Université Badji Mokhtar - Annaba
Faculté des sciences de l'ingéniorat

Département d'Electrotechnique

Domaine : Sciences et technologie
Année Universitaire : 2020/2021
Filière : Electrotechnique

- **Semestre: 3**
- **Unité d'enseignement: UE Découverte**
- **Matière: Simulation en Matlab/Simulink**
- **Code : UED 2.1**
- **VHS: 22h30 (Cours: 1h30)**
- **Crédits: 02**
- **Coefficient: 02**

Parcours :

Master 02 : Electrotechnique Industrielle
Master 02 : Commandes électriques
Master 02 : Réseaux électriques

Simulation en MATLAB/Simulink

Objectifs de l'enseignement:

Connaitre les logiciels de simulation, être capable de reproduire un système électro-énergétique en vue de son étude et sa simulation et sa programmation.

Connaissances préalables recommandées:

Outille informatique ;Notions de simulation et programmation.

Contenu de la matière:

Introduction générale

Chapitre I :Matlab

I.1 Introduction
I.2Opérations mathématiques
I.3 Simulation et Programmation à l'aide de Matlab (opérations simples).
I.4 Graphiques

Chapitre II:Simulink

II.1 Introduction
II.2 . Simulation des systèmes composés électriques.

Chapitre IIIPSIM

Mode d'évaluation:

Examen: 100%.

Références bibliographiques:

- Introduction à MATLAB ◦ J.T. Lapresté. ◦ Ellipses, 2009
- MATLAB pour l'ingénieur ◦ A. Biran, M. Breiner ◦ Pearson Education, 2004
- Initiation à MATLAB ◦ O. LOUISNARD ◦ Ebook, 2009
- Apprendre et maitriser Matlab ◦ M. Mokhtari ◦ Springer, 1998
- Introduction à Matlab J.-T. Lapresté (Ellipses, 1999)
- Mastering Matlab 6 D. Hanselman B. Littlefield (Prentice Hall, 2001)
- Apprendre et maîtriser Matlab M. Mokhtari A. Mesbah, (Springer, 1997)
- Solving problems in scientific computing using Maple and Matlab W. Gander, J. Hrebicek (Springer, 1995, second edition)
- Numerical Methods Using Matlab G. Lindfield J. Penny (Prentice Hall, 2nd edition : 2000)
- M. Mokhtari et A. Mesbah : Apprendre et maitriser Matlab. Springer.
- B. Laurent : Documents pour les TPs de Matlab.

Introduction générale

Ce document est un guide simplifiédes logiciels de simulation(Matlab ;Simulink ;PSIM…) Les notions de base sont présentées de façon simple, des exemples seront illustréspour permettre aux étudiants de démarrer rapidement et être capable de reproduire un système électro-énergétique en vue de son étude et sa simulation. Ce cours est répartie en trois chapitres logiciel de Simulation en Matlab ; Simulink ; PSIM .

MATLAB est un :

- Logiciel de calcul numérique dont le nom provient de MATRIX LABORATORY
- Un langage de calcul scientifique basé sur le type de variables matricielles ;
- Logiciel destiné au calcul scientifique tels le traitement de signal, l'automatique, etc ;
- logiciel qui intègre des fonctionnalités graphiques de grande qualité en 2D ou 3D ;
- environnement puissant, complet et facile à utiliser ;
- un interpréteur: les instructions sont interprétées et exécutées ligne par ligne…...

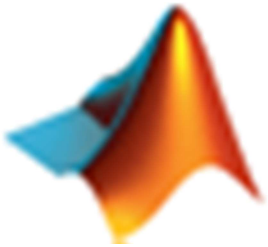
SIMULINK est une plateforme de modélisation et de simulation de systèmes dynamiques. Il offre un environnement de développement graphique et une bibliothèque de blocs qui permettent de simuler divers systèmes de contrôle, communication, traitement de signaux.

PSIM est un logiciel de simulation pour l'électrotechnique et l'électronique de puissance . Il est disponible sur Internet sur le site de POWERSIMTECH à l'adresse www.powersimtech.com.Le logiciel PSIM est constitué de trois programmes :
- SIMCAD : Dessin du schéma ;
- PSIM: Simulation (Calcul des variables) ;
-SIMVIEW : Tracé des courbes.

CHAPITRE I

I. Matlab

I.1 Introduction



Ce document est un guide simplifié de MATLAB et Simulink. Les notions de base sont présentées de façon simple pour permettre aux étudiants de démarrer rapidement et les exemples seront illustrés.

Origine:C'estun logiciel commercialisé par la société américaine Mathworks

◦ Création : fin des années 1970

◦ Commercialisation : 1984

- ✓ **MATLAB** : Matrix Laboratory

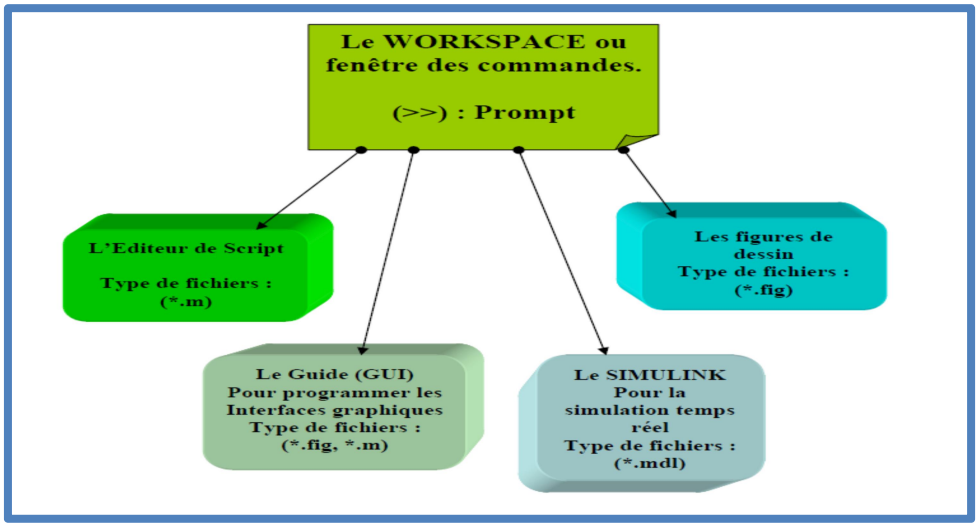
MATLAB est un :

- Logiciel de calcul numérique dont le nom provient de MATRIX LABORATORY
- Un langage de calcul scientifique basé sur le type de variables matricielles ;
- Logiciel destiné au calcul scientifique tels le traitement de signal, l'automatique, etc ;
- logiciel qui intègre des fonctionnalités graphiques de grande qualité en 2D ou 3D ;
- environnement puissant, complet et facile à utiliser ;
- un interpréteur: les instructions sont interprétées et exécutées ligne par ligne…...

MATLAB possède son propre langage et comprend :

- Une vaste gamme de bibliothèques de fonctions spécialisées (Toolboxes)
- Simulink, un environnement puissant de modélisation basée sur les schémas-blocs et de simulation de systèmes dynamiques linéaires et non linéaires
- Des bibliothèques de blocs Simulink spécialisés (Blocksets) 4) D'autres modules dont un Compilateur, un générateur de code C, un accélérateur,...
- Un ensemble d'outils intégrés dédiés au Traitement du Signal : le DSP Workshop.

✓ **Type de fichiers en Matlab :**

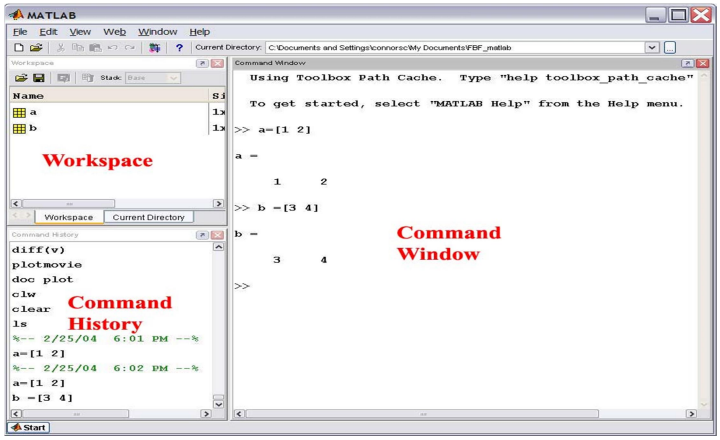


✓ **C'est un outille pour mathématiciens et les ingénieurs techniques**

Appliqué à différents domaines scientifiques tels le traitement de signal, l'automatique, etc.,

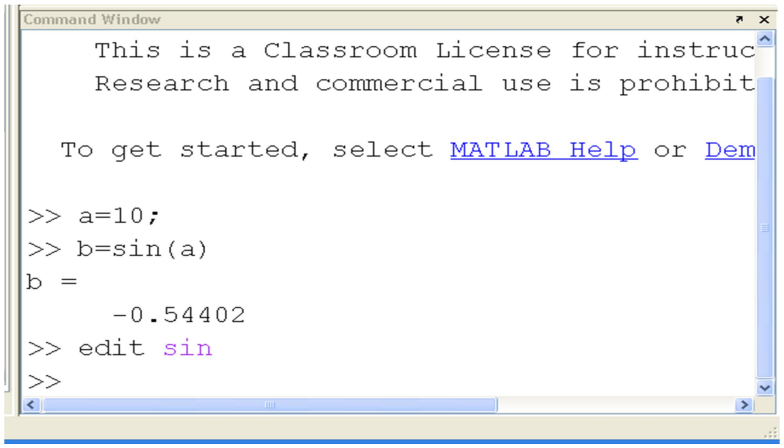
- **Le bureau**

Lorsque vous démarrez Matlab, vous verrez le bureau qui, par défaut, est composé de la fenêtre de commande, de l'historique des commandes et de l'espace de travail.



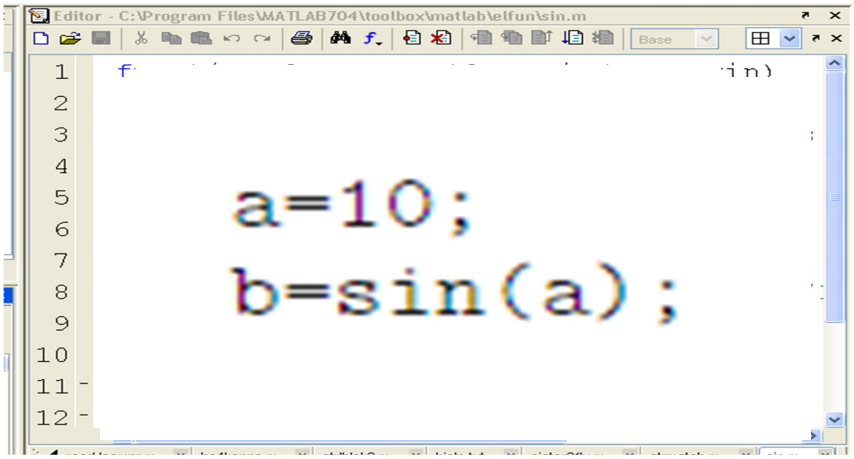
- **La fenêtre de commande**

Souvent, vous travaillerez sur la ligne de commande dans la fenêtre de commande. La fenêtre de commande vous permet de taper des commandes directement et de voir les résultats immédiatement. Par exemple au >> , définissez une variable "a" en tapant.



- **La fenêtre de l'éditeur**

Si non , vous pouvez écrire un programme dans la fenêtre de l' éditeur. La fenêtre de l'éditeur est un traitement de texte spécialement conçu pour les commandes Matlab Ce programme peut consister en une ou plusieurs commandes à exécuter. Une fois le programme enregistré, vous pouvez simplement taper le nom du fichier et toutes les commandes seront exécutées.

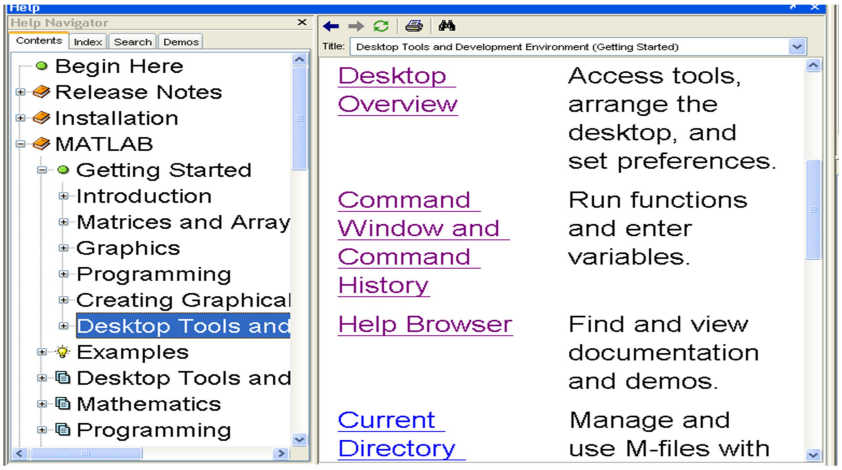


Si vous voulez faire des commentaires qui aident à expliquer placez-vous un symbole «%» devant le texte

```
>> s=2+3    % je fais une somme
s =
 5
```

- **Obtenir de l'aide**

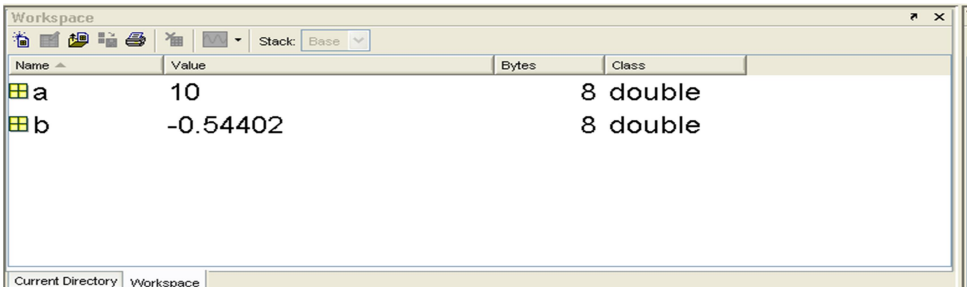
L'aide est intégrée à Matlab accessible à partir du menu d'aide



- **help** -> donne de l'aide sur une fonction ou un toolkit (help help) ;
- **helpdesk** -> documentation en hypertexte (requiert Netscape ou autre) ;
- **helpwin** -> aide en ligne dans une fenêtre séparée ;
- **lookfor** -> recherche d'un mot clé (lent) ;
- **which** -> localise fonctions et fichiers ;
- **what** -> liste des fichiers Matlab dans le répertoire courant ;
- **exist** -> check si une fonction ou une variable existe dans le Workspace who ;
- **whos** -> liste des variables dans le Workspace .

• Workspace

Workspace est l'espace mémoire commun à la fois au Command Window et également aux fichiers scripts. Lorsque MATLAB démarre et exécute des tâches, il s'attribue des espaces de mémoires pour stocker les variables



I.2 Opérations mathématiques

Quelque opération mathématique

```
Arithmetic operators.
  plus        - Plus                                  +
  uplus       - Unary plus                            +
  minus       - Minus                                 -
  uminus      - Unary minus                           -
  mtimes      - Matrix multiply                       *
  times       - Array multiply                       .*
  mpower      - Matrix power                          ^
  power       - Array power                          .^
  mldivide    - Backslash or left matrix divide       \
  mrdivide    - Slash or right matrix divide          /
  ldivide     - Left array divide                    .\
  rdivide     - Right array divide                   ./
  kron        - Kronecker tensor product           kron

Special characters.
  colon       - Colon                                 :
  paren       - Parentheses and subscripting        ( )
  paren       - Brackets                            [ ]
  paren       - Braces and subscripting             { }
  punct       - Function handle creation             @
  punct       - Decimal point                        .
  punct       - Structure field access               .
  punct       - Parent directory                     ..
  punct       - Continuation                         ...
  punct       - Separator                            ,
  punct       - Semicolon                            ;
  punct       - Comment                              %
  punct       - Invoke operating system command      !
  punct       - Assignment                           =
  punct       - Quote                                '
  transpose   - Transpose                           .'
  ctranspose  - Complex conjugate transpose          '
  horzcat     - Horizontal concatenation            [,]
  vertcat     - Vertical concatenation              [;]
  subsasgn    - Subscripted assignment          ( ),{ },.
  subsref     - Subscripted reference           ( ),{ },.
  subsindex   - Subscript index
```

- Format réel :

```
>> x = 6.248
    x =
          6.2480
```

- Format complexe :

```
>> z = 4 + 5j
        z =
             4.0000 + 5.0000i
```

Les vecteurs :

- **La saisie manuelle d'un vecteur**

```
L = [1.2 5.6 5 10 -5]
```

- **L'addition des vecteurs**

```
>>x = [1, 5, 8];
>> y = [7, -5, 5];
>> z = x+y
        z =
              8     0    13
```

- **La soustraction des vecteurs**

```
>> d = x-y
    d =
          -6 10 3
```

- **La multiplication**

Élément par élément

```
>> p = x.*y
     p =
         7    -25   40
```

- **La division**

```
>> E = x.\y
      E =
          7.0000  -1.0000  0.6250
```

- **Opération avec un scalaire**

```
>> x+2
     ans =
           3   7   10

>> x-3
    ans =
         -2    2    5

>> x*4
        ans =
              4    20    32

>> x/4
     ans =
        0.2500   1.2500    2.0000
```

– **La puissance et la racine carrée d'un vecteur :**

```
>> x = [1 5 8];
>> x^2 = x.^2
     x^2 =
                1  25  64

>>sqrt (x) =
1.0000   2.2361    2.8284

>> x = [1 5 8];
>> y = [7 -5 5];
>> x.^y
 ans =
 1.0e+004 * 0.0001 0.0000 3.2768
```

- La somme des éléments d'un vecteur :

```
>> Vecteur = [1 4 5 -2 6 -9 8 2.3 6.5 1.4];
>> somme = sum(Vecteur)
          somme =
                    23.2000
```

- Le produit des éléments d'un vecteur :

```
>> s = [1 4 5];
>> p = prod(s)
                    p =
                              20
```

- La valeur moyenne d'un vecteur :

```
>> d = [1 4 5 6 82 5];
>> vmoy = mean(d)
                    vmoy =
17.1667
```

- La différence élémentaire d'un vecteur :

Algorithme :

Soit « x » est un vecteur de N éléments, sa différence élémentaire est la suivante :

$y(n) = x(n+1) – x(n)$ avec $n=1..N-1$.

```
>> d = [1 4 5 6 82 5];
>> diff(d)
          ans =
                    3 1 1 76 -77
```

- Consultation de quelques éléments d'un vecteur

```
>> R = [4 5 7 8 1 22 5]
 R =
 4 5 7 8 1 22 5
```

% consultation de l'élément d'indice 3 :

```
>> R(3)
                ans = 7
```

- Modification de quelques éléments d'un vecteur

% modification de l'élément d'indice 3 : >> R(3) = 100

```
R =
4  5  100  8   1  22  5
```

- On peut définir un vecteur en donnant la suite qui forme le vecteur

```
» x=1:0.5:3
x =
1.0000    1.5000    2.0000    2.5000    3.0000
```

- Ou en utilisant une fonction qui génère un vecteur à espacement linéaire :

```
 » x=linspace(1,10,6)
 x =
 1.0000   2.8000   4.6000    6.4000     8.2000     10.0000
```

-Ou exponentiel :

```
 » x=logspace(0,2,7)
 x =
1.0000        2.1544         4.6416      10.0000     21.5443     46.4159     100.0000
```

- Matrices

On définit une matrice A en donnant ses éléments :

```
» A=[0.3 2.7 4.1;3.1 -0.74 2.1;6.7 -4.3 -2.1]

A = 0.3000   2.7000    4.1000
    3.1000   -0.7400    2.1000
    6.7000   -4.3000   -2.1000
```

- Matrice unitaire :

```
» B=eye(4)
 B =
     1   0   0   0
     0   1   0   0
     0   0   1   0
     0   0   0   1
```

Les éléments d'un vecteur ou d'une matrice peuvent être adressés en utilisant les indices sous la forme suivante :

t(10) élément n° 10 du vecteur t
A(2,9) élément se trouvant à ligne 2, colonne 9 de la matrice A
B(:,7) la colonne 7 de la matrice B
C(3,:) la ligne 3 de la matrice C

Opérations matricielles

Les opérations matricielles exécutées par MATLAB sont illustrées dans le tableau suivant :
B = A' la matrice B est égale à la matrice A transposée
E = inv(A) la matrice E est égale à la matrice A inversée
D = A-B soustraction
Z = X*Y multiplication
X = A\B résout le système d'équations Ax = B
X = A/B équivaut à (B'/A')'

```
» A = ones(2)
  A = 1    1
      1    1

» B = zeros(2)
B = 0    0
      0    0

» C = [A   B]
  C = 1  1   0   0
      1  1   0   0

» D = [A ; B]
  D =
      1    1
      1    1
      0    0
      0    0
```

Autres opérateurs :

> supérieur à
<inférieur à
>= supérieur ou égal
<= inférieur ou égal

Dans le tableau ci-dessous, on dresse les noms de variables et les nombres spéciaux utilisés par MATLAB :

Nom de la variable	*Signification*	*Valeur*
eps	*précision relative des nombres réels (distance entre 1.0 et le nombre réel flottant le plus proche)*	$2.2204.10^{-16}$
pi	π	*3.14159……*
i et j	*unités imaginaires*	$\sqrt{-1}$
inf	*nombre infini (1/0=inf)*	∞
NAN	*ce n'est pas un nombre : 0/0=NAN*	
date	*date*	
nargin	*nombre d'arguments d'appel transmis à une fonction*	
flops	*compteur opérations en virgule flottante*	
nargout	*nombre d'arguments de retour demandés à une fonction*	

Fonctions Mathématiques dans MATLAB :

Fonctions trigonométriques	*Remarques*
sin(x)	
cos(x)	
tan(x)	
asin(x)	
acos(x)	
atan(x)	$-\pi/2 \le a\tan(x) \le \pi/2$

atan2(x,y)	$a\tan(x/y)$; $-\pi/2 \le a\tan 2(x,y) \le \pi/2$
sinh(x)	
cosh(x)	
tanh(x)	
asinh(x)	
acosh(x)	
atanh(x)	

Autres fonctions mathématiques (élémentaires)	*Remarques*
abs(x)	Valeur absolue de x
angle(x)	Valeur complexe de l'angle de phase : Si x=réel → angle=0 Si $x=\sqrt{-1}$ → angle=π/2
sqrt(x)	Racine carrée de x
real(x)	Partie réelle de la valeur complexe de x
imag(x)	Partie imaginaire de la valeur complexe de x
conj(x)	Complexe conjugué de x
round(x)	Arrondi entier de x
fix(x)	Arrondi par défaut du réel x
floor(x)	Arrondi au voisinage de $-\infty$ du réel x
ceil(x)	Arrondi au voisinage de $+\infty$ du réel x
sign(x)	=+1 six>0 ; =-1 si x<0
rem(x,y)	Le reste de la division : = x-y*fix(x/y)
exp(x)	Exponentielle (de base e)
log(x)	Log (de base e)
log10(x)	log (de base 10)

II. Simulation et Programmation à l'aide de Matlab (opérations simples)

- **Exemple 01 :**

Soit à calculer le volume suivant $V=4/3.\pi*R^3$: où R=4cm Pour calculer V, on exécute les commandes suivantes (Ici, pi=π):

```
>>R=4
R =
4
>>V=4/3*pi*R^3
 V =
268.0826
```

– Test **'if'** Ce test s'emploie, souvent, dans la plupart des programmes. Un test 'if' est toujours suivi par un 'end'.

- **Exemple 02 :**

```
>>V=268.0826
V =
268.0826
>>if V>150, surface=pi*R^2,end
surface =
50.2655
```

- L'opérateur 'égal' (==) dans 'if' Il est noté (ou symbolisé) par '=='.

- **Exemple 03 :**

```
>>R=4
 R =
4
>>if R==4, V=4/3*pi*R^3;end
```

- L'opérateur '**ou**' Il est noté (ou symbolisé) par '|'

- **Exemple :**

Si R=4 ou m=1, alors $V=4/3.\pi*R^3$

```
>>if R==4 | m==1, V=4/3*pi*R^3;end
```

- **Exemples :**

Si g>2 ou g<0 alors a=4

```
>>if g>2 |g<0 ,a=4 ,and
```

- Les opérateurs '**&**' et '|' peuvent être utilisés dans la même chaîne :

```
>>if ((a==2 | b==3)&(c<5), g=1, end
```

- Opérateurs '**for/end**' et '**while/end**'

- **Exemples :**

```
>>for R=1 :5, V=4/3*pi*R^3; disp([R,V]), end
```

Dans ce cas, R varie de 1 à 5, et la commande "disp([R,V])" retourne la matrice suivante : [R=1 :5,V (V(1) :V(5)]

```
>>while R<5,  R=R+1 ; V=4/3*pi*R^3; disp([R,V]), end
```

while exécute l'instruction qui suit tant que le test logique est vrai.

– Exemple de pas dans la boucle **for** :

```
>>for R=5 :-1 :1, V=4/3*pi*R^3; disp([R,V]), end
```

Ici, le pas utilisé est dégressif (=-1). On peut utiliser les imbrications de 'for' autant de fois que l'on souhaite.

- **Exemple :**

```
>> for i=0 :10, for j=1 :5, V=4/3*pi*R^3;disp([R,V]);end,end
```

format :
>>format %% affiche les résultats avec 4 chiffres après la virgule.
>>format long %% affiche les résultats avec 16 chiffres après la virgule.

* break :
Interrope l'exécution d'un 'for' ou d'un 'while'
* goto :
Dans MATLAB l'instruction 'goto' est remplacée par 'break'.

Exemple :

```
while R==1
 .
 .
if x>Xlimite, break, end
.
.
End
```

* clear :
% Efface toutes les variables existantes en mémoire
* clc :
% Efface l'écran (fenêtre) de MATLAB

```
if     : commandes à exécuter si expression1 est"vrai"
elseif : commandes à exécuter si expression2 est"vrai"
else   : commandes à exécuter si aucune expression est "vrai"
end
```

Exécution conditionnelle

Les instructions

if… … end ,
if…. .. else……end ,
if…. … elseif…. .. else …. .. end ,
switch……case…. .. case…….. .. end

Les exécutions conditionnelles permettent de choisir entre plusieurs options.

La commande suivante permet d'affiner l'aide sur les fonctions mathématiques élémentaires :
help elfun

sin ⟶ % Sine.
sinh ⟶ % Hyperbolic sine.
asin ⟶ %Inverse sine.
asinh ⟶ % Inverse hyperbolic sine.
cos ⟶ % Cosine.
cosh ⟶ % Hyperbolic cosine.
acos ⟶ % Inverse cosine.
acosh ⟶ % Inverse hyperbolic cosine.
tan ⟶ % Tangent.
tanh ⟶ % Hyperbolic tangent.
atan ⟶ % Inverse tangent.
atan2 ⟶ % Four quadrant inverse tangent.
atanh ⟶ % Inverse hyperbolic tangent.
sec ⟶ % Secant.
sech ⟶ % Hyperbolic secant.
asec ⟶ % Inverse secant.
Asech ⟶ % Inverse hyperbolic secant.
csc ⟶ % Cosecant.
acsc ⟶ % Inverse cosecant.
acsch ⟶ % Inverse hyperbolic cosecant.
cot ⟶ % Cotangent.
coth ⟶ % Hyperbolic cotangent.
acot ⟶ % Inverse cotangent.
acoth ⟶ % Inverse hyperbolic cotangent.
exp ⟶ % Exponential.
log ⟶ % Natural logarithm.
log10 ⟶ % Common logarithm.
sqrt ⟶ % Square root.
abs ⟶ % Absolute value.
angle ⟶ % Phase angle.
conj ⟶ % Complex conjugate.

imag → % Complex imaginary part.
real → % Complex real part.
fix → % Round towards zero.
floor → % Round towards minus infinity.
ceil → % Round towards plus infinity.
round → % Round towards nearest integer.
rem → % Remainder after division.
sign → % Signum function.

Les Polynômes :

MATLAB représente les polynômes sous forme de vecteurs lignes dont les composantes sont ordonnées par ordre des puissances décroissantes. Un polynôme de degré « n » est présenté par un vecteur de taille « n+1 ».

- **Exemple :**

$f(x) = 3x^5 + 2x^3 - 1x^1 + 7$

```
>> f = [3 0 2  0 -1 7]
f =
3 0 2 0 -1 7
```

- **Les racines d'un polynôme (roots)**

- **xemples :**

```
>> roots(f)
ans =

  -1.1208 + 0.0000i
  -0.2926 + 1.2653i
  -0.2926 - 1.2653i
   0.8530 + 0.7119i
   0.8530 - 0.7119i
```

-Reconstruction d'un polynôme par ses racines (poly)

- **Exemples :**

```
>> poly([2 4 7])
ans =
     1   -13    50   -56 %Le polynôme f(x)= x³-13x²+50x-56
```

- Le produit polynomial (conv)

Pour faire le produit des fonctions polynomiales; on trouve la fonction **conv** :

- **Exemples :**

```
>> v1 = [2  3  7]  % f1(x) = 2x^2+3x+7
v1 =
2  3  7
>> v2 = [1  5  9]    % f2(x) = 1x^2+5x+9
v2 =
1   5    9
>> conv(v1,v2)
ans =

    2    13    40    62    63   % F(x) = 2x^4+13x^3+40x^2+62x+63
```

- **La division polynomiale (deconv)**

Pour faire la division des fonctions polynomiales; on trouve la fonction **deconv** :

- **Exemples :**

```
>> [Q, R] = deconv(v1,v2)
    Q =
          2
     R =
    0    -7   -11
```

- **L'évaluation d'un polynôme (polyval)**

Pour déterminer la valeur d'une fonction polynomiale dans un point ou intervalle ; on trouve la fonction suivante :

- **Exemples :**

```
>> x0 = 4;
>> f = [2  1 -2];    % f(x) = 2x^2+x-2
>> polyval(f, x0)  % f(x=4)
ans =
34
```

- **La dérivée d'une fonction polynomiale (polyder)**

polyder(f) : calcul de la dérivée d'une fonction polynomiale f(x).

- **Exemples :**

```
>> f = [8 9 -1]
f =
8  9  -1
>> polyder(f)
ans =
   16     9
                % f'(x) = 16x+9
```

Les nombres complexes (complex):

MATLAB accepte les nombres complexes sous leur forme algébrique « a+i*b » ou exponentielle R . exp (j . Theta). Les symboles « i » et « j » représentes le nombre imaginaire pur vérifiant : $i^2 = j^2 = -1$.

- **Exemples :**

```
>> z = 5 + 6i
z =
5.0000 + 6.0000i
>> z = complex(5 , 6)
z =
5.0000 + 6.0000i
```

- Partie réelle d'un nombre complexe (real)

```
>> real(z)
ans = 5
```

- **Partie imaginaire d'un nombre complexe (imag)**

```
>> imag(z)
ans =
6
```

- **Module et Phase d'un nombre complexe (abs ;angle)**

```
>> abs(z) % le module
ans = 7.8102
>> angle(z) % la phase
ans =    0.8761
```

- **Conjugué d'un nombre complexe (conj)**

- **Exemples :**

```
>> z = 5 + 6i
z =
5.0000 + 6.0000i
>> conj(z)
ans =
5.0000 - 6.0000i
```

III . Représentation Graphique sous Matlab

Matlab possède des possibilités d'édition de graphes (2D et 3D) et d'images très développées. Voir **demo** → visualization.

Le graphisme 2D :

plot(y) : affiche le graphe du vecteur (ou des colonnes de matrice) y en fonction de son indice.
plot(x,y) : affiche y en fonction de x.
plot(x,y,'+k') : ajoute l'option de couleur et de ligne et/ou de marque. ici 'k' corresponde a la couleur 'black' et '+' à la marque +.
plot(x1,y1,'-g',x2,y2,':rx',x3,y3,'b+') : affiche y1 en fonction de x1, y2 en fonction de x2, y3 en fonction de x3, etc… la première est affichées en couleur verte (green) avec une ligne continue, la seconde en rouge (red) avec la marque x et une ligne discontinue superposée et enfin la troisième en bleu avec une marque +. Taper **helpplot** pour explorer toutes les possibilités de cette commande.

- **Format du graphique**

On peut choisir le format du graphique :

```
plot(x,y)          % tracer y(x) avec échelles linéaires
semilogx(f,A)      % tracer A(f) avec échelle log(f)
semilogy(w,B)      % tracer B(w) avec échelle log(B)
polar(theta,r)     %  tracer r(theta) en coordonnées polaires
bar(x,y)           % tracer y(x) sous forme de barres
```

- Ajout de texte au graphique

```
title('Titre du graphique')          % donner un titre au graphique
xlabel('Temps')                      % étiquette de l'axe x
ylabel('Amplitude')                  % étiquette de l'axe x
gtext('Valeur absolue')              % ajouter du texte au graphique avec la souris
legend('sortie réelle','sortie simulée')  % ajouter une légende
```

- Manipulation de graphiques et de fenêtres

```
grid, grid on, grid off      % ajouter une grille
axis ([-2 8 -20 30])         % choix des échelles x = (-2; 8) et y = (-20; 30)
 hold, hold on, hold off     % garder le graphique sur l'écran (pour tracer plusieurs courbes sur
                                                                le même graphique)
figure                       % ouvre une nouvelle fenêtre graphique
close all                    %ferme toutes les fenêtres graphiques
```

- Les couleurs

y %jaune ; m %magenta ; c %cyan ; r %rouge ; g %vert ; b %bleu ;
w %blanc ; k %noir.

- Tracés discontinus

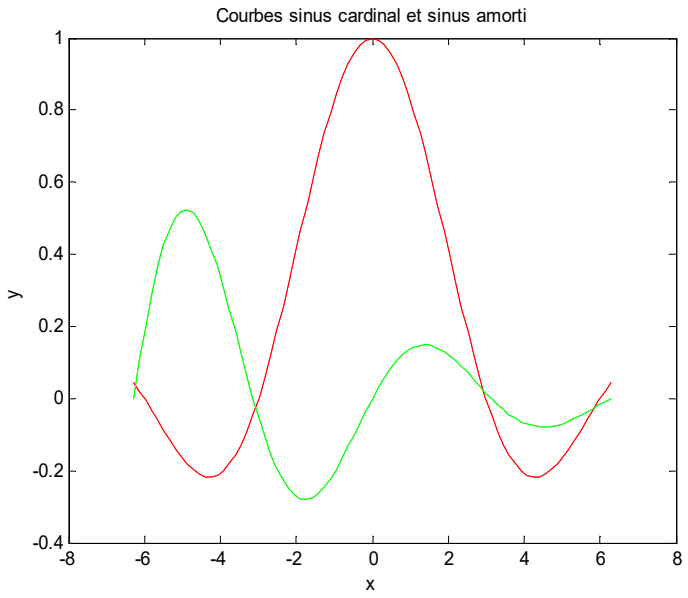
. %point ; o %cercles ; x %croix ; + %plus ; * %étoiles .

- Tracés continus

- %trait continu ; : %pointillés ; -. %trait-point ; -- %trait-trait.

- **Exemple :**

```
» x= -2*pi : pi/20 : 2*pi ;
» y1= sinc (x/3);
» plot(x , y1 , 'r :' )
» hold on
» y2=sin(x).*exp(-0.2*x)/5;
» plot(x , y2 , 'g');
» xlabel ('x')
» ylabel('y')
» title('Courbes sinus cardinal et sinus amorti')
```

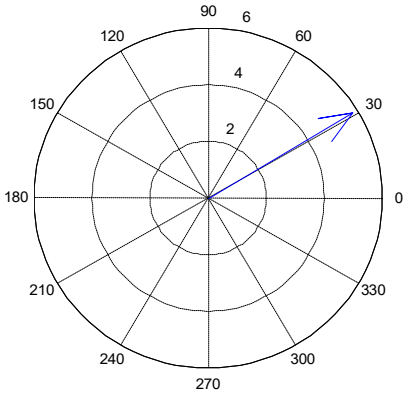


Représentation graphique d'un nombre complexe :

- Représentation en module et argument (compass)

- **Exemples :**

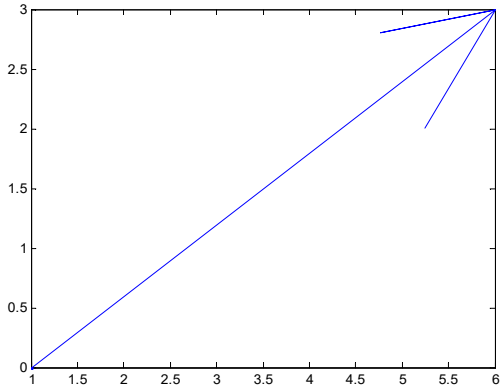
```
>> z = 5 + 3i
z =
5.0000 + 3.0000i
>> compass(z)
```



- Représentation en partie réelle et imaginaire (feather)

- **Exemples :**

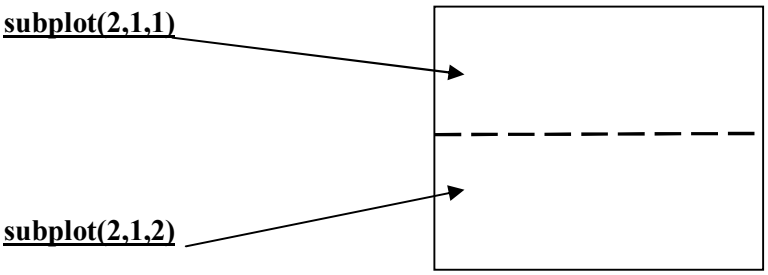
```
>> z = 4 + 7i
z =
4.0000 + 7.0000i
>> feather(z)
```



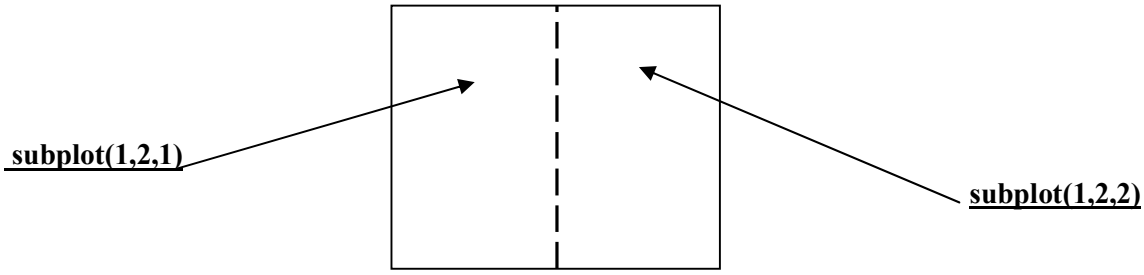
- **Graphique multiple (subplot)**

On peut tracer plusieurs graphiques dans la même fenêtre en utilisant l'instruction **subplot** pour diviser la fenêtre en plusieurs parties (le tracés se feront ensuite avec **plot**, **semilogx**, etc.) :

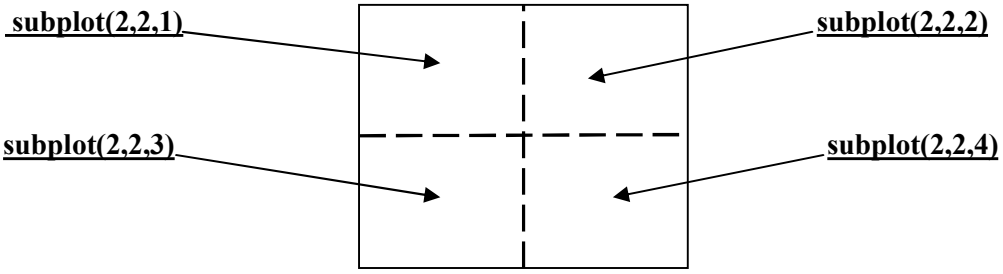
- diviser la fenêtre en deux parties (2 x 1)



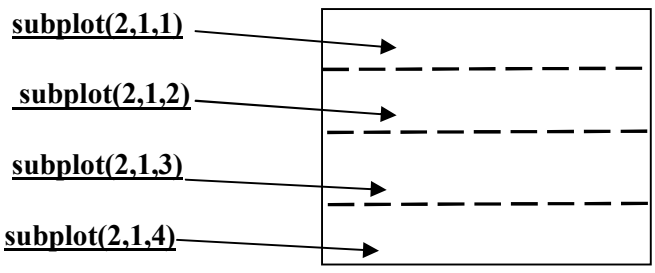
- diviser la fenêtre en deux parties (1 x 2)



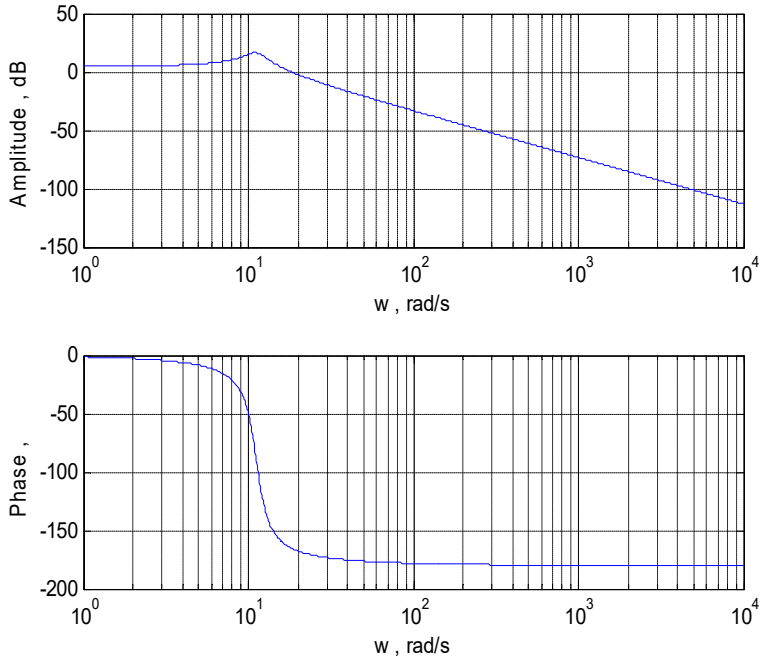
- diviser la fenêtre en quatre parties (2 x 2)



- diviser la fenêtre en quatre parties (4 x 1)



```
» w = logspace(0,3,1000);
» s =  j*w;
» H = 225./(s.*s+3*s+225);
» AdB = 20*log10(abs(H));
» phase = angle(H)*(180/pi);
» subplot(2,1,1),semilogx(w,AdB),grid
» xlabel('w , rad/s'),ylabel('Amplitude , dB')
» subplot(2,1,2),semilogx(w,phase),grid
» xlabel('w , rad/s'),ylabel('Phase , ')
```



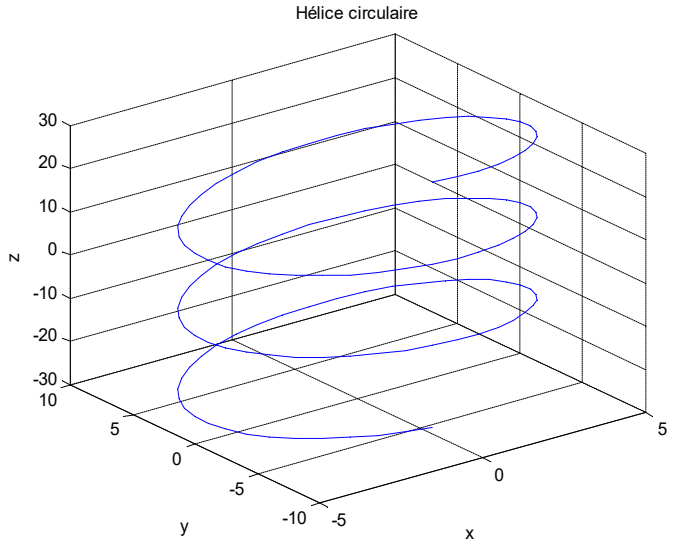
- Représentation graphique 3D

Le tracé de courbes dans l'espace se fera à l'aide de l'instruction plot3 qui obéit à une syntaxe analogue à celle de plot. plot3(x,y,z,'symb')

○ Exemple :

%tracé d'une hélice circulaire définie par une équation paramétrique.

```
t=-3*pi :pi/20 :3*pi;
x=5*sin(t);
y=6*cos(t);
z=3*t;
plot3(x,y,z)
title('Hélice circulaire')
grid
xlabel('x')
ylabel('y')
zlabel('z')
```



- **Grillage en perspective (mesh*)***

  - **Exemple :**

```
>> clf
>> x = linspace(-3, 3, 30);
>> y = linspace(-3, 3, 30);
>> [X Y] = meshgrid(x,y);
>> Z = peaks(X, Y) % peaks est une fonction à 2-D prédéfinie dans Matlab
>> mesh(X, Y, Z)    % grillage 3D
>> meshc(X, Y, Z)   % grillage 3D avec les contours sur le plan de base
```