



Université Abdelmalek Essaadi  
École Nationale des Sciences Appliquées Al Hoceima



Cours d' *Informatique 3*: ***MATLAB***

# **MATLAB POUR L'INGÉNIEUR**

***AP-2***

***ENSAH-2020-2021***

Partie 1

Pr. *Amina GHADBAN*

*Amina GHADBAN*

# MATLAB POUR L'INGÉNIEUR

## Chapitre 1

# *Introduction*

## Partie 1

Prof. Amina GHADBAN

## Présentation du logiciel (1/12)

- ➔ L'utilisation de logiciel de calcul est devenue absolument indispensable depuis les années 70 dans le domaine scientifique.
- ➔ MATLAB pour MATrix LABoratory, est une application qui a été conçue afin de fournir un environnement de calcul **matriciel** simple, efficace et interactif.
- ➔ Philosophie de Matlab : l'objet le plus commun dans Matlab est la matrice.
- ➔ Matlab est un logiciel de calcul **matriciel** à syntaxe 'simple' (relativement à des langages évolués comme C, C++, ...).
- ➔ Matlab est un interpréteur de commandes : les instructions sont interprétées et exécutées ligne par ligne (pas de compilation avant de les exécuter).
- ➔ Matlab est souvent utilisé pour une analyse efficace des données et pour les simulations numériques des systèmes physiques. Il manipule des variables numériques.

## Présentation du logiciel (2/12)

- ➔ Matlab est une console d'exécution, il permet d'exécuter des fonctions, d'effectuer des opérations mathématiques, de manipuler des matrices, d'attribuer des valeurs à des variables, de tracer ou de représenter des graphiques, ...
- ➔ Matlab est constitué d'un noyau relativement réduit, capable d'interpréter puis d'évaluer les expressions numériques matricielles qui lui sont adressées :
  - soit directement au clavier depuis une fenêtre de commande;
  - soit sous forme de **séquences d'expressions** (ou **scripts**) enregistrées dans des fichiers-texte appelés *m-files* (ou *fichiers .m*) et exécutées depuis la fenêtre de commande;
  - soit sous forme de fichiers binaires appelés mex-files (ou fichiers .mex) générés à partir d'un autre compilateur (C ou fortran, ...).
- ➔ Matlab traduit en langage machine à l'exécution.

### → Environnement de développement

#### \* Éléments principaux :

- **Commande Window** : écriture des commandes puis affichage des résultats.
- **Commande History** : conserve une trace de toutes les opérations effectuées. Traçabilité et de la reproductibilité des résultats scientifiques.
- **Workspace** : contient la liste des variables connues par Matlab.
- **Current Directory** : chemin ou ramification des dossiers.
- **Éditeur pour les scripts Matlab** (fichiers .m) : pour enregistrer et exécuter ce type de fichier .m "F5".
- **Help** : Avoir de l'aide sur une instruction, une commande, ...

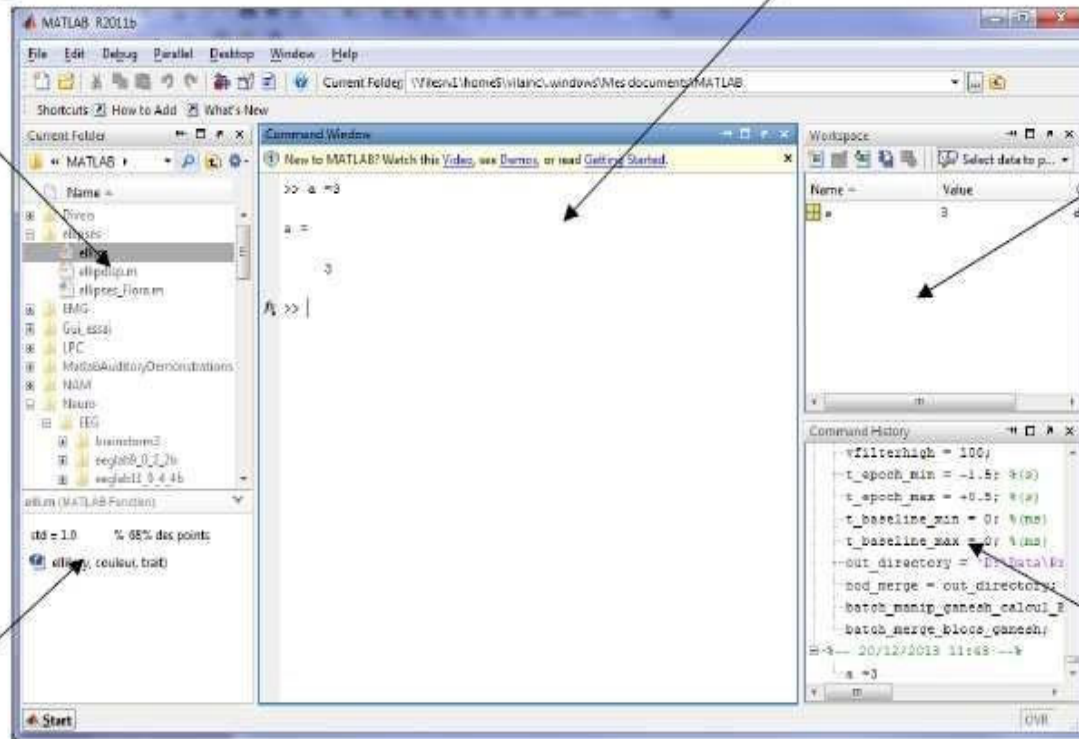
\* **Interface Matlab** : peut changer légèrement selon la version utilisée, mais les points centraux sont identiques.

# Présentation du logiciel (4/12)

Éditeur de commandes  
(pour taper les instructions)

Arborescence du dossier  
en cours

Liste des variables  
déjà définies



Historique des  
dernières commandes

Aide du fichier  
sélectionné

## Présentation du logiciel (5/12)

### ➔ Modes de fonctionnement :

- \* **mode interactif**: MATLAB exécute les instructions au fur et à mesure qu'elles sont données par l'utilisateur.
- \* **mode exécutif**: MATLAB exécute ligne par ligne un fichier ".m" (programme en langage MATLAB).

### ➔ Richesse de Matlab :

L'utilisation de boîtes à outils (toolboxes) © Popularité dans plusieurs disciplines.

### ➔ Session MATLAB :

L'interface-utilisateur de MATLAB varie légèrement en fonction de la version et du type de la machine utilisée. Elle est constituée d'une fenêtre de commande qui peut être complétée par une barre de menu et pour les versions les plus récentes de plusieurs fenêtres, affichant, l'historique de la session, la structure des répertoires, ...

## Présentation du logiciel (6/12)

### ➔ Lancer, quitter MATLAB

Dans l'environnement unix, pour lancer MATLAB on tape la commande **Matlab** sur la ligne de commande active ; dans les environnements Windows ou MacOs, il suffit de cliquer sur l'icône de l'application.

La fenêtre de commande de MATLAB s'ouvre alors et on tape les commandes ou les expressions à évaluer.

➔ L'environnement MATLAB est orienté commande comme UNIX. Une invite apparaît à l'écran et une instruction MATLAB peut être entrée. Lorsque la touche <ENTER> est enfoncée, l'instruction est exécutée et une autre invite apparaît.

### ➔ Historique

MATLAB conserve l'historique des commandes. Il est donc possible à l'aide des flèches du clavier de remonter dans la liste des instructions déjà entrées pour retrouver une instruction particulière pour la réutiliser et éventuellement la modifier avant de la réutiliser à nouveau.



### → Ouverture de Matlab

- \* Les différentes fenêtres :
- \* Dans la fenêtre d'édition de commande taper par exemple  $x = -35$  puis visualiser les conséquences sur la fenêtre d'espace de travail et d'historique de commandes.

→ Le moyen le plus simple pour utiliser Matlab est d'écrire directement dans la fenêtre de commande, mais professionnellement, il est préférable d'utiliser les scripts.

### → Les variables

- \* La variable  $x$  a été définie très simplement sans avoir à préciser le type de donnée qu'elle contient ni sa taille.

**Rq :**  $\pi = 3.14$ ,  $\sin = 14$ , il faut éviter l'affectation de valeurs à des variables déjà connues de Matlab, au risque de se tromper par la suite dans l'interprétation.

- \* Une variable est définie par sa classe (son type). Ex : double, char...

### ➔ Types de variables dans Matlab

- Les variables utilisées par matlab sont des types :
  - \* réel;
  - \* complexe;
  - \* chaîne de caractères;
  - \* logique;
  
- On ne peut pas déclarer le type d'une variable, il est établi automatiquement à partir des valeurs affectées à cette variable.
  
- Le type logique est associé au résultat de certaines fonctions.
  
- Les commandes **ischar(x)**, **islogical(x)** et **isreal(x)** retournent **1** si x est de type réel, logique ou chaîne de caractères respectivement, et **0** sinon.

### ➔ Types de variables dans Matlab : Exemples

```
>> x = 17;
```

```
>> y = 4 + 3i;
```

```
>> z = 'ENSAH';
```

```
>> who
```

Your variables are:

```
x y z
```

```
>> whos
```

Name	Size	Bytes	Class	Attributes
x	1x1	8	double	
y	1x1	16	double	complex
z	1x5	10	char	

```
>> ischar(x)
```

```
ans =  
0
```

```
>> ischar(z)
```

```
ans =  
1
```

```
>> isreal(x)
```

```
ans =  
1
```

```
>> isreal(y)
```

```
ans =  
0
```

```
>> isreal(z)
```

```
ans =  
1
```

## Présentation du logiciel (10/12)

### Quelques commandes usuelles de Matlab :

- \* Pour avoir une information sur une variable x : **who x**
- \* Pour effacer la variable x de l'espace de travail : **clear x**.
- \* Pour effacer toutes les variables de l'espace de travail : **clear all**.
- \* Pour lister tous les m-fichiers dans le répertoire courant : **what**.
- \* Pour lister tous les fichiers du répertoire courant : **dir/ls**.
- \* Pour afficher le répertoire courant : **pwd**.
- \* Pour avoir toutes les variables connues : **who**.
- \* Pour avoir toutes les variables connues avec plus de détails : **whos**.
- \* Pour effacer la fenêtre de commande : **clc**.
- \* le prompt matlab '»' qui indique que matlab attend des instructions.
- \* ...

### Fonctions et commandes

Certaines fonctions de MATLAB ne calculent pas de valeur numérique ou vectorielle, mais effectuent une action sur l'environnement de la session en cours. Ces fonctions sont alors appelées **commandes**.

Les commandes sont caractérisées par le fait que leurs arguments (lorsqu'ils existent) ne sont pas placés entre parenthèses. Les autres fonctions se comportent de façon assez semblable aux fonctions mathématiques et la valeur qu'elles calculent peut être affectée à une variable.

### ➔ Aide en ligne - help -

MATLAB comporte un très grand nombre d'opérateurs, de commandes et de fonctions.

Une aide en ligne efficace peut être utilisée en tapant les commandes suivantes :

- help permet d'obtenir de l'aide et donne une liste thématique;
- help nom de fonction donne la définition de la fonction désignée et des exemples d'utilisation ;

Exemple:    **help cumsum**

## Information sur le cours (3/3)

### ➔ Téléchargement et installation de Matlab :

- \* Version d'évaluation (courte durée)

- \* Site : [www.getinto/pc](http://www.getinto/pc)

- \* Version payante

- \* Alternatives open source à Matlab (très ou partiellement compatible avec Matlab)

et accessibles à partir de leurs sites :

- Scilab;

- Octave;



Cours d' *Informatique 3* : *MATLAB*

# MATLAB POUR L'INGÉNIEUR

*CP-2*

*ENSAH-2020-2021*

Partie 1

Prof. Amina GHADBAN





Cours d' *Informatique 3* : *MATLAB*

# MATLAB POUR L'INGÉNIEUR

## Chapitre 2

# *Vecteurs et Matrices*

Partie 1

Pr. Amina GHOU MID

## Manipulations des variables : Calculs simples (1/14)

- ➔ MATLAB est un logiciel utilisé plus pour faire du calcul numérique.
- ➔ MATLAB dispose désormais de fonctions intégrées pour résoudre les problèmes nécessitant l'analyse des données, le traitement du signal, l'optimisation et plusieurs autres types de calculs scientifiques. Il contient également des fonctions pour les graphiques et l'animation 2D et 3D.
- ➔ Matlab gère les nombres entiers, réels, complexes, les chaînes de caractères ainsi que les tableaux de nombres de façon transparente.
- ➔ On assigne une valeur au nom de la variable avec l'instruction '='.
- ➔ Matlab conserve en permanence en mémoire les variables créées.
- ➔ Les variables créées sont affichées dans la fenêtre 'workspace' de l'interface graphique.
- ➔ La commande 'who' en ligne de commande permet d'avoir la liste de ces variables en mode texte.
- ➔ La commande 'clear all' permet la suppression des variables créées.

## Manipulations des variables : Calculs simples (2/14)


- ➔ Dans Matlab, on peut saisir les commandes dans l'espace ou la zone de commande, puis le logiciel les exécute (comme ce que fait une calculatrice).
- ➔ Lorsqu'on souhaite écrire un code (programme) complet, il est très souhaitable d'utiliser les scripts (m-files).
- ➔ Les membres réels en Matlab s'écrivent avec un point pour séparer les deux parties entière et décimale (non pas avec une virgule).
- ➔ On déclare une variable avant de l'assigner une valeur, puis on l'attribue directement une valeur pour la créer.
- ➔ Utilisation d'opérateurs arithmétiques classiques usuelles '+', '-', '/', \*, ^ pour effectuer des opérations directement sur des nombres ou sur des variables après leur avoir affecté des valeurs.

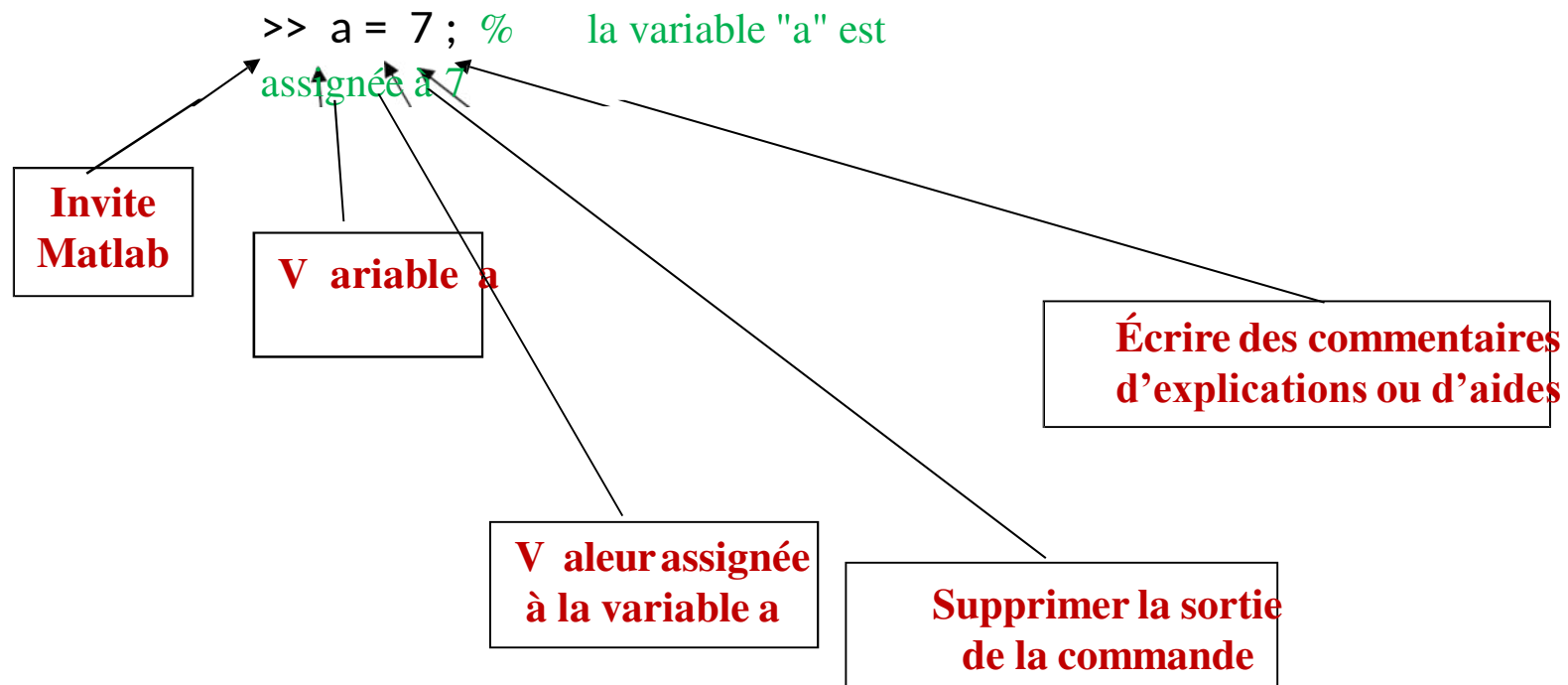
## Manipulations des variables : Calculs simples (3/14)

➔ Pas besoin d'initialiser.

➔ Attribution simple dans la fenêtre de commandes. Par exemple à la variable 'a' elle est attribuée la valeur '7'.

Il apparait donc dans la zone des variables :

Workspace		
Name ▲	Value	Min
 a	7	7



## Manipulations des variables : Calculs simples (4/14)

- ➔ MATLAB traite toutes les variables comme des matrices. Pour nos besoins, une matrice peut être considérée comme un tableau, en fait, c'est ainsi qu'elle est stockée.
  - Les vecteurs sont des formes spéciales de matrices et ne contiennent qu'une seule ligne ou une seule colonne.
  - Les scalaires sont des matrices avec une seule ligne ou une seule colonne.
  
- ➔ Type de données : une matrice (le tout est traité comme une matrice).
  - **Matrice** ( $m \times n$ ) :  $m$  lignes et  $n$  colonnes ( $m, n > 1$ ).
  - **Vecteur** ( $1 \times n$ ) :  $1$  ligne et  $n$  colonnes (vecteur ligne).
  - **Vecteur** ( $n \times 1$ ) :  $n$  ligne et  $1$  colonnes (vecteur colonne).
  - **Scalaire** ( $1 \times 1$ ) :  $1$  ligne et  $1$  colonne.
  
- ➔ Les noms des variables et de fonctions sont composés de lettres (majuscules ou minuscules) et de chiffres (ne pas commencer par un chiffre, seul le symbole underscore '\_' est autorisé).
  
- ➔ Les matrices (vecteurs, variables) peuvent être redimensionnées.

## Manipulations des variables : Calculs simples (5/14)

- ➔ Afficher le contenu des variables (et les opérations liées à la variable) en tapant simplement le nom de la variable à l'invite de commande.

```
>> a
ans =
    7
```

```
>> 8*a
ans =
   56
```

```
>> a^2 -
3*a ans
=
   28
```

- ➔ L'espace de travail est la mémoire de Matlab : on peut manipuler des variables stockées dans l'espace de travail.

```
>> b= 41;
>> c=a+b
c =
   48
```

```
>> d = -3;
>> e = a+b+c+d
e =
   93
```

```
>> f = -5;
>> g = sqrt(c+f-7)
g =
    6
```

- ➔ On peut supprimer des variables de l'espace de travail.

```
>> clear a b; % supprimer a et b de l'espace de travail
```

```
>> clea all; % supprime toutes les variables de l'espace de travail
```

```
>> a
```

## Manipulations des variables : Calculs simples (6/14)

```
>> a = 10;
```

```
>> a = 10
```

```
a =  
10
```

```
>> b = 31
```

```
b =  
31
```

```
>> c = -7
```

```
c =  
-7
```

```
>> a + b
```

```
ans =  
41
```

```
>> a*b
```

```
ans =  
310
```

```
>> d = (a + b)/c
```

```
d =  
-5.8571
```

```
>> c ^ 2
```

```
ans =  
49
```

```
>> d^2 - a
```

```
ans =  
24.3061
```

```
>> e = (7*a - 0.5*b - 0.32*c)^2/(1.85*b - d)
```

```
e =  
50.9346
```

## Manipulations des variables : Calculs simples (7/14)

```
>> pi
```

```
ans =
```

```
3.1416
```

```
>> format long
```

```
>> pi
```

```
ans =
```

```
3.141592653589793
```

```
>> log(2)
```

```
ans =
```

```
0.693147180559945
```

```
>> sin(63)
```

```
ans =
```

```
0.167355700302807
```

```
>> log(a*5*b-33*c)
```

```
ans =
```

```
7.484930283289661
```

```
>> log10(a*5*b-33*c)
```

```
ans =
```

```
3.250663919463244
```

```
>> format short
```

```
>> (exp(a - 0.5*b)*cos(4.63*b - c))^0.7
```

```
ans =
```

```
0.0208
```

```
>> tan(5*pi - 0.87*b)
```

```
ans =
```

```
3.6636
```



## Manipulations des variables : Calculs simples (8/14)

```
>> Z 1 = 3 - 5i
```

```
Z1 =
```

```
3.0000 - 5.0000i
```

```
>> Z 2 = a + b * i
```

```
Z 2 =
```

```
10.0000 + 31.0000i
```

```
>> Z 1 + 5.4 * Z 2
```

```
ans =
```

```
5.7000e+01 + 1.6240e+02i
```

```
>> real(Z 1 + 5.4 * Z 2)
```

```
ans =
```

```
57
```

```
>> imag(Z 1 + 5.4 * Z 2)
```

```
ans = 162.4000
```

```
>> angle(Z 1)
```

```
ans =
```

```
-1.0304
```

```
>> conj(Z 2)
```

```
ans =
```

```
10.0000 - 31.0000i
```

```
>> abs(Z 1)
```

```
ans =
```

```
5.8310
```

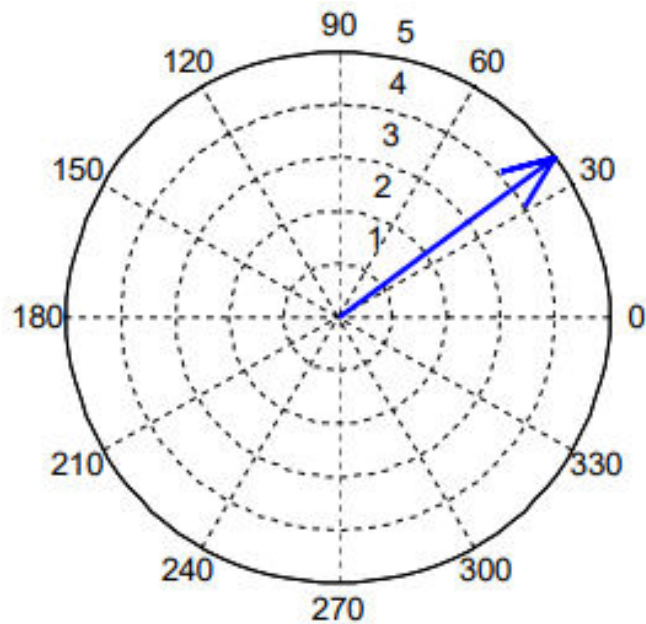
## Manipulations des variables : Calculs simples (9/14)

```
>> Z 1 = 4 + 3i
```

```
Z 1 =
```

```
4.0000 + 3.0000i
```

```
>> compass(Z  
1)
```

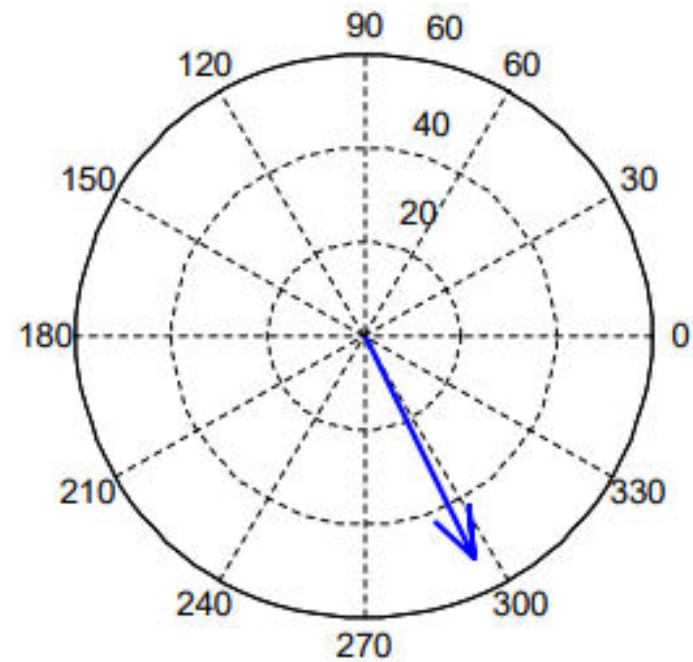


```
>> Z2 = 23 - 47i
```

```
Z 2 =
```

```
23.0000 - 47.0000i
```

```
>> compass(Z 2)
```



## Manipulations des variables : Calculs simples (10/14)

```
>> 'Bonjour'
```

```
ans =
```

```
Bonjour
```

```
>> upper('titi toto')
```

```
ans =
```

```
TITI TOTO
```

```
>> lower('TITI TOTO')
```

```
ans =
```

```
titi toto
```

```
>> upper('niveau première année stpi1')
```

```
ans =
```

```
NIVEAU PREMIÈRE ANNÉE STPI1
```

```
>> double('titi toto')
```

```
ans =
```

```
116 105 116 105 32 116 111 116 111
```

```
>> double('TITI TOTO')
```

```
ans =
```

```
84 73 84 73 32 84 79 84 79
```

```
>> char([73 84 73 32 84 79 84 79])
```

```
84 ans =
```

```
TITI TOTO
```

```
>> char(37)
```

```
ans =
```

```
%
```

## Manipulations des variables : Calculs simples (11/14)

```
>> x = 2;
>> y = 3*x + log(2+x ^ 2-cos(x))
y =
    7.8588

>> z = sin(0.36*x*pi) + (cos(3*x)) ^ 4
z =
    1.6205

>> x = 3;
>> y
y =
    11.4841

>> z
z =
    0.4405
```

```
>> z = 2*x;
>> z
z =
    6

>> A = x + y - z
A =
    8.4841

>> B = x + 2*y*i
B =
    3.0000 + 22.9681i

>> abs(B)
ans =
    23.1632
```

## Manipulations des variables : Calculs simples (12/14)

➔ Quelques fonctions utilisables dans Matlab :

***exp(x)*** : exponentielle de x

***log(x)*** : logarithme népérien de x

***log10(x)*** : logarithme en base 10 de x

***x^n*** : x à la puissance n

***sqrt(x)*** : racine carrée de x

***abs(x)*** : valeur absolue de x

***sign(x)*** : 1 si  $x > 0$  et 0 si  $x \leq 0$

***sin(x)*** : sinus de x

***cos(x)*** : cosinus de x

***tan(x)*** : tangente de x

## Manipulations des variables : Calculs simples (13/14)

➔ Q uelques fonction utilisables dans Matlab :

***asin(x)*** : sinus inverse de x (arcsin de x)

***sinh(x)*** : sinus hyperbolique de x

***asinh(x)*** : sinus hyperbolique inverse de x

***round(x)*** : entier le plus proche de x

***floor(x)*** : arrondi par défaut de x

***ceil(x)*** : arrondi par excès de x

***rem(m,n)*** : reste de la division entière de m par n

***lcm(m,n)*** : plus petit commun multiple de m et n

***gcd(m,n)*** : plus grand commun diviseur de m et n

***factor(n)*** : décomposition en facteurs premiers de n

## Manipulations des variables : Calculs simples (14/14)

➔ Q uelques fonction utilisables dans Matlab :

***conj(z)*** : conjugué de z

***abs(z)*** : module de z

***angle(z)*** : argument de z

***real(z)*** : partie réelle de z

***imag(z)*** : partie imaginaire de z

...

## Formats d'affichage (1/2)

### ➔ Formats des réels:

- MATLAB dispose de plusieurs formats d'affichage des réels. Par défaut le format utilisé est 'short' 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.

**format rational**: sous forme d'un ratio.



## Formats d'affichage (2/2)

### ➔ Formats des réels:

#### ■ Exemples:

```
>> x = [ 7/3 1.2345e-6]
```

```
2.3333 0.0000
```

```
>> format short e
```

```
2.3333e+00 1.2345e-06
```

```
>> format long
```

```
2.3333333333333333 0.000001234500000
```

```
>> format rational:
```

```
>> pi
```

```
ans =
```

```
355/113
```

## Vecteurs et Matrices (1/17)

➔ Sous Matlab, les données sont généralement définies comme des **matrices** : tableaux rectangulaires 2D composés de  $m$  lignes et de  $n$  colonnes dont chaque élément correspond à une valeur numérique.

```
>> a = 7;    %  scalaire : dimension 1x1
>> b = [ 3 0 -8 13]; %  vecteur : dimension 1x4
>> c = [ 7 -3 9 ; 5 4 2 -5]; %  matrice : dimension 2x3
>>
>> whos
```

Name	Size	Bytes	Class
a	1x1	8	double
b	1x4	32	double
c	2x3	48	double

## Vecteurs et Matrices (2/17)



Pas besoin d'initialiser le type ou les dimensions.

```
>> A = [ 6 -5 13 ; 3 10 -4 ; 2 8 -11] A
```

```
=
```

```
6   -5   13
```

```
3   10   -4
```

```
2    8   11
```

Point-virgule pour la  
ligne suivante dans la  
matrice

Crochets pour  
définir les matrices

## Vecteurs et Matrices (3/17)

➡ Accéder aux éléments d'une matrice.

```
>> A = [ 6 -5 13 ; 3 10 -4 ; 2 8 -11]
```

```
A =
```

```
     6     -5     13
```

```
     3     10     -4
```

```
     2      8     11
```

```
>> A(2,3)
```

```
ans
```

```
-4
```

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

```
ans
```

```
-5
```

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

```
ans
```

```
13
```

```
-4
```

```
11
```

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

```
ans
```

```
     6     -5     13
```

# Vecteurs et Matrices

(4/17)

➔ Accéder aux éléments d'une matrice.

```
>> B = [ 1 2 7 -3 ; 2 -6 8 -4 ; 9 0 8 1 ; 2 0 7 5]
```

```
B =  
 1  2  7 -3  
 2 -6  8 -4  
 9  0  8  1  
 2  0  7  5
```

```
>> B(:) % Concaténer les vecteurs colonnes de  
B
```

```
ans =
```

```
1  
2  
9  
2  
2  
-6  
0  
0  
7  
8  
8  
7  
-3  
-4  
1  
5
```

```
>> B(:,2:4)
```

```
ans =  
 2  7 -3  
-6  8 -4  
 0  8  1  
 0  7  5
```

```
>> B([ 2 3] ,end)
```

```
ans =  
-4  
 1
```

```
>> B(1:3,:)
```

```
ans =
```

```
 1  2  7 -3  
 2 -6  8 -4  
 9  0  8  1
```

```
>> B(1,3:4)
```

```
ans =
```

```
 7 -3
```

```
>> B(:,1:3)=[] % Supprimer les 3 premières  
colonnes
```

```
B =
```

```
-3  
-4  
 1
```

## Vecteurs et Matrices (5/17)

```
>> x = [ 1 -2 14 9]
```

```
x =
```

```
1 -2 14 9
```

```
>> transpose(x)
```

```
ans =
```

```
1
```

```
-2
```

```
14
```

```
9
```

```
>> x = [ 1; -2; 14; 9]
```

```
x =
```

```
1
```

```
-2
```

```
14
```

```
9
```

```
>> x = [ 1 -2 14 9];
```

```
>> y = [ 3 0 -4 5]
```

```
y =
```

```
3 0 -4 5
```

```
>> x+y
```

```
ans =
```

```
4 -2 10 14
```

```
>> x+y'
```

Error using +  
Matrix dimensions must agree.

```
>> x*y'
```

```
ans =
```

```
-8
```

## Vecteurs et Matrices (6/17)

```
>> A = [ 1 ; 2 ;  
3] A =  
1  
2  
3
```

```
>> B = ones(3,4)  
B =  
1 1 1 1  
1 1 1 1  
1 1 1 1
```

```
>> C = [ -5 ; -23 ; -96]  
C =  
-5  
-23  
-96
```

```
>> X = [ A , B , C]  
X =  
1 1 1 1 1 -5  
2 1 1 1 1 -23  
3 1 1 1 1 -96
```

```
>> size(X)  
ans =  
3 6
```

```
>> X(2,5)  
ans =  
1
```

```
>> X(3,6)  
ans =  
-96
```

```
>> repmat(A,2,3)  
ans =  
1 1 1  
2 2 2  
3 3 3  
1 1 1  
2 2 2  
3 3 3
```

```
>> repmat(A,1,5)  
ans =  
1 1 1 1 1  
2 2 2 2 2  
3 3 3 3 3
```

## Vecteurs et Matrices (7/17)

```
>> A = [ 6 -3; 2 10]
```

```
A =  6  -3  
     2  10
```

```
>> size(B)
```

```
ans =
```

```
 2  3
```

**Vecteur :**

```
>> a = [ 1 2 3]
```

```
;
```

```
>> a'
```

```
 1
```

```
 2
```

```
 3
```

```
>> B = [ 1 2 3
```

```
 4 5 6]
```

```
B = 1 2 3
```

```
 4 5 6
```

**Matrice :**

```
>> A = [ 1 2 ; 3 4]
```

```
;
```

```
>> A'
```

```
ans =  
 1  3
```

```
 2  4
```

```
>> magic(3)
```

```
ans =
```

```
 8  1  6
```

```
 3  5  7
```

```
 4  9  2
```

```
>> magic(5)
```

```
ans =
```

```
 17  24  1  8  15
```

```
 23  5  7  14  16
```

```
 4  6  13  20  22
```

```
 10  12  19  21  3
```

```
 11  18  25  2  9
```



### ➔ Création d'un vecteur

```
>> x = 0 : 0.5 : pi %Valeur initiale=0, Pas=0.5, Valeur finale=pi
```

```
x =
```

```
0 0.5000 1.0000 1.5000 2.0000 2.5000 3.0000
```

```
>> y = -2 : 5 %Valeur initiale=-2, Pas=1 (par défaut), Valeur finale=5
```

```
x =
```

```
-2 -1 0 1 2 3 4 5
```

### ➔ Vecteurs : Instructions 'linspace', 'logspace' :

```
>> v = linspace(0, pi, 7) %Valeur initiale=0, Valeur finale=pi, Nombre de valeurs (7, uniformement)
```

```
v =
```

```
0 0.5236 1.0472 1.5708 2.0944 2.6180 3.1416
```

```
>> w = logspace(1,5,5) %Valeur initiale=1, Valeur finale= 5, Nombre de valeurs (5 en log)
```

```
w =
```

```
10.0000 100.0000 1000.0000 10000,0000 100000,0000
```

## Vecteurs et Matrices (9/17)

```
>> X = [2 17 -7 0 5 -1]
X =
     2    17    -7     0     5    -1
```

```
>> Y = [-4 : 3 : 11]
Y =
    -4    -1     2     5     8    11
```

```
>> X+Y
ans =
    -2    16    -5     5    13    10
```

```
>> X*Y'
ans =
   -10
```

```
>> sin(X)
ans =
    0.9093  -0.9614  -0.6570   0  -0.9589  -0.8415
```

```
>> sum(Y)
ans =
    21
```

```
>> cumsum(Y)
ans =
    -4    -5    -3     2    10    21
```

```
>> Z = [2*X', 3*Y']
Z =
     4    -12
    34     -3
   -14     6
     0    15
    10    24
    -2    33
```

```
>> prod(Z)
ans =
     0  2566080
```

## Vecteurs et Matrices (10/17)

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

```
A =
```

```
1 2 3
2 4 1
-1 -2 2
```

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

```
B =
```

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

```
>> A^2
```

```
ans =
```

```
2 4 11
9 18 12
-7 -14 -1
```

```
>> A.^2
```

```
ans =
```

```
1 4 9
4 16 1
1 4 4
```

```
>> A*B
```

```
ans =
```

```
16 -1 10
12 -2 10
4 1 0
```

```
>> A.*B
```

```
ans =
```

```
-2 2 0
6 -4 2
-4 0 4
```

```
>> A/B
```

```
ans =
```

```
2.3333 0.3333 1.1667
1.3333 -2.6667 3.1667
1.0000 3.0000 -2.0000
```

```
>> A ./B
```

```
ans =
```

```
-0.5000 2.0000 Inf
0.6667 -4.0000 0.5000
-0.2500 -Inf 1.0000
```

## Vecteurs et Matrices (11/17)

```
>> A = [6 -5 13 ; 3 10 -4 ; 2 8 -11] A
```

```
=
```

```
6 -5 13
3 10 -4
2 8 -11
```

```
>> rank(A) %Rang de la matrice A
```

```
ans =
```

```
3
```

```
>> poly(A) %Polynôme caractéristique
```

```
ans =
```

```
1.0000 -5.0000 -95.0000 541.0000
```

```
>> diff(A) %[A(2,:)-A(1,:); A(3,:)-A(2,:)]
```

```
ans =
```

```
-3 15 -17
-1 -2 -7
```

```
>> C = [6 -5 13 ; 12 -10 26 ; 2 8 -11] C
```

```
=
```

```
6 -5 13
12 -10 26
2 8 -11
```

```
>> rank(C)
```

```
ans =
```

```
2
```

```
>> poly(C)
```

```
ans =
```

```
1.0000 15.0000 -190.0000 -0.0000
```

```
>> diff(C)
```

```
ans =
```

```
6 -5 13
-10 18 -37
```

### → Matrices particulières

**zeros(m, n)** : matrice  $m$  lignes et  $n$  colonnes contenant que des zéros.

**ones(m, n)** : matrice qui contient que des 1.

**eye(m, n)** : matrice identité.

**rand(m, n)** : matrice aléatoire avec des valeurs comprises entre 0 et 1.

**magic(n)** : carré magique de taille  $n$ .

.....

### ➔ Quelques opérations avec les matrices et les vecteurs :

$M * N$  : produit matricielle (M et N sont deux matrices).

$M .* N$  : produit élément-élément des deux matrices.

$M / N$  : division matricielle.

$M ./ N$  : division terme à terme des deux matrices.

$M ^ n$  : puissance matricielle d'ordre  $n$ .

$M .^ n$  : puissance élément-élément d'ordre  $n$ .

$\det(M)$  : déterminant d'une matrice M.

$\text{inv}(M)$  : inverse d'une matrice M.

$\text{size}(M)$  : nombre de lignes et de colonnes de la matrice M.

$\text{diag}(M)$  : éléments diagonaux de la matrice M.

### ➔ Quelques opérations avec les matrices et les vecteurs :

*mean(A)* : valeur moyenne d'un vecteur A.

*max(A)* : valeur maximale d'un vecteur A.

*min(A)* : valeur minimale d'un vecteur A.

*sum(A)* : somme des éléments du vecteur A.

*Cumsum(A)* : somme cumulée des éléments du vecteur A.

*prod(A)* : produit des éléments du vecteur A.

*cumprod(A)* : produit cumulée des éléments du vecteur A.

*std(A)* : écart type des éléments du vecteur A.

*sort(A)* : tri par ordre croissant des éléments du vecteur A.

*diff(A)* : vecteur de différence entre deux éléments consécutifs du vecteur A.

## Vecteurs et Matrices (15/17)

```
>> A = ones(2,5)
```

```
A =
```

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

```
>> B = zeros(3,4)
```

```
B =
```

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

```
>> C = eye(3,3)
```

```
C =
```

```
 1  0  0
 0  1  0
 0  0  1
```

```
>> D = [eye(4),zeros(4,3)]
```

```
D =
```

```
 1  0  0  0  0  0  0
 0  1  0  0  0  0  0
 0  0  1  0  0  0  0
 0  0  0  1  0  0  0
```

```
>> M = [1,-3,7,12 ; 3,6,-7,-4;2,-5,11,3;8,9,13,-6]
```

```
M = 1  -3  7  12
```

```
 3  6  -7  -4
 2  -5  11  3
 8  9  13  -6
```

```
>> diag(M)
```

```
 1  6  11  -6
```

```
>> diag(M-1)
```

```
 3  -5  13
```



## Vecteurs et Matrices (16/17)

```
>> E = [1 5 -3 ; 0 4 7 ; 2 1 3]
```

```
E =
```

```
1 5 -3
0 4 7
2 1 3
```

```
>> det(E)
```

```
ans =
```

```
99
```

```
>> F = inv(E)
```

```
F =
```

```
0.0505 -0.1818 0.4747
0.1414 0.0909 -0.0707
-0.0808 0.0909 0.0404
```

```
>> E*F
```

```
ans =
```

```
1.0000 0.0000 0.0000
0 1.0000 0
0 0 1.0000
```

```
>> F*E
```

```
ans =
```

```
1.0000 -0.0000 0
0 1.0000 0.0000
0 0.0000 1.0000
```

```
>> G = [E,-F;5*ones(3),7*rand(3)]
```

```
G =
```

```
1.0000 5.0000 -3.0000 -0.0505 0.1818 -0.4747
0 4.0000 7.0000 -0.1414 -0.0909 0.0707
2.0000 1.0000 3.0000 0.0808 -0.0909 -0.0404
5.0000 5.0000 5.0000 5.7031 6.3936 1.9495
5.0000 5.0000 5.0000 6.3405 4.4265 3.8282
5.0000 5.0000 5.0000 0.8889 0.6828 6.7025
```

## Vecteurs et Matrices (17/17)

```
>> M = [2 -5 7 -11 ; 6 -4 0 23 ; 7 -8 9 14]
```

```
M =
```

```
2 -5 7 -11
```

```
6 -4 0 23
```

```
7 -8 9 14
```

```
>> N = rand(4,2)
```

```
N =
```

```
0.4218 0.6557
```

```
0.9157 0.0357
```

```
0.7922 0.8491
```

```
0.9595 0.9340
```

```
>> V = [6 23 -1 7]
```

```
V =
```

```
6 23 -1 7
```

```
>> X = [M zeros(3) ; N 3*V' -ones(4,1) eye(4,3)]
```

```
X =
```

```
2.0000 -5.0000 7.0000 -11.0000 0 0 0
```

```
6.0000 -4.0000 0 23.0000 0 0 0
```

```
7.0000 -8.0000 9.0000 14.0000 0 0 0
```

```
0.4218 0.6557 18.0000 -1.0000 1.0000 0 0
```

```
0.9157 0.0357 69.0000 -1.0000 0 1.0000 0
```

```
0.7922 0.8491 -3.0000 -1.0000 0 0 1.0000
```

```
0.9595 0.9340 21.0000 -1.0000 0 0 0
```

```
>> size(X)
```

```
ans =
```

```
7 7
```

```
>> length(X)
```

```
ans =
```

```
7
```

## Résolution d'un système d'équation linéaire (1/4)

➔ Soit le système d'équation suivant :

$$\begin{cases} 5x - 2y + 4z = 20 \\ 3x + 2y + 7z = 7 \\ -2x + y - z = -8 \end{cases}$$

➔ Cette équation peut être écrite sous la forme :

$$\mathbf{A} * \mathbf{X} = \mathbf{B}$$

$$\mathbf{A} = \begin{pmatrix} 5 & -2 & 4 \\ 3 & 2 & 7 \\ -2 & 1 & -1 \end{pmatrix} \quad \mathbf{X} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 20 \\ 7 \\ -8 \end{pmatrix}$$

➔ La résolution de cette équation est :

- Première méthode :  $\mathbf{X} = \mathit{inv}(\mathbf{A}) * \mathbf{B}$

- Deuxième méthode :  $\mathbf{X} = \mathbf{A} \setminus \mathbf{B}$

## Résolution d'un système d'équation linéaire (2/4)

```
>> A=[5 -2 4;3 2 7;-2 1 -1] %matrice A liée au système
```

```
A=
```

```
5 -2 4
3 2 7
-2 1 -1
```

```
>> B = [20 ; 7 ; -8] %matrice B colonne des éléments qui sont après '='
```

```
B =
```

```
20
7
-8
```

```
>> X = inv(A)*B %1er méthode :solution de l'équation 'matrice X, colonne'
```

```
X =
```

```
2
-3
1
```

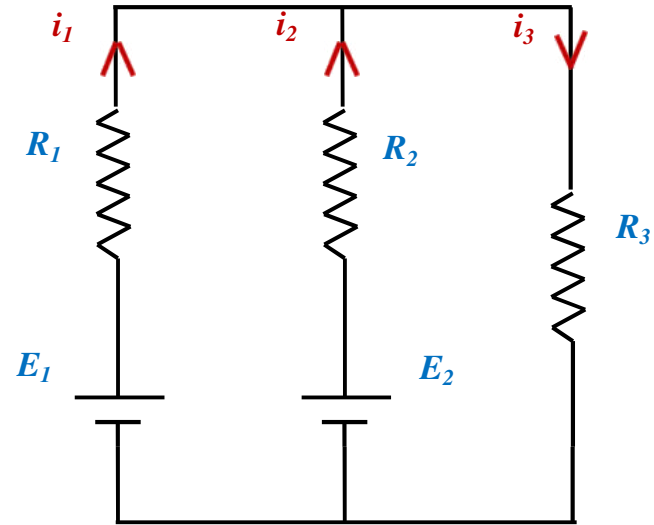
```
>> X = A\B %2ème méthode : la solution de l'équation 'matrice X, colonne'
```

```
X =
```

```
2.0000
-3.0000
1.0000
```

## Résolution d'un système d'équation linéaire (3/4)

➔ Soit le circuit électrique suivant :



➔ Le système d'équation lié à ce circuit est :

$$\begin{cases} I_1 + I_2 - I_3 = 0 \\ -R_1 I_1 + R_2 I_2 = E_2 - E_1 \\ R_2 I_2 + R_1 I_1 = E_2 \end{cases}$$

## Résolution d'un système d'équation linéaire (4/4)

➡ Pour les valeur :  $R_1 = 20\Omega$ ,  $R_2 = 50\Omega$ ,  $R_3 = 100\Omega$ ,  $E_1 = 20V$  et  $E_2 = 70V$  , le système s'écrit :

$$\begin{cases} I_1 + I_2 - I_3 = 0 \\ -20 I_1 + 50 I_2 = 50 \\ 50 I_2 + 100 I_3 = 70 \end{cases}$$

```
>> A = [1,1,-1; -2,5,0; 0,5,10] %matrice A
```

```
A =  
 1  1 -1  
-2  5  0  
 0  5 10
```

```
>> B = [0 ; 50 ; 70] %matrice B
```

```
B =  
 0  
50  
70
```

```
>> X = inv(A)*B %solution de l'équation : matrice X
```

```
X =  
-0.5  
 0.8  
 0.3
```

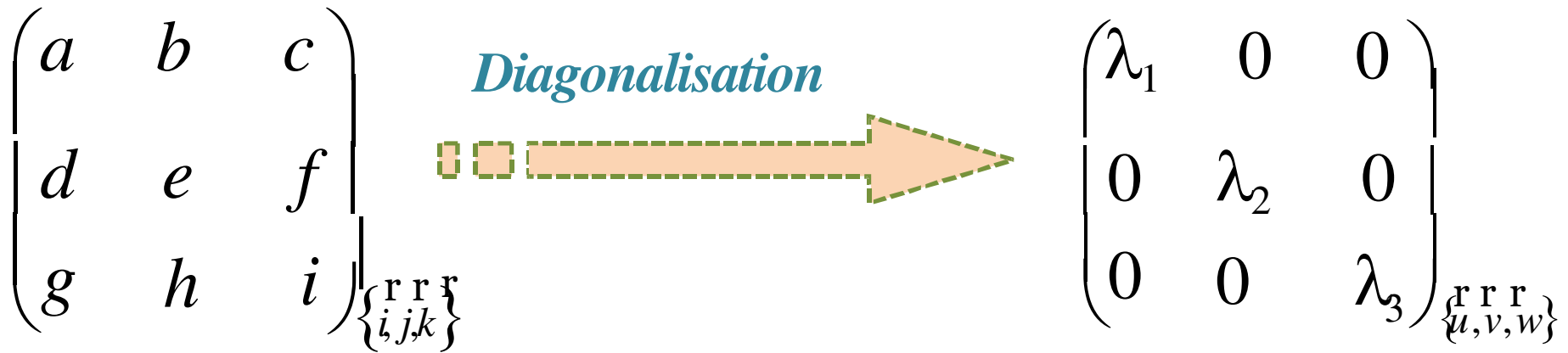
```
>> X = A\B %autre écriture de la solution de l'équation : matrice X
```

```
X =  
-0.5  
 0.8  
 0.3
```

%On doit changer le sens de  $i_1$  dans le circuit électrique (électrocinétique)

**Diagonalisation d'une matrice :  
Valeurs propres, Vecteurs propres**

(1/4)



**Diagonalisation** → il faut trouver les **valeurs propres**  $\lambda_1, \lambda_2, \lambda_3$  et les **vecteurs propres**  $u, v, w$ .

$$P = \begin{pmatrix} \begin{matrix} u \\ \downarrow \\ x_1 \\ y_1 \\ z_1 \end{matrix} & \begin{matrix} v \\ \downarrow \\ x_2 \\ y_2 \\ z_2 \end{matrix} & \begin{matrix} w \\ \downarrow \\ x_3 \\ y_3 \\ z_3 \end{matrix} \end{pmatrix} \quad D = \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix}$$

## Diagonalisation d'une matrice : (2/4) Valeurs propres, Vecteurs propres

```
>> A = [4 -2 ; 1 1] % une matrice carrée A  
=  
 4  -2  
 1   1
```

Avec le calcul, Pour trouver les valeurs propres de la matrice A, il faut résoudre le système suivant:

$$\det(A - \lambda \cdot I) = 0; \text{ où } I \text{ est la matrice identité.}$$

La résolution de l'équation, donne les deux valeurs propres suivantes:  $\lambda_1 = 2$  et  $\lambda_2 = 3$ . Pour trouver les vecteurs propres associés aux valeurs propres  $\lambda_1$  et  $\lambda_2$ , il faut résoudre le système suivant (avec  $i = 1, 2$ ):

$$\begin{pmatrix} 4 & -2 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \lambda_i \begin{pmatrix} x \\ y \end{pmatrix}$$

On trouve les deux vecteurs propres liés à  $\lambda_1$  et à  $\lambda_2$  respectivement:

$$\mathbf{r}_u = \begin{pmatrix} 0,8944 \\ 0,4472 \end{pmatrix} \quad \mathbf{r}_v = \begin{pmatrix} 0,7071 \\ 0,7071 \end{pmatrix}$$

On peut vérifier que :  $\mathbf{A} = \mathbf{P} * \mathbf{D} * \mathbf{P}^{-1}$  &  $\mathbf{A}^n = \mathbf{P} * \mathbf{D}^n * \mathbf{P}^{-1}$

Où P est la matrice des vecteurs propres et D est la matrice diagonalisable de A.



## Diagonalisation d'une matrice : (3/4) Valeurs propres, Vecteurs propres

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

```
A =  
    4   -2  
    1    1
```

```
>> [P,D] = eig(A)
```

```
P =  
    0.8944    0.7071  
    0.4472    0.7071
```

```
D =  
    3    0  
    0    2
```

**Vérification :**

```
>> P*D*inv(P)
```

```
ans =  
    4   -2  
    1    1
```

```
>> M = [3,2,4;-1,3,-1;-2,-1,-3]
```

```
M =  
    3    2    4  
   -1    3   -1  
   -2   -1   -3
```

```
>> poly(M)
```

```
ans =  
    1.0000   -3.0000   -0.0000    4.0000
```

```
>> [P,D]=eig(M)
```

```
P =  
   -0.7071    0.8165   -0.8165  
   -0.0000    0.4082   -0.4082  
    0.7071   -0.4082    0.4082
```

```
D =  
   -1.0000         0         0  
         0    2.0000         0  
         0         0    2.0000
```

```
>> M^5
```

```
ans =  
   -95    160   -94  
   -80    112   -80  
    47   -80    46
```

```
>> P*D^5*inv(P)
```

```
ans =  
  -95.0000  160.0000  -94.0000  
  -80.0000  112.0000  -80.0000  
   47.0000  -80.0000   46.0000
```

## Diagonalisation d'une matrice : (4/4) Valeurs propres, Vecteurs propres

B =

```
5  1  -1
2  4  -2
1  -1  3
```

```
>> [VecteursPropres, ValeursPropres] = eig(B)
```

VecteursPropres =

```
0.0000  0.7071  0.7071
0.7071  0.7071  0.0000
0.7071  0.0000  0.7071
```

ValeursPropres =

```
2.0000    0    0
0  6.0000    0
0    0  4.0000
```

Vérification :

```
>> P = VecteursPropres;
```

```
>> D = ValeursPropres;
```

```
>> P*D*inv(P)
```

ans =

```
5.0000  1.0000 -1.0000
2.0000  4.0000 -2.0000
1.0000 -1.0000  3.0000
```

## Factorisation LU (1/3)

### *Triangularisation*

$$\begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{pmatrix} = \begin{pmatrix} a_1 & b_1 & c_1 & d_1 \\ 0 & e_1 & f_1 & g_1 \\ 0 & 0 & h_1 & i_1 \\ 0 & 0 & 0 & j_1 \end{pmatrix} * \begin{pmatrix} a_2 & 0 & 0 & 0 \\ b_2 & c_2 & 0 & 0 \\ d_2 & e_2 & f_2 & 0 \\ g_2 & h_2 & i_2 & j_2 \end{pmatrix}$$

*Triangulaire Supérieure*

*Triangulaire Inférieure*

## Factorisation LU (2/3)

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

B =

```
5  1  -1
2  4  -2
1  -1  3
```

```
>> [LB UB]=lu(B)
```

LB =

```
1.0000  0  0
0.4000  1.0000  0
0.2000 -0.3333  1.0000
```

UB =

```
5.0000  1.0000 -1.0000
0  3.6000 -1.6000
0  0  2.6667
```

**Vérification :**

```
>> LB*UB
```

ans =

```
5.0000  1.0000 -1.0000
2.0000  4.0000 -2.0000
1.0000 -1.0000  3.0000
```

## Factorisation LU (3/3)

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

```
C =
```

```
 3  1  2
 2  3 -1
 3  1  1
```

```
>> [LC UC] = lu(C)
```

```
LC =
```

```
 1.0000  0  0
 0.6667  1.0000  0
 1.0000  0  1.0000
```

```
UC =
```

```
 3.0000  1.0000  2.0000
 0  2.3333 -2.3333
 0  0 -1.0000
```

```
>> LC*UC
```

```
ans =
```

```
 3.0000  1.0000  2.0000
 2.0000  3.0000 -1.0000
 3.0000  1.0000  1.0000
```

**Attention**

```
>> UC*LC
```

```
ans =
```

```
 5.6667  1.0000  2.0000
-0.7778  2.3333 -2.3333
-1.0000  0 -1.0000
```



# MATLAB POUR L'INGÉNIEUR

*CP-2*

*ENSAH-2020-2021*

Partie 1

Pr. *Amina GHADBAN*



Cours d' *Informatique3*: **MATLAB**

# MATLAB POUR L'INGÉNIEUR

## Chapitre 3

# *Représentations Graphiques*

Partie 1

Pr. Amina GHADBAN

## Graphisme avec Matlab (1/1)

- ➔ Matlab est une console d'exécution, il permet d'exécuter des fonctions, d'effectuer des opérations mathématiques, de manipuler des matrices, d'attribuer des valeurs à des variables, **de tracer ou de représenter des graphiques, ...**
- ➔ La représentation graphique aide l'utilisateur à résoudre et à profiler un problème lié à un phénomène naturel ou physique.
- ➔ Pour les représentations graphiques, Matlab s'appuie sur des données numériques discrètes rangées dans des matrices ou des vecteurs aux dimensions compatibles.
- ➔ L'utilisation des instructions de dessins pour tracer les courbes (1D, 2D, 3D) complétées par des attributs et des arguments optionnels.
- ➔ La visualisation des résultats s'effectue dans une fenêtre graphique avec possibilité de zoom, d'impression, de modification de couleurs, de polices, de types de trait, ...



## Représentation 1D (1/21)

- ➔ La commande **plot** permet de tracer un ensemble de points de coordonnées  $(x_i, y_i)$ ,  $i=1, \dots, N$ .
- ➔ La syntaxe est **plot(x,y)** où x est le vecteur contenant les valeurs  $x_i$  en abscisse et y est le vecteur contenant les valeurs  $y_i$  en ordonnée (x et y doivent être de même dimension).
- ➔ Par défaut, les points  $(x_i, y_i)$  sont reliés entre eux par des segments de droites → **Le pas d'échantillonnage doit être très petit et la qualité du tracé dépend du nombre de points.**  
Par défaut :
  - Couleur bleu ;
  - Trait continu ;
  - Largeur de trait égale à 0,5.
- ➔ Les fenêtres graphiques possèdent un grand nombre d'attributs. Dans la commande plot, on peut spécifier par exemple :
  - la couleur d'une courbe ;
  - le style de trait ;
  - le symbole attribué à chaque point  $(x_i, y_i)$  ;

## Représentation 1D (3/21)

- ➔ La commande **grid on** permet d'obtenir un quadrillage de la figure. Pour l'annuler, on utilise **grid off**.
  
- ➔ Par défaut, les fonctions de tracés effacent systématiquement le graphique précédent. Pour éviter ceci, on peut :
  - garder l'ancienne figure et ouvrir une nouvelle fenêtre graphique par l'instruction **figure**. Matlab affecte à chaque fenêtre un numéro *n* selon l'ordre. L'instruction **gcf** (get current figure) retourne le numéro de la fenêtre active. Pour fermer une fenêtre graphique, il suffit d'employer la commande **close(n)**.
  
  - utiliser l'instruction **hold on**, qui donne la possibilité de tracer et de superposer plusieurs courbes de natures différentes (avec **plot**, **bar**, **hist**, etc ...), dans une même fenêtre graphique. Pour la désactiver, on utilise **hold off**.
  
  - Pour effacer la fenêtre graphique courante, on utilise la commande **clf**. Pour effacer une fenêtre particulière, il suffit d'en préciser le numéro en argument de la fonction.

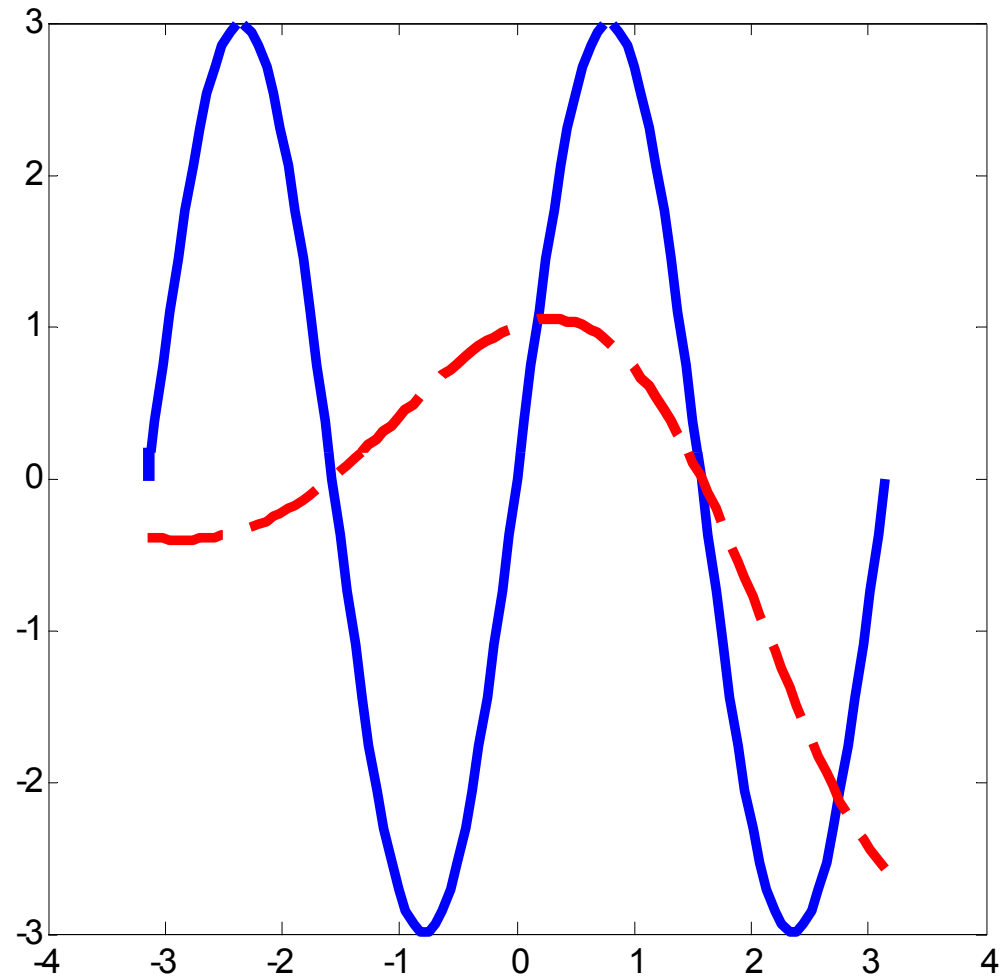
## Représentation 1D (4/21)

- ➔ L'instruction **ginput(N)** permet d'identifier les coordonnées de N points dans la fenêtre graphique (on peut faire **[x y] = ginput(N)**).
- ➔ Lorsqu'on utilise une superposition de plusieurs courbes dans une même fenêtre graphique, il est préférable de faire appel à l'instruction **legend** (légende) pour reconnaître et différencier les courbes empilées.
- ➔ L'instruction **gtext** permet d'insérer un texte à l'emplacement indiqué par un clic de souris.
- ➔ Les étiquettes **xlabel** et **ylabel** sont utilisées pour mettre un texte en légende sur les axes x et y respectivement de la fenêtre graphique.
- ➔ La commande **title** permet de donner un titre à la figure.
- ➔ Pour sauvegarder une figure d'une fenêtre graphique dans un fichier sous divers formats d'images, on peut se servir de l'instruction **print**.

## Représentation 1D (5/21)

### ➔ Exemple

```
x = -pi : pi/50 : pi; y =  
3*sin(2*x);  
z = cos(x).*exp(0.3*x);  
plot(x,y,'linewidth',3)  
hold on  
plot(x,z,'-r','linewidth',3)
```



## Représentation 1D (6/21)

### ➔ Exemple

```
x = -pi : pi/50 : pi;  
y = 3*sin(2*x);  
z = cos(x).*exp(0.3*x);  
plot(x,y,'linewidth',4);  
hold on  
plot(x,z,'-r','linewidth',4);  
xlabel('x');  
ylabel('y(x) , z(x)');  
title('y(x) = 3*sin(2x) , z(x)=cos(x)*e^{0.3*x}');  
legend('y(x)','z(x)');  
grid on
```

