$\begin{aligned} y_1 + 2y_2 + 3y_3 &= -1, \\ 4y_1 + 5y_2 + 6y_3 &= 0, \\ 7y_1 + 8y_2 &= 2, \end{aligned}$$

on écrira en Scilab

```
--> y=A\x

 y  =

    1.5555556
  - 1.1111111
  - 0.1111111
```

Lorsque Scilab interprète cette expression, il n'inverse pas la matrice `A` avant de la multiplier à droite par `x`, il résout effectivement le système d'équations avec la méthode d'élimination de Gauss, ce qui est à peu près 3 fois plus rapide que si l'on avait écrit

```
--> y=inv(A)*x
```

```
  y  =

     1.5555556
   - 1.1111111
   - 0.1111111
```

car dans ce cas il faut d'abord calculer l'inverse de la matrice (3 systèmes linéaires à résoudre) puis faire une multiplication matrice-vecteur. On peut vérifier l'exactitude des résultats de la façon suivante :

```
  --> A*y-x

 ans  =

    2.220D-16
    5.551D-16
    0.
```

On notera comme pour l'exemple précédent la présence d'erreurs de l'ordre de la précision machine.

2.5 Opérations élément par élément

Les opérations usuelles sur les matrices peuvent être effectuées élément par élément ; cela revient à considérer les matrices comme des tableaux de chiffres et non plus comme des objets mathématiques de l'algèbre linéaire. Pour l'addition et la soustraction les deux points de vues sont les mêmes puisque ces deux opérations agissent déjà élément par élément pour les matrices.

Le symbole `.*` désigne la multiplication élément par élément. Si `A` et `B` ont les mêmes dimensions, alors `A.*B` désigne le tableau dont les éléments sont simplement les produits des éléments individuels de `A` et `B`. Par exemple si on définit `x` et `y` de la façon suivante :

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

alors la commande

```
   --> z=x.*y
```

produit le résultat

```
z =

      4.    10.    18.
```

La division fonctionne de la même manière :

```
--> z=x./y

z =

     0.25    0.4    0.5
```

Le symbole `.^` désigne l'élévation à la puissance élément par élément :

```
--> x.^y

ans =

     1.    32.    729.

--> x.^2

ans =

      1.     4.     9.

--> 2.^x

ans =

      2.     4.     8.
```

On notera que pour `.^` l'un des deux opérandes peut être un scalaire.

2.6 Opérateurs relationnels

Six opérateurs relationnels sont disponibles pour comparer deux matrices de dimensions égales :

<	plus petit que
<=	plus petit ou égal
>	plus grand
>=	plus grand ou égal
==	égal
<>	différent

Scilab compare les paires d'éléments correspondants ; le résultat est une matrice de constantes booléennes, le `F` (false) représentant la valeur "faux" et le `T` (true) la valeur "vrai". Par exemple

```
--> 2+2<>4

ans =

    F
```

Les opérateurs logiques permettent de voir dans une matrice la disposition des éléments vérifiant certaines conditions. Par exemple prenons la matrice

```
--> A=[1  -1  2; -2  -4  1; 8  1  -1]

A =

      1.    -1.     2.
     -2.    -4.     1.
      8.     1.    -1.
```

La commande

```
--> P=(A<0)

 P  =

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

renvoie une matrice `P` indiquant par des `T` les éléments négatifs de `A`.

2.7 Opérateurs logiques

Les opérateurs `&`, `|` et `~` désignent les opérateurs logiques "et", "ou" et "non". Il sont utilisés pour élaborer des expressions logiques. Par exemple si l'on prend la matrice de l'exemple précédent, la commande

```
--> P=(A<0) & (modulo(A,2)==0)
```

```
P =

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

permet de repérer dans `A` les éléments négatifs et multiples de 2.

2.8 Fonctions usuelles appliquées à une matrice

Les fonctions usuelles s'appliquant sur réels et complexes s'appliquent aussi élément par élément sur les matrices. Par exemple :

```
--> A=[0  1/4;  1/2  3/4]

A =

   0.     0.25
   0.5    0.75

--> cos(%pi*A)

ans =

   1.     0.7071068
   0.   - 0.7071068
```