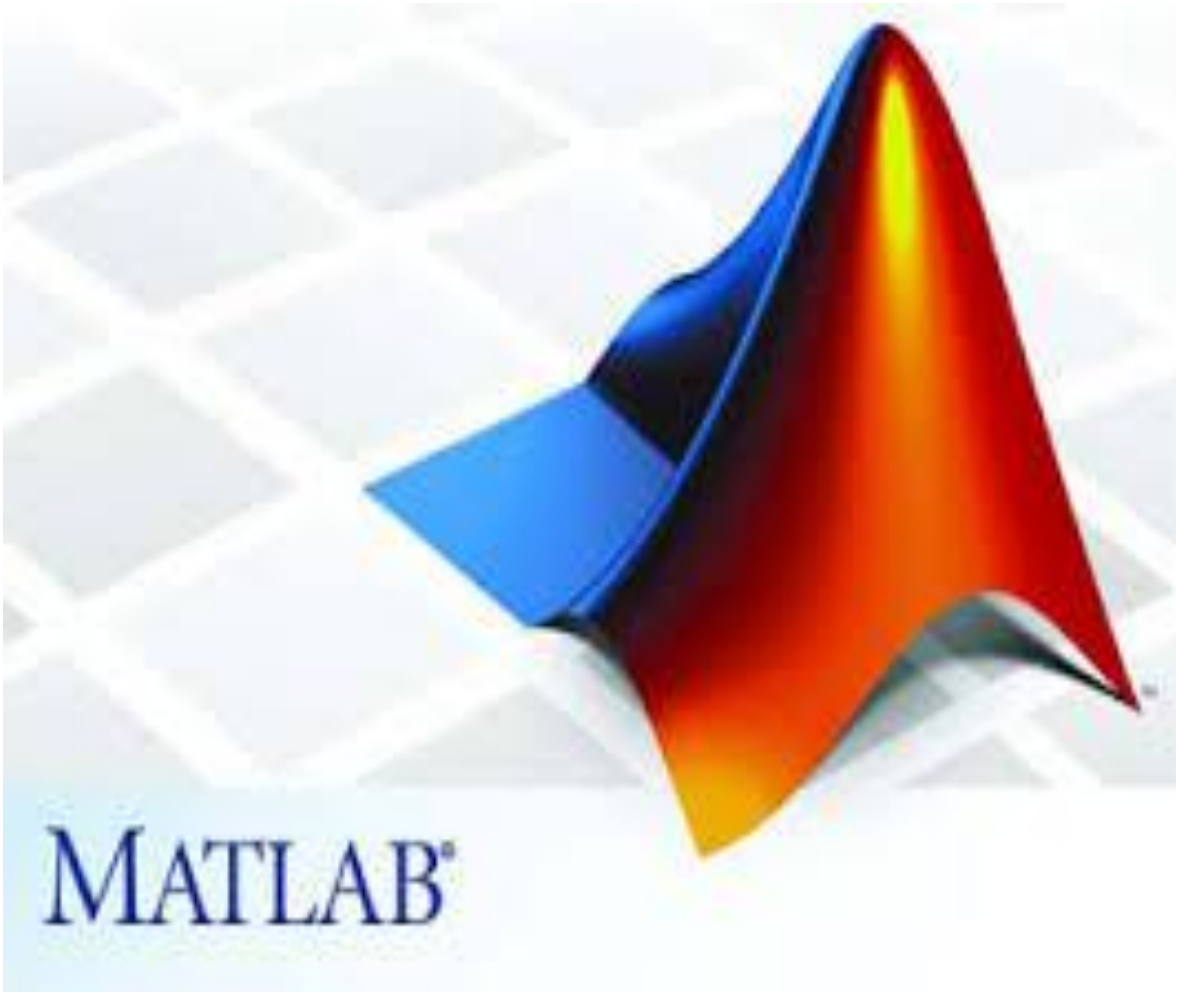
Centre Universitaire Abdelhafid-Boussouf –Mila-

Institut de science et technologie

Département de science et technologie

TP hydro-informatique



Réalisé par : Dr. Keblouti Mehdi

Table des matières

Matlab :Matrix Laboratory

Matlab Un logiciel de calcul numérique produit par MathWorks (voir le site web http://www.mathworks.com/)

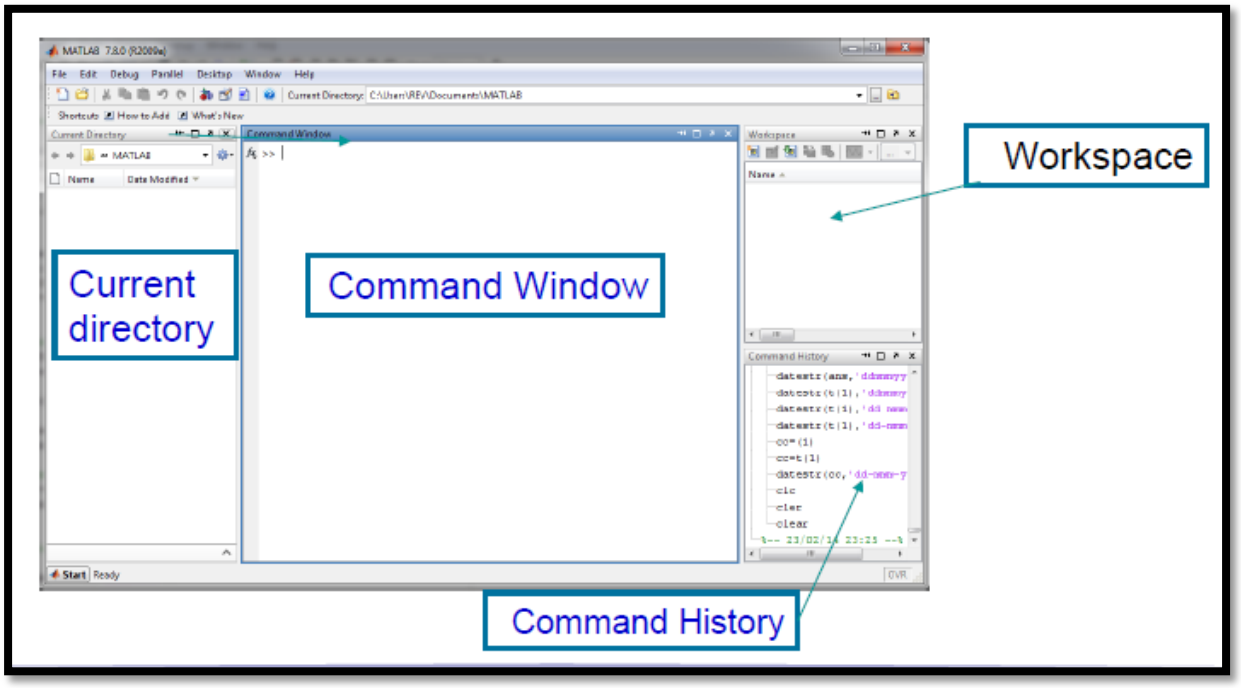
Matlab est un langage simple et très efficace, optimisé pour le traitement des matrices, d'où son nom. Pour le calcul numérique, Matlab est beaucoup plus concis que les "vieux" langages (C,Pascal, Fortran, Basic).

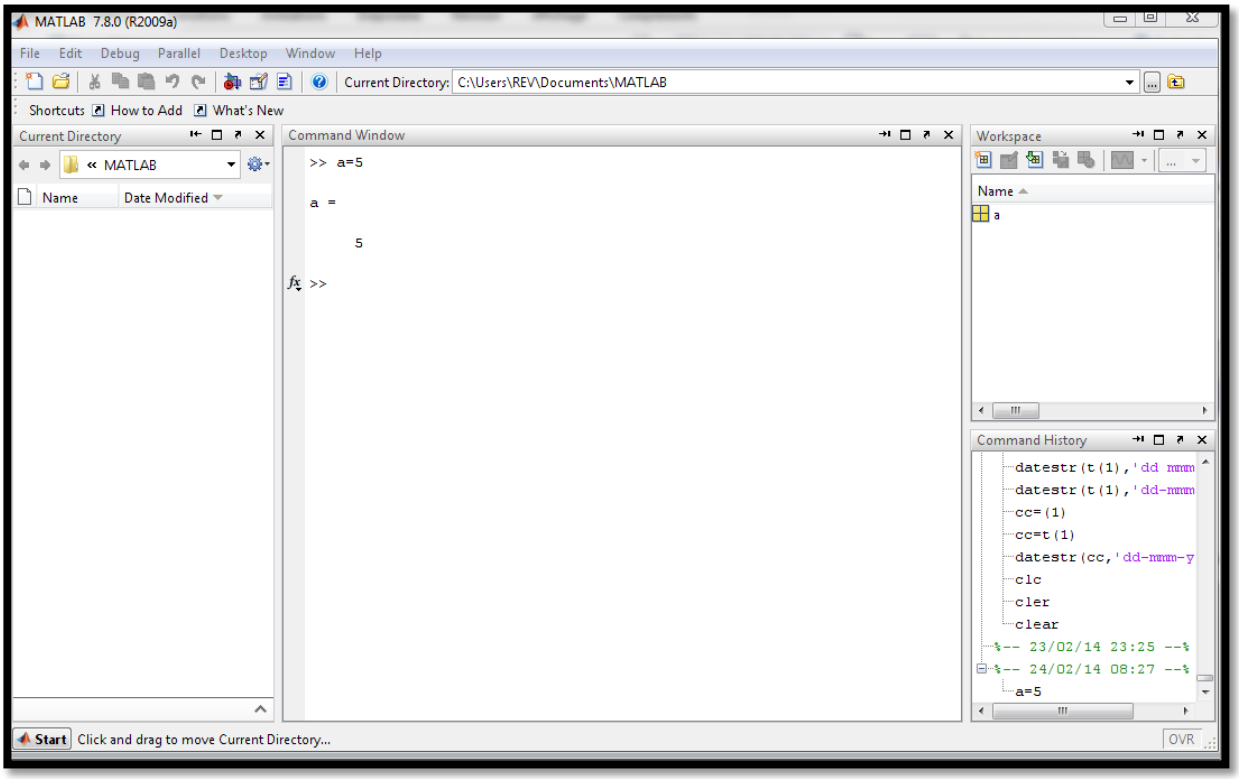
Matlab est enrichi avec des « toolbox » qui sont des ensembles de fonctions supplémentaires, profilées pour des applications particulières (traitement de signaux, analyses statistiques, optimisation, etc.)

Lancer / Arrêter

Pour Lancer Sous Windows selectionner Start-> tous les Programmes-> MATLAB > MATLAB R2009a
Pour Arreter Selectioner File -> Exit MATLAB Ou taper quit dans le menu MATLAB command window

Interface Matlab





1. Types des formats connus par MATLAB :

1.1 Format réel :

- Peut-être en n'importe quelle chaîne de lettres majuscules et minuscules aussi que des chiffres et des caractères de soulignement, mais il doit toujours commencer par une lettre
- Les noms réservés sont IF, WHILE, ELSE, END, SUM, etc

```
>> x = 5.500

x =

    5.5000
```

1.2 Format complexe :

```
>> z = 7+8j

z =

   7.0000 + 8.0000i
```

1.3 Format code ASCII ou chaîne de caractères :

String = 'Formation MATLAB'

String = Formation MATLAB

4) Quelques commandes et constantes souvent utilisées :

- La commande WHOS : permet d'avoir la liste des variables connues dans le l'espace de travail « WORKSPACE » avec leur nom, taille, nombre d'octets et classe.
- La commande CLEAR : on utilise cette commande pour supprimer une variable dans l'espace de travail.
- La commande CLC : pour effacer la page du prompt.
- La commande CLOSE : pour fermer une fenêtre « figure ».
- La constante PI, le plus grand réel et le plus petit réel représentées sous MATLAB, la constante INF et EPS.

2. Variables, vecteurs et matrices

2.1 Opérations usuelles sur les vecteurs :

2.1.1 Vecteur ligne

Utiliser les crochets [] pour contenir les nombres

```
>> v= [ 1 2 3 4 5 6 7 8]

v =

     1     2     3     4     5     6     7     8
```

2.1.2 Vecteur colonne

Pour crier un vecteur colonne on utilise ';' to separate the content

```
>> v=[2;4;6;8;10;12;14;16]

v =

     2
     4
     6
     8
    10
    12
    14
    16
```

Un vecteur ligne peut être converti en vecteur colonne en utilisant l'opérateur de transposition :

v = v'

```
v =

     2     4     6     8    10    12    14    16

>> v = v'

v =

     2
     4
     6
     8
    10
    12
    14
    16
```

2.1.3 La somme des éléments d'un vecteur :

```
>> v=[2;4;6;8;10;12;14;16];
>> sum(v)

ans =

    72
```

2.1.4 La somme cumulée des éléments d'un vecteur :

```
>> v=[2 4 6 8 10 12 14 16];
>> cumsum(v)

ans =

     2     6    12    20    30    42    56    72
```

2.1.5 La valeur moyenne d'un vecteur :

```
>> v=[2 4 6 8 10 12 14 16];
>> mean(v)

ans =

     9
```

2.2 Incrémentation

2.2.1 L'Opérateur colon (:)

Colon : est en fait un opérateur, qui génère un vecteur ligne. Ce vecteur ligne peut être considérée comme un ensemble d'indices lors de l'accès d'un des éléments d'une matrice

L'expression générale est : [start:stepsize:end]

>> V = (1:2:8)

V = 1 3 5 7

Stepsize n'est pas obligatoirement un nombre entier (ou positive)

K =(20:-2.15:10)

K = 20.0000 17.8500 15.7000 13.5500 11.4000

2.2.2 Longueur d'un vecteur :

```
>> v=[2 4 6 8 10 12 14 16];
>> length(v)

ans =

     8
```

2.2.3 Consultation, modification et suppression de quelques éléments d'un vecteur :

a. Consultation :

```
>> v=[2 4 6 8 10 12 14 16];
>> v(4)

ans =

     8
```

b. Modification

```
>> v=[2 4 6 8 10 12 14 16];
>> v(5)=[15]

v =

     2     4     6     8    15    12    14    16
```

c. Suppression

```
>> v=[2 4 6 8 10 12 14 16];
>> v(1)=[]

v =

     4     6     8    10    12    14    16
```

3. Matrices

Une matrice MATLAB est un réseau rectangulaire de nombres

- Scalaires et les vecteurs sont considérés comme des cas particuliers de matrices
- MATLAB vous permet de travailler avec toute une gamme à la fois

- Vous pouvez créer des matrices (tableaux) de toute taille en utilisant une combinaison de méthodes de création des vecteurs

- Dressez la liste des numéros à l'aide ',' pour séparer chaque colonne, puis ';' pour définir une nouvelle ligne

	A	B	C	D	E
1	Results				
2					
3	Subject	Test 1	Test 2	Test 3	
4	1	0.5	0.6	0.3	
5	2	0.3	0.33	0.75	
6	3	0.54	0.2	0.99	
7	4	0.7	0.6	0.1	
8	5	0.89	0.73	0.3	
9	6	0.2	0.9	0.94	
10	7	1	0.4	0.3	
11					

```
>> Results = [1, 0.5, 0.6, 0.3; 2, 0.3, 0.33, 0.75;
              3, 0.54, 0.2, 0.99; 4, 0.7, 0.6, 0.1;
              5, 0.89, 0.73, 0.3; 6, 0.2, 0.9, 0.94;
              7, 1, 0.4, 0.3]

Results =

    1.0000    0.5000    0.6000    0.3000
    2.0000    0.3000    0.3300    0.7500
    3.0000    0.5400    0.2000    0.9900
    4.0000    0.7000    0.6000    0.1000
    5.0000    0.8900    0.7300    0.3000
    6.0000    0.2000    0.9000    0.9400
    7.0000    1.0000    0.4000    0.3000

>>
```

Vous pouvez également utiliser des fonctions intégrées pour créer une matrice

>> A = zeros(2, 4)

créer une matrice appelée A avec 2 lignes et 4 colonnes contenant la valeur 0

>> A = zeros(5) or >> A = zeros(5, 5)

```
>> zeros(5)

ans =

     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
```

Créer une matrice appelée A avec 5 lignes et 5 colonnes

Vous pouvez également utiliser

>> ones (lignes, colonnes), exemple ones (4,3) :

```
>> ones(4,3)

ans =

     1     1     1
     1     1     1
     1     1     1
     1     1     1
```

>> rand (lignes, colonnes)

```
>> rand(3,3)

ans =

    0.8147    0.9134    0.2785
    0.9058    0.6324    0.5469
    0.1270    0.0975    0.9575
```

>> magic (n)

```
>> magic(2)

ans =

     1     3
     4     2
```

Remarque: MATLAB se réfère toujours à la première valeur que le nombre de lignes puis la seconde que le nombre de colonnes

3.1 Effacez les variables

Vous pouvez utiliser la commande "**clear all**" pour effacer toutes les variables présentes dans le workspace

Vous pouvez effacer une variable en spécifiant son nom:

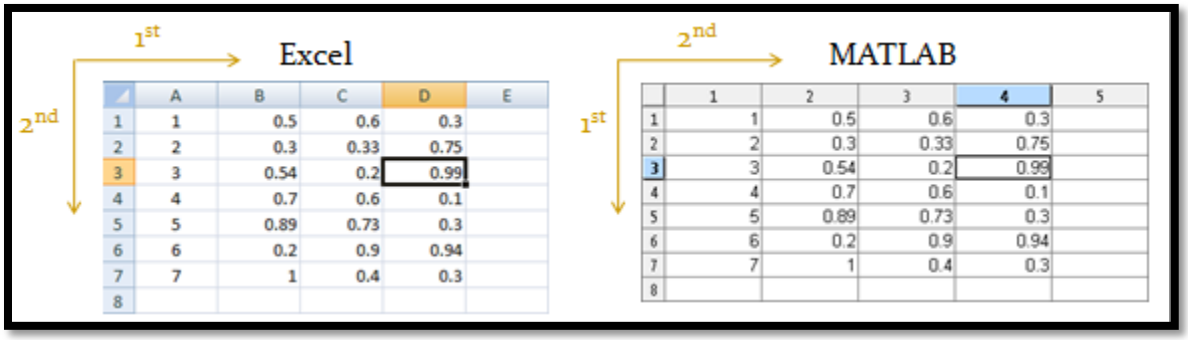
>> clear Variable_Name

3.2 Accès aux éléments des matrices

- Un élément est un numéro unique au sein d'une matrice ou d'un vecteur;
- Pour accéder aux éléments d'un type de matrice: Le nom de matrices suivie par des parenthèses contenant une référence au numéro de ligne et de colonne:

>> Nom_Variable (Numero_Ligne, Numero_Colonne)

Note : Dans Excel les valeurs sont référencées par le numéro de la colonne puis celui de la ligne. Dans Matlab c'est l'inverse.



Pour acceder au résultat de l'exemple

Dans Excel (Colonne, Ligne): D3

Dans MATLAB (Ligne, Colonne): >> results(3, 4)

3.3 Modification des éléments d'une matrice

L'élément référencé peut également être modifié

>> A (2,2) A(2,2) = [15]

ans = 6

```
A =

     1     2     3     4
     5     6     7     8
    10    -2    -9     5
```

```
A =

     1     2     3     4
     5    15     7     8
    10    -2    -9     5
```

3.3.1 Accès aux lignes d'une matrice

Vous pouvez également accéder à plusieurs valeurs à partir d'une matrice en utilisant le symbole:

Pour accéder à toutes les colonnes d'une ligne tapez (:)

>> Nom_Variable (Numero_ligne,:)

```
A =

     1     2     3     4
     5     6     7     8
    10    -2    -9     5
```

```
>> A(2,:)

ans =

     5    15     7     8
```

3.3.2 Accès aux colonnes d'une matrice

Pour accéder à toutes les lignes d'une colonne tapez >> Nom_Variable (:, Numero_Colonne)

```
A =

     1     2     3     4
     5     6     7     8
    10    -2    -9     5
```

```
>> A(:,3)

ans =

     3
     7
    -9
```

3.3.3 Modification des lignes ou colonnes d'une matrice

Ces méthodes de référence peuvent être utilisées pour modifier les valeurs de multiples éléments d'une matrice

Pour modifier toutes les valeurs d'une ligne ou d'une colonne à la valeur zéro

3.3.3.1 Modification d'une colonne :

```
A =

     1     2     3     4
     5     6     7     8
    10    -2    -9     5
```

```
>> A(:,3)=[0]

A =

     1     2     0     4
     5    15     0     8
    10    -2     0     5
```

3.3.3.2 Modification d'une ligne :

```
A =

     1     2     3     4
     5     6     7     8
    10    -2    -9     5
```

```
>> A(1,:)=[0]

A =

     0     0     0     0
     5    15     0     8
    10    -2     0     5
```

NOTE: Si vous écrasez une valeur unique, les données saisies doivent être de la même taille que la partie de la matrice à écrasé.

3.3.3.3 Accès aux plusieurs lignes et colonnes

Pour accéder à des lignes ou colonnes consécutives utiliser :
avec des points de début et de fin:

a. Plusieurs Lignes:

>> Nom de matrice (start:end, :)

```
>> A(1:2,:)

ans =

     0     0     0     0
     5    15     0     8
```

b. Plusieurs Colonnes:

>> Nom_Variable(:, start:end)

```
>> A(:,3:4)

ans =

     0     0
     0     8
     0     5
```

3.3.3.4 Modification de plusieurs lignes, colonnes

Le même référencement peut être utilisé pour modifier plusieurs lignes ou colonnes

Modification plusieurs lignes :

>> A ([2,3],:)=0

```
A =

     1     2     3     4
     5     6     7     8
    10    -2    -9     5
```

```
>> A([2,3],:)=0

A =

     1     2     3     4
     0     0     0     0
     0     0     0     0
```

Modification plusieurs colonnes :

A(:,[2,3])=0

```
A =

     1     2     3     4
     5     6     7     8
    10    -2    -9     5
```

```
>> A(:,[2,3])=0

A =

     1     0     0     4
     5     0     0     8
    10     0     0     5
```

Sauvegarde et chargement des données

Les variables qui sont actuellement dans l'espace de travail peuvent être sauvegardés et chargés en utilisant les commandes de sauvegarde et de chargement

MATLAB enregistre le fichier dans le répertoire courant

Pour enregistrer les variables :

>> save Nom_Ficher [variable variable …]

Pour charger une variable :

>> load Nom_Fichier [variable variable …]

4. Opérations sur les polynômes

Matlab répresente les polynômes sous forme d'un vecteur lignes dont les composants sont ordonnées par ordre des puissances décroissante . Un polynôme de taille n est représenté par un vecteur de taille (n+1).

Considérons le polynôme suivant :

$$f(x) = 2x^5 - x^3 + 5x^2 + 8x + 2$$

Elle représente par le vecteur ligne suivant :

F = [2 0 1 5 8 2]

Exemple 01 : trouvez les racines de polynôme suivant

$$p(x) = -x^2 + 3x + 4$$

Solution :

```
>> p = [-1 3 4 ];
>> roots(p)

ans =

   4
  -1
```

5. Les entrées est les sorties

5.1 Les formats d'affichage des réels

MATLAB dispose de plusieurs formats d'affichage des réels. Par défaut le format est le format court à 5 chiffres. Les autres principaux formats sont:

format long : format long à 15 chiffres.

format short e : format court à 5 chiffres avec notation en virgule flottante.

format long e : format long à 15 chiffres avec notation en virgule flottante.

MATLAB dispose également des formats format short g et format long g qui utilise la << meilleure >> des deux écritures à virgule fixe ou à virgule flottante. On obtiendra tous les formats d'affichage possibles en tapant help format. On impose un format d'affichage en tapant l'instruction de format correspondante dans la fenêtre de contrôle, par exemple format long. Pour revenir au format par défaut on utilise la commande format ou format short.

```
>> pi
ans =
   3.1416
>> format long
>> pi
ans =
  3.141592653589793
>>  format short e
>> pi^3
ans =
  3.1006e+01
>> format short g
>> pi^3
ans =
     31.006
>> format short
```

5.2 Affichage simple, la commande 'disp'

La commande disp permet d'afficher un tableau de valeurs numériques ou de caractères. L'autre façon d'afficher un tableau est de taper son nom. La commande disp se contente d'afficher le tableau sans écrire le nom de la variable ce qui peut améliorer certaines présentations.

```
>> A = magic(4);

>>  disp(A)

   16    2    3   13

    5   11   10    8

    9    7    6   12

    4   14   15    1

>> A
```

```
A =

   16    2    3   13

    5   11   10    8

    9    7    6   12

    4   14   15    1

>>
```

On utilise fréquemment la commande disp avec un tableau qui est une chaîne de caractères pour afficher un message. Par exemple disp('Calcul du déterminant de la matrice A'). On utilise également la commande disp pour afficher un résultat. Par exemple disp(['Le déterminant de la matrice A vaut ', num2str(det(A))]). On remarque que l'usage de la commande disp est alors un peu particulier. En effet un tableau doit être d'un type donné, les éléments d'un même tableau ne peuvent donc être des chaînes de caractères et des valeurs numériques. On a donc recours à la commande num2str (<< number to string >>) pour convertir une valeur numérique en une chaîne de caractères. Par défaut la commande **num2str** affiche 4 décimales mais il est possible de lui spécifier le nombre de décimales souhaité en second paramètre. De même il est possible de lui spécifier un format d'affichage particulier de la valeur numérique; on consultera la documentation MATLAB pour plus de détails. Attention, si la chaîne de caractères contient une apostrophe il est impératif de doubler l'apostrophe.

5.3 Lecture

La commande input permet de demander à l'utilisateur d'un programme de fournir des données. La syntaxe est var = input(' une phrase '). La phrase une phrase est affichée et MATLAB attend que l'utilisateur saisisse une donnée au clavier. Cette donnée peut être une valeur numérique ou une instruction MATLAB. Un retour chariot provoque la fin de la saisie. Une valeur numérique est directement affectée à la variable var tandis qu'une instruction MATLAB est évaluée et le résultat est affecté à la variable var. Il est possible de provoquer des sauts de ligne pour aérer le présentation en utilisant le symbole \n de la manière suivante: var = input('\n une phrase : \n '). Pensez à mettre un point virgule (;) à la fin de l'instruction si vous ne souhaitez pas voir s'afficher var = .

Sous cette forme il est impossible d'avoir une donnée de type chaîne de caractères dans la mesure où MATLAB essaie d'interpréter cette chaîne de caractères comme une instruction. Si l'on souhaite saisir une réponse de type chaîne de caractères on utilise la syntaxe var = input(' une phrase ','s'). Signalons qu'un retour chariot (sans autre chose) initialise la variable var au tableau vide []. Voici un exemple d'utilisation de la commande input (on suppose que la variable res contient une valeur numérique).

5.4 Impressions dirigées par format

La commande **sprintf** permet l'impression de variables selon un modèle donné. Un modèle d'édition se présente sous la forme du symbole pourcent (%) suivi d'indications permettant de composer le contenu du champ à imprimer, en particulier sa longueur en nombre de caractères. Le modèle d'édition utilisé par MATLAB est le modèle d'édition du langage C. La syntaxe de la commande **sprintf** est:

sprintf(format, variables)

où

- variables est le nom des variables à imprimer suivant le modèle d'édition spécifié dans ;
- format; format est le format d'édition. Il s'agit d'une chaîne de caractères contenant les modèles d'éditions des variables à imprimer.

6. Les Instructions de Contrôle : FOR, WHILE et IF

Les commandes de contrôle des boucles d'itération, suivent une syntaxe assez semblable aux langages de programmation classique du style C ou FORTRAN.

Ce sont les boucles et les branchements. Les boucles permettent de répéter commodément une suite d'instructions ; le nombre de fois est soit connu d'avance (boucles inconditionnelles), soit déterminé au cours de l'exécution (boucles conditionnelles). Les branchements conditionnels (ou tests) permettent de choisir un traitement parmi plusieurs possibles en fonction de critères évalués lors de l'exécution.

a. Boucle FOR : parcours d'un intervalle

Une première possibilité pour exécuter une séquence d'instructions de manière répétée consiste à effectuer une boucle pour les valeurs d'un indice, incrémenté à chaque itération, variant entre deux bornes données. Ce processus est mis en œuvre par la « boucle for ».

b. Boucle FOR : parcours d'un intervalle

Syntaxe : for indice = borne_inf : borne_sup séquence d'instructions end

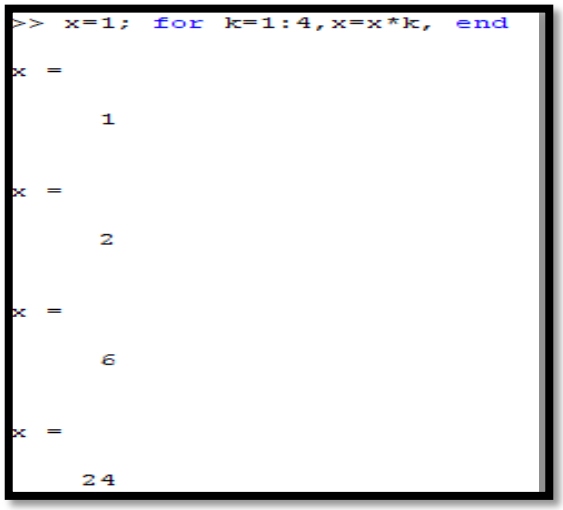
Où

- indice est une variable appelée l'indice de la boucle ;
- borne_inf et borne_sup sont deux constantes réelles (appelées paramètres de la boucle) ;
- séquence d'instructions est le traitement à effectuer pour les valeurs d'indices variant entre borne_inf et borne_sup avec un incrément de On parle du corps de la boucle.

Exemple 02 :

```
x=1;
for k=1:4
k
x=x*k
end
```

Solution



Boucles imbriquées : ces boucles peuvent être imbriquées comme dans l'exemple ci-dessous (calcul des 6 premières lignes du triangle de Pascal)

```
A=zeros(6);
for i=1:6
A(i,1)=1;
for j=2:i
A(i,j)=A(i-1,j-1)+A(i-1,j);
end
end
```

6.1 Boucle WHILE : tant que . . . faire

Une seconde possibilité pour exécuter une séquence d'instructions de manière répétée consiste à effectuer une boucle tant qu'une condition reste vérifiée. On arrête de boucler dès que cette condition n'est plus satisfaite. Ce processus est mis en œuvre par la « boucle while ».

Exemple 03 :

```
x=1;
while x<14,
x=x+5,
end
```

```
>> x=1;
     while x<14
     x=x+5,
     end

x =

      6


x =

     11


x =

     16
```

6.2 L'instruction conditionnée IF

On a parfois besoin d'exécuter une séquence d'instructions seulement dans le cas où une condition donnée est vérifiée au préalable. Différentes forms instruction conditioner existent sous matlab.

Syntaxe :

if expression logique
séquence d'instructions
end

Où:

- Expression logique est une expression dont le résultat peut être vrai ou faux ;
- Séquence d'instructions est le traitement à effectuer si expression logique est vraie.

6.2.1 Choix ventilé, l'instruction IF

Il est possible d'effectuer un choix en cascade :

Syntaxe :

if expression logique 1

séquence d'instructions 1

end

Exemple 04 : trouver les valeurs de y si :

$$y = x^2 + \sqrt{x+9}$$

$$x \geq -3$$

Solution :

```
x= input('Entrer la valeur de x:')
if x<=-3;
    y = x^2+sqrt(x+9);
end
display(y);
```

6.2.2 l'instruction 'elseif'

Syntaxe

if expression logique 1

séquence d'instructions 1

elseif expression logique 2

séquence d'instructions 2

... elseif expression logique N

séquence d'instructions N

else séquence d'instructions par défaut

end

Exemple 05 : trouver les valeurs de y si :

$y = x^2 + \sqrt{x+9}$ $\quad x > -3$

x^2 $\quad x \leq 2$

Solution :

```
x= input('Entrer la valeur de x:')
if x>2;
    y = x^2+sqrt(x+9);
else
    y = x^2;
end
display(y);
```

6.2.3 Choix ventilé, l'instruction switch

Une alternative à l'utilisation d'une séquence d'instructions conditionnées pour effectuer un choix en cascade. Il s'agit de l'instruction switch.

Syntaxe :

switch var
case cst_1 ,
séquence d'instructions 1
case cst_2 ,
séquence d'instructions 2
... case cst_N ,
séquence d'instructions N
otherwise
séquence d'instructions par défaut
end

Où

- var est une variable numérique ou une variable chaîne de caractères ;
- cst_1, . . ., cst_N, sont des constantes numérique ou des constantes chaîne de caractères;
- séquence d'instructions i est la séquence d'instructions à exécuter si var==cst_i.

Exemple 06 :

Ecrire un programme qui nous permet de dire si y est paire ou impaire

Solution

```
y = input('entre la valeur de y:');
switch mod(y,2);
    case 0
        display('y est un numéro paire');
    case 1
        display('y est un numéro impaire');
end
```

7. Plotting (Tracé)

La fonction plot (tracé) peut être utilisée de différentes façons:

>> plot(data)

>> plot(x, y)

>> plot(data, 'r.-')

Dans le dernier exemple, le style de trait est définie par:

Colour: r, b, g, c, k, y etc.

Point style: . + * x o > etc.

Line style: - -- : .-

Tapez **'help plot'** pour une liste complète des options

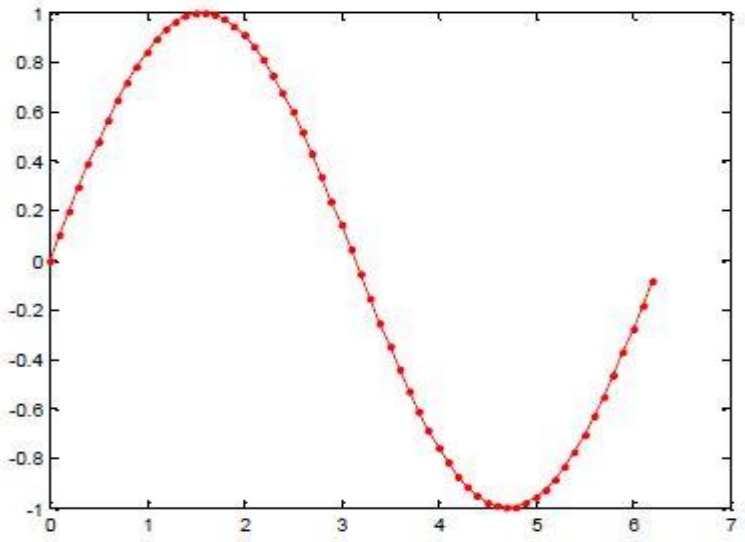
Exemple 07 :

Tracé de base

>> x = [0:0.1:2*pi]

>> y = sin(x)

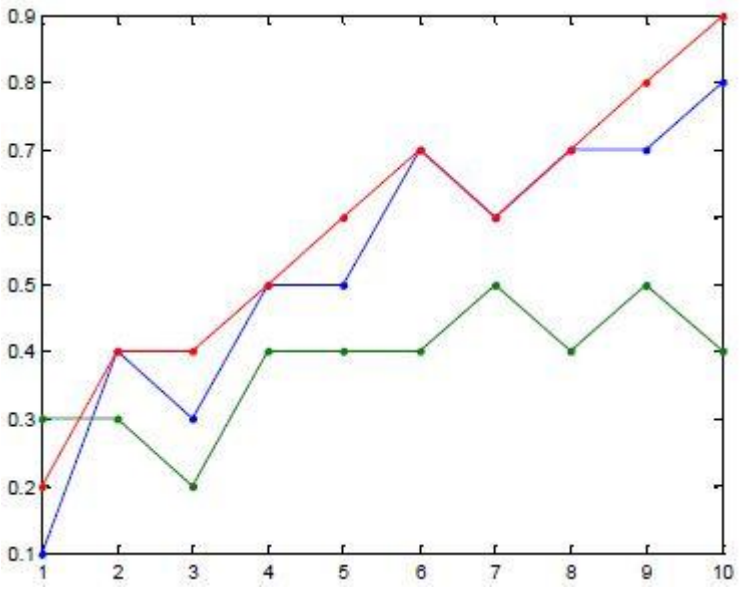
>> plot(x, y, 'r.-')



Tracé d'une matrice

MATLAB permettra de traiter chaque colonne comme un ensemble différent de données

```
results =

    0.1000    0.3000    0.2000
    0.4000    0.3000    0.4000
    0.3000    0.2000    0.4000
    0.5000    0.4000    0.5000
    0.5000    0.4000    0.6000
    0.7000    0.4000    0.7000
    0.6000    0.5000    0.6000
    0.7000    0.4000    0.7000
    0.7000    0.5000    0.8000
    0.8000    0.4000    0.9000

>> plot(results, '.-')
```



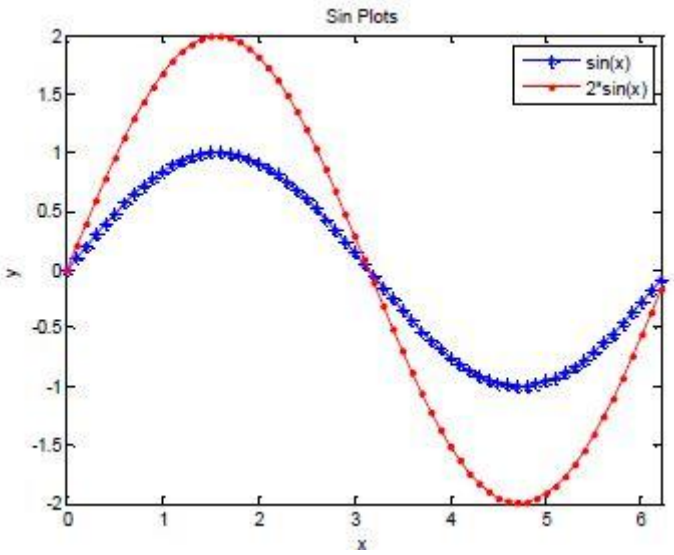
Quelques autres fonctions qui sont utiles pour créer des tracés:

hold on and hold off
title
legend
axis
xlabel
ylabel

Exemple 08 :

```
>> x = [0:0.1:2*pi];

>> y = sin(x);

>> plot(x, y, 'b*-')
```

```
>> hold on
>> plot(x, y*2, ‘r.-')
>> grid on
>> title('Sin Plots');
>> legend('sin(x)', '2*sin(x)');
>> axis([0 6.2 -2 2])
>> xlabel(‘x’);
>> ylabel(‘y’);
>> hold off
```



Exemple 09 :

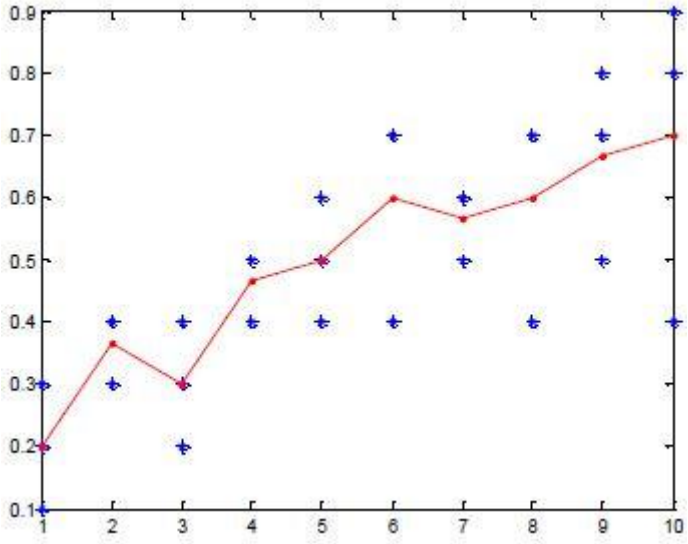
```
results =

    0.1000    0.3000    0.2000
    0.4000    0.3000    0.4000
    0.3000    0.2000    0.4000
    0.5000    0.4000    0.5000
    0.5000    0.4000    0.6000
    0.7000    0.4000    0.7000
    0.6000    0.5000    0.6000
    0.7000    0.4000    0.7000
    0.7000    0.5000    0.8000
    0.8000    0.4000    0.9000

>>
```

>> results = rand(10, 3)

>> plot(results, 'b*')

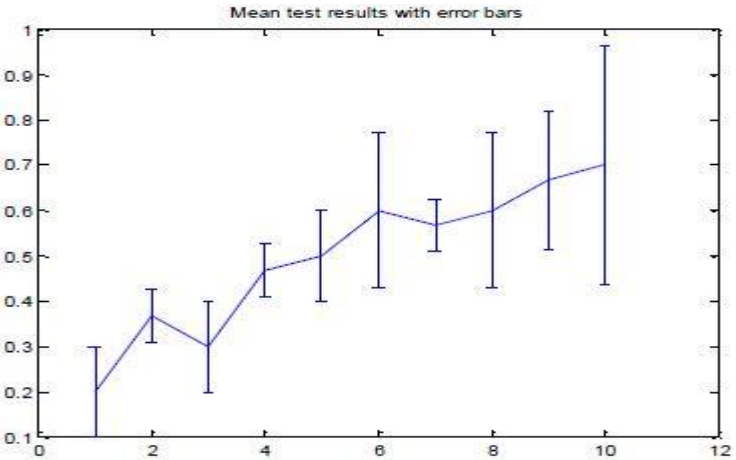
>> hold on

>> plot(mean(results, 2), ‘r.-’)



Tracé error bar

>> errorbar(mean(data, 2), std(data, [], 2))



Vous pouvez fermer toutes les graphes en cours en utilisant ‘close all’

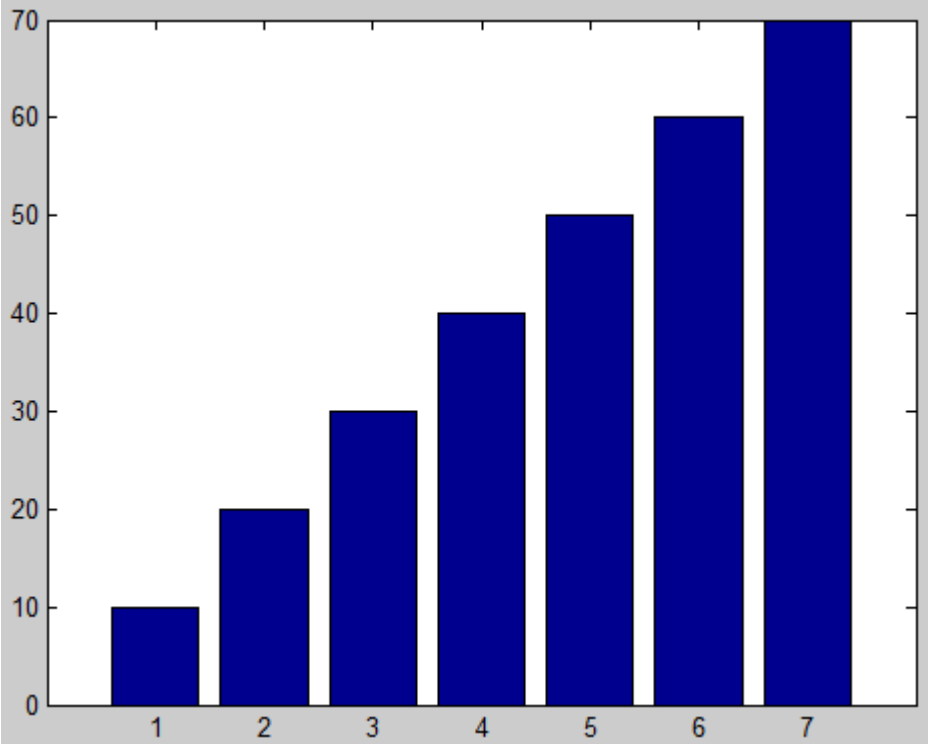
La commande Bar : permet de présenter le graphe sous forme des bars

Exemple 10 :

>> x = [1 2 3 4 5 6 7];

>> y = [10 20 30 40 50 60 70];

>> bar(x,y)

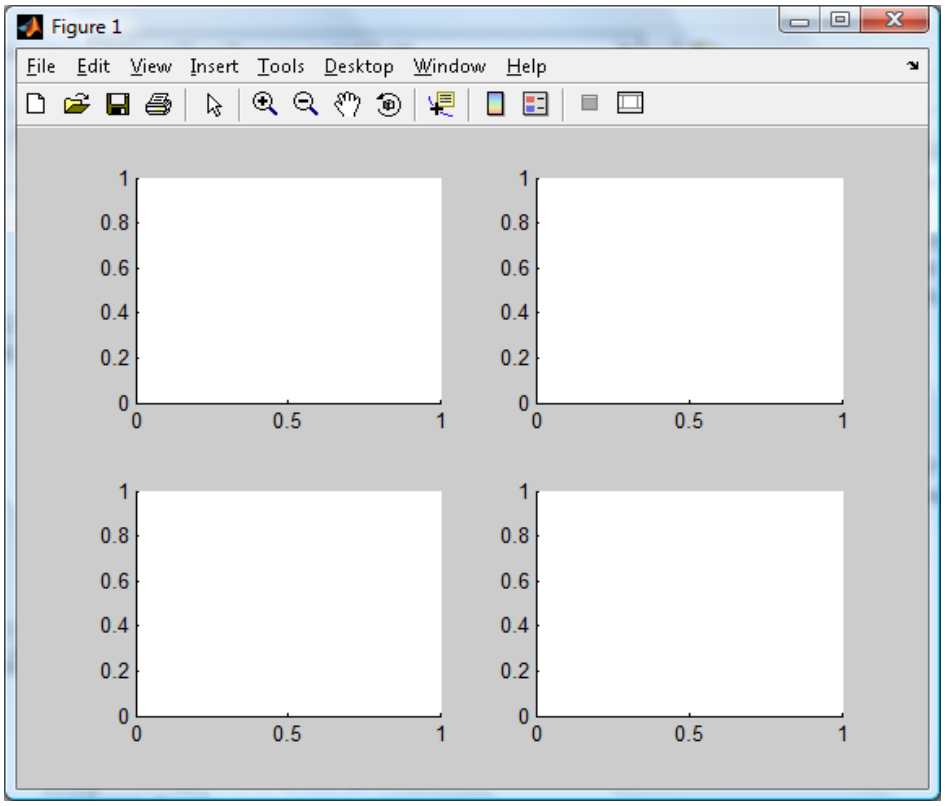


La commande Subplot :

- ✓ De multiple tracés (plots) peuvent être placé dans la même fenêtre
- ✓ Cette tache peut être effectuer avec la commande subplot
- ✓ La fenêtre est divisée en une grille dont la taille est spécifiée lors de la saisie de la commande >> subplot(2,2,1)

Cela crée une grille de 2 x 2 de taille (4 tracés) et définit le tracé actuel pour le premier de ceux-ci.

subplot(2,2,1)



Références bibliographique

H.Boutaghane, 2015 : Cours initiation matlab, université badji mokhtar annaba 32 pages.

R. Dukkipati, 2010: Matlab an introduction with applications, livre , 680 pages.

Site internet

https://perso.univ-rennes1.fr/stephane.balac/matlab/node6.html

Annexe : Quelques commandes et fonctions

Gestions des commandes et des fonctions	
help	aide
what	listing du nom des M_files présents
type	impression d'un M_file
lookfor	recherche d'une entrée dans le help
which	localise les fonctions et fichiers
demo	lance la démonstration
path	défini les chemins d'accès aux fichiers et fonctions
cedit	paramètres d'édition d'une ligne de commande
version	affiche le numéro de version de MATLAB
whatsnew	affiche les fichiers README de la toolbox
info	information sur MATLAB et The MathWorks
why	renvoie une réponse aléatoire non 'neutre'

Gestion des variables et de l'espace de travail	
who	affiche les variables courantes
whos	affiche les variables courantes, format long
save	sauve l'espace de travail sur disque
load	restaure l'espace de travail à partir du disque
clear	efface les variables et fonctions de la mémoire
pack	réorganise la mémoire
size	renvoie la taille d'une matrice
length	renvoie la longueur d'un vcteur
disp	affiche une matrice de texte

Commandes système	
cd	change le directory courant
pwd	affihe le directory courant
dir, ls	liste les fichiers
delete	suppression de fichiers
getenv	renvoie la variable d'environnement
!	appelle et exécute une commande système
unix	appelle et exécute une commande système, renvoie le résultat
diary	sauvegarde le texte d'une session MATLAB

Fenêtre de commande MATLAB	
clc	efface la fenêtre de commande
home	curseur en haut de l'écran
format	définit le format d'affichage
echo	affiche les instructions exécutées par un script
more	contrôle de l'affichage paginé

Démarrer et quitter MATLAB	
quit, exit	quitter MATLAB
startup	M_file de démarrage de MATLAB
matlabrc	M_file principal de démarrage

Opérateurs sur les Matrices		Opérateurs sur les Tableaux	
+	addition	+	addition
-	soustraction	-	soustraction
*	multiplication	.*	multiplication
^	puissance	.^	puissance
/	division à droite	./	division à droite
\	division à gauche	.\	division à gauche
'	transpose conjugué		
.'	transpose		
kron	produit de Kronecker		

Opérateurs Relationnels		Opérateurs Logiques	
<	inférieur à	&	et
>	supérieur à	|	ou
<=	inférieur ou égal à	~	non
>=	supérieur ou égal à	xor	ou exclusif
==	égal à		
~=	différent de		

Caractère spéciaux	
=	assignation
[ ]	définition de matrices ou vecteurs;
	ensère les arguments de sortie des fonctions
( )	gère la priorité des opérations arithmétique
	ensère les arguments d'entrée des fonctions
.	point décimal
..	directory parent
...	indique une ligne suite
,	séparateur d'arguments ou d'instructions

invhilb	inverse de Hilbert (exact)
vander	Vandermonde
pascal	Pascal
hadamard	Hadamard
hankel	Hankel
rosser	matrice test pour le calcul des valeurs propres
wilkinson	matrice test pour le calcul des valeurs propres
gallery	deux matrices test spéciales
Manipulation de Matrices	
diag	création ou extraction de la diagonale
rot90	rotation de 90°
fliplr	retournement gauche-droit
flipud	retournement haut-bas
reshape	redimensionnement
tril	partie triangulaire inférieure
triu	partie triangulaire supérieure
.'	transposition
:	conversion matrice → vecteur
Fonctions logiques	
exist	teste l'existence d'une variable ou d'une fonction
any	vrai si un élément est vrai
all	vrai si tous les éléments sont vrais
find	cherche l'indice des éléments non nuls
isnan	vrai si l'élément n'est pas un nombre
isinf	vrai pour tout élément infini
finite	vrai pour tout élément fini
isieee	vrai si la représentation est au format IEEE
isempty	vrai pour une matrice vide
issparse	vrai pour une matrice creuse
isstr	vrai pour une chaîne de caractères
strcmp	comparaison de deux chaînes
Instruction de contrôle	
if	test conditionnel
else	complète if
elseif	complète if
end	terminaison de if, for et while
for	instruction de répétition avec compteur

while	instruction de répétition avec test
break	interrompt une boucle for ou while
return	retour
error	affiche un message et interrompt l'exécution
Instructions spécifiques	
input	indicateur d'attente d'entrée
keyboard	considère le clavier comme un fichier script
menu	génère un menu de choix pour l'utilisateur
pause	attente
function	définition de fonction
eval	exécute un chaîne de caractère
feval	exécute une fonction définie dans une chaîne
global	définit les variables comme globales
nargchk	valide le nombre d'arguments d'entrée
Textes et chaînes	
string	à propos des chaînes dans MATLAB
abs	convertit une chaîne en valeur numérique
blanks	une chaîne d'espaces
eval	évalue une chaîne contenant du code MATLAB
num2str	convertit un nombre en chaîne
int2str	convertit un nombre entier en chaîne
str2num	convertit une chaîne en nombre
isstr	vrai si l'élément est une chaîne
strcmp	comparaison de chaînes
upper	conversion en majuscule
lower	conversion en minuscule
hex2num	convertit une chaîne héxadécimale en flottant
hex2dec	convertit une chaîne héxadécimale en entier
dec2hex	convertit un entier en une chaîne héxadécimale
Mise au point (debug)	
dbstop	met un point d'arrêt
dbclear	supprime un point d'arrêt
dbcont	reprend l'exécution
dbdown	change le contexte local
dbstack	affiche qui appelle qui
dbstatus	liste des points d'arrêt
dbstep	exécute une ou plusieurs lignes