FICHE D'INTRODUCTION À MATLAB

Variables

Les variables sont définies avec l'opérateur d'affectation =. MATLAB est un langage à typage faible (dans le sens où les conversions de type sont implicites). Les variables ne sont pas déclarées avec leur type, sauf pour les objets symboliques. Par exemple :

```
» x = 17
x =
  17
» x = 'un_texte'
x =
  un_texte
```

Afin d'afficher à l'écran un texte ou le contenu d'une variable, on peut avoirs recours à la commande (Voir l'aide de Matlab pour plus de détails)

```
fprintf()
```

Vecteur/Matrice

Comme son nom l'indique, l'intérêt de MATLAB est de pouvoir manipuler facilement des tableaux à une dimension (« vecteur » dans le vocabulaire MATLAB) ou deux dimensions (« matrices ») ou plus. Par défaut, toutes les variables sont des tableaux, MATLAB permet de faire de la programmation en tableaux. En pratique, les tableaux sont définis de la manière suivante :
Pour un vecteur ligne :

```
» A = [1 2 2 3 6]
A =
    1 2 2 3 6
```

Pour un vecteur colone :

```
» A = [1;2;2;3;6]
A =
    1
    2
    2
    3
    6
```

Pour une matrice :

```
» A = [1 1 1 1 1;2 2 2 2 2;1 5 9 7 3]
A =
    1 1 1 1 1
    2 2 2 2 2
    1 5 9 7 3
```

Manipulations des tableaux

On liste une série de commande qui s'appliquent à des matrice ou des vecteurs.

```
>> A=[1 2 3 4]; B=[6 1 0 0]; M=[1 0 0 0;0 2 0 0;0 0 3 0;0 0 0 4];

>> A+B        % Somme
A+B =
    7 3 3 4

>> 2*B        % Multiplication par un scalaire
2*B =
    12 2 0 0

>> A*M        % Multiplication matricielle
A*M =
    1 4 9 16

>> M*A        % Multiplication matricielle impossible
??? Error using ==> mtimes
Inner matrix dimensions must agree.

>> A.*B        % Multiplication terme à terme
A.*B=
    6 2 0 0

>> M^2        % Puissance (matrice carré)
M^2=
    1 0 0 0
    0 4 0 0
    0 0 9 0
    0 0 0 16

>> B.^2        % Puissance terme à terme
B.^2=
    36 1 0 0

>> A'        % Transposition
A'=
    1
    2
    3
    4
```

Notons que la très grande majorité des fonctions Matlab (comme cos, exp...) opèrent sur des tableaux (dans leurs intégralité).

Commande de fabrication de tableaux

```
>> A = eye(n);        % Matrice identité de taille n.

>> A= ones(n,m);        % Matrice n×m remplie de 1.

>> A= zeros(n,m);        % Matrice n×m remplie de 0.

>> A= linspace(a,b,n);         % Vecteur ligne formé des n valeurs
                               % équiréparties entre a et b.

>> D = [16 3 2 13;5 10 11 8;9 6 7 12;4 15 14 1]
D =
  16 3 2  13
  5 10 11 8
  9 6  7  12
  4 15 14 1

>> D(2,3)         % Valeur de D aux coordonnées (2,3)
ans =
  11

>> D(1:3,3:4)         % Sous matrice de D composée des lignes 1 à 3 et
                     % des colones 3 à 4.
ans =
  2  13
  11 8
  7  12


>> D(4,:)         % 4e ligne de D
ans =
  4 15 14 1
```

L'indexation des tableaux commence à 1, ce qui est la convention utilisée pour les matrices en mathématiques mais qui est contraire à la plupart des langages de programmation où l'indexation commence à 0.

Quelques fonctions utiles

```
>> D = [16 3 2;5 10 11;9 6 7 ;4 15 14];

>> max(D)         % vecteur ligne formé du max de chaque colone.
ans =
  16 15 14

>> abs(D);         % Valeur absolue des termes de D

>> size(D)         % Taille de D.
ans =
  (4,3)

>> length(D)         % Longeur de D ( =max(size(D)) ).
ans =
  4
```

Structures de contrôle

```
>> for i=1:n         % boucle pour
    :
  instructions
    :
end

>> if (condition)         % si condition
    :
  instructions
    :
elseif (condition)         % sinon si condition
    :
else                       % sinon
    :
end

>> while (condition)         % boucle tant que
    :
  instructions
    :
end
```

Liste des conditions possibles

```
>> (i==j)        % i égal àj
>> (i<j)         % i strictement inférieur à j
>> (i>j)         % i strictement supérieur à j
>> (i<=j)        % i inférieur ou égal à j
>> (i>=j)        % i supérieur ou égal à j
>> (i~=j)        % i différent de j
```

Fonctions et scripts

Pour simplifier la programmation en Matlab, il convient d'utiliser des scripts (pour le programme principal) ainsi que des fonctions (appelée par ce programme). Dans les deux cas il s'agit de fichiers textes d'extention ".m".

- Le script est juste une suite d'instruction qui sont éxécutées par Matlab dans l'ordre.
- Les fonctions sont appelées par le script et ont une sintaxe particulière.

```
function [sortie1,sortie2,...]=nom_de_la_fonction(entrée1,entrée2,...)
    :
  instructions utilisant les entrées pour construire les sorties.
    :
  end
```

Par exemple

```
function y=puissance(x,n)
y=x.^n
end
```

Dans le script, l'appel de la fonction et effectué par exemple par

```
>> x=[-1 2 3]; n=3;
>> y=puissance(x,n)
y=
  -1 8 27
```

Il est également possible de déclarer des fonctions de mainère concises en utilisant la syntaxe suivante

```
>> f1=@(x) cos(3x+2)^2;
>> f2=@(x,y,z) (x^2-y^2)/z;
>> a=f2(4,0,2)
a=8
```