Université Mohammed Premier
École Nationale des Sciences Appliquées d'Oujda

Cours d' *Informatique 2 : MATLAB*

Version 3.0 (Janvier 2023)

MATLAB POUR L'INGÉNIEUR

STPI-1

ENSAO, 2022 – 2023

Partie 1

Prof. *Kamal GHOUMID*

Université Mohammed Premier
École Nationale des Sciences Appliquées d'Oujda

Cours d' *Informatique 2 : MATLAB*

Version 3.0 (Janvier 2023)

MATLAB POUR L'INGÉNIEUR

Chapitre 1

Introduction

Partie 1

Prof. **Kamal GHOUMID**

Présentation du logiciel (1/12)

- La transformation digitale est une démarche incontournable dans notre vie.
- L'utilisation de logiciel de calcul est devenue absolument indispensable depuis les années 70 dans le domaine scientifique.
- MATLAB pour MATtrix 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.

- 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** : concerve une trace de toutes les opérations effectuées. Traçabilite 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".

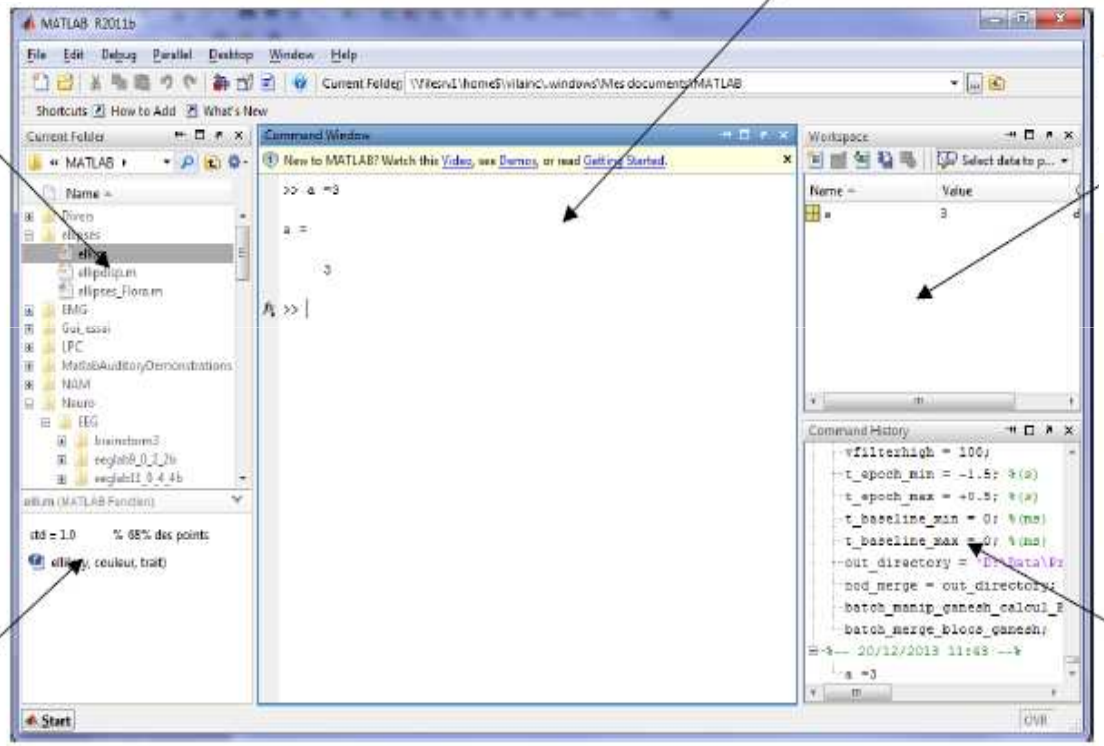
- **Help** : Avoir de l'aide sur une instruction, une commande, …

* **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

Modes de fonctionnement :

* **mode interactif**: MATLAB exécute les instructions au fur et à mesure qu'elles sont données par l'usager.

* **mode exécutif**: MATLAB exécute ligne par ligne un fichier ".m" (programme en langage MATLAB).

Richesse de Matlab :

L'utilisation de boites 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, …

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'aides 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 = 23, 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 = 'ENSAO';
>> 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
```

Quelques commandes usuelles de Matlab :

* Pour avoir une information sur une variable x : **whos 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 (.m et .mat) 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 : **who**.

* 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 l'aide 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 (1/3)

Organisation du cours :

* 7 séances de cours de 2 heures chacune;
* 7 séances de travaux pratiques (TP) de 2 heures chacune;
* 5 séries de TP;
* 2 devoirs;
* 1 séance magistrale;

Chapitres concernés :

* Partie I :
  - Ch. 1 : Introduction.
  - Ch. 2 : Matrices et Vecteurs.
  - Ch. 3 : Représentations Graphiques.
  - Ch. 4 : Manipulations des Polynômes.

* Partie II :
  - Ch. 5 : Programmation Matlab (**Partie majeure et importante du cours**).
  - Ch. 6 : Les Macros

* Partie III :
  - Ch. 7 : Calcul formel avec Matlab.

Information sur le cours (2/3)

Modalités d'évaluations :

* Examen finale : 75%.
* Examen TP : 25%.

Durant les séances des TPs :

* L'étudiant doit créer dans le bureau de l'ordinateur son propre dossier :

Bureau \ Clique droit \ Nouveau \ Dossier \ Nom de l'étudiant

* Pour chaque exercice, l'étudiant doit créer un nouveau script: (File\New\script), qui doit être sauvegardé sous le nom "Exn°TPn°STPI1.m" dans son dossier.

Le cours suppose une familiarité avec des notions de bases d'algèbre, d'analyse, de statistiques et quelques problèmes liés à la physique...

Recommandations :

*** Pensez à sauvegarder régulièrement** vos données et vos paramètres (fichiers m-file) pour se protéger contre les défaillances et les pannes du système et/ou de l'ordinateur.

* Veuillez SVP à ne pas ouvrir plusieurs sessions Matlab sur le même ordinateur.

Téléchargement et installation de Matlab :

* Version d'évaluation (courte durée)

* Site : 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;

MATLAB POUR L'INGÉNIEUR

STPI-1

ENSAO, 2022 – 2023

Partie 1

Prof. **Kamal GHOUMID**

MATLAB POUR L'INGÉNIEUR

Chapitre 2

Vecteurs et Matrices

Partie 1

Prof. **Kamal GHOUMID**

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.

- 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 cherche 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 lui assigner une valeur, puis on lui 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

```
>> a = 7 ;% la variable "a" est assignée à 7
```

Invite Matlab

Variable a

Valeur assignée à la variable a

Supprimer la sortie de la commande

Écrire des commentaires d'explications ou d'aides

- 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* x *n*)** : ***m*** lignes et ***n*** colonnes (***m***, ***n*** > 1).
  - **Vecteur (*1* x *n*)** : ***1*** ligne et ***n*** colonnes (vecteur ligne).
  - **Vecteur (*n* x *1*)** : ***n*** lignes et ***1*** colonne (vecteur colonne).
  - **Scalaire (*1* x *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
Undefined function or variable ‘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

>> e*d/c
ans =
    42.6187

>> floor(e*d/c) %Parie entière
ans =
    42
```

```
>> f = 48;

>> gcd(a,f) %plus grands diviseurs communs
ans =
     2

>> lcm(a,f) %plus petits communs multiples
ans =
    240

>> lcm(a,b)
ans =
    310

>> round(e) %pour arrondir
ans =
    51

>> round(d^2 - a)
ans =
    24
```

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
```

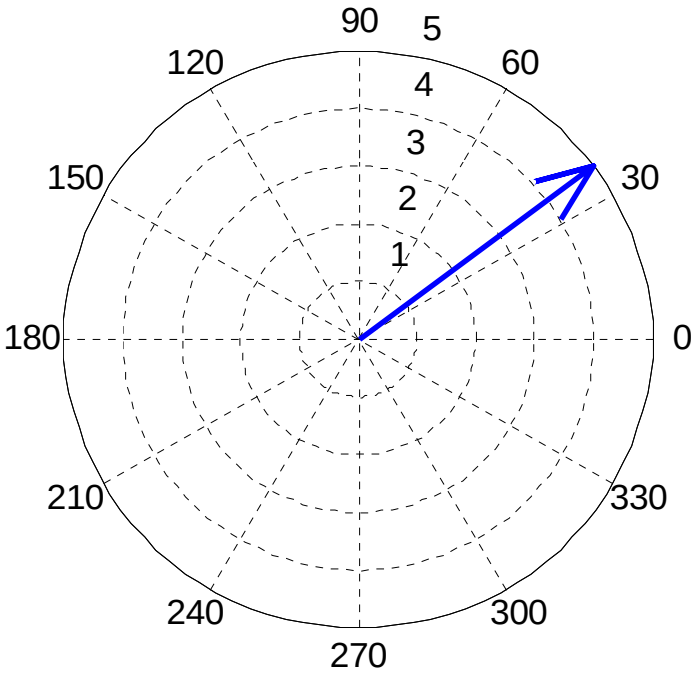
```
>> log(a*5*b-33*c)
ans =
   7.484930283289661

>> 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)

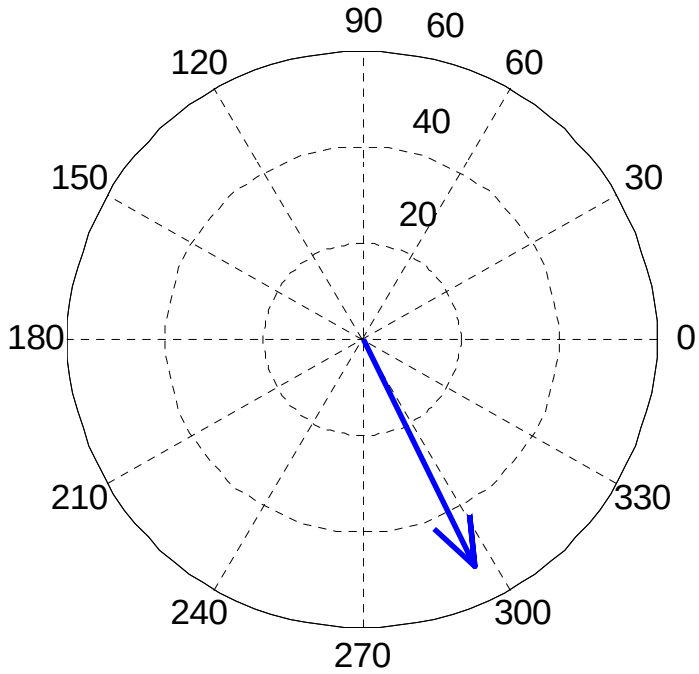
```
>> Z1 = 3 - 5i
Z1 =
   3.0000 - 5.0000i

>> Z2 = a + b * i
Z2 =
  10.0000 +31.0000i

>> Z1 + 5.4 * Z2
ans =
   5.7000e+01 + 1.6240e+02i

>> real(Z1 + 5.4 * Z2)
ans =
    57

>> imag(Z1 + 5.4 * Z2)
ans =
   162.4000

>> angle(Z1)
ans =
   -1.0304

>> conj(Z2)
ans =
   10.0000 -31.0000i

>> abs(Z1)
ans =
    5.8310
```

Manipulations des variables : Calculs simples (9/14)

```
>> Z1 = 4 + 3i

Z1 =

   4.0000 + 3.0000i

>> compass(Z1)
```




```
>> Z2 = 23 - 47i

Z2 =

   23.0000 - 47.0000i

>> compass(Z2)
```



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([84   73   84   73   32   84   79   84
79])
ans =
TITI TOTO

>> char(37)
ans =
%

>> datestr(now)
ans =
27-Mar-2022  10:09:30

>> clock
ans =
  1.0e+003 *
   2.0220   0.0030   0.0270   0.0100   0.0090   0.0572
```

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
```

```
>> 3 ~= 7
ans =
     1

>> 3 ~= 3
ans =
     0

>> 1 & 0
ans =
     0

>> 0 & 0
ans =
     0

>> 1 & 1
ans =
     1

>> 0 | 0
ans =
     0

>> 1 | 0
ans =
     1
```

Quelques fonction 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 <= 0

sin(x) : sinus de x

cos(x) : cosinus de x

tan(x) : tangente de x

Quelques 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

Quelques fonction qui agissent sur les nombres complexes, 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
          2.333333333333333    0.000001234500000

>> format rational
          >> pi
               ans =
                    355/113
```

```
>> format short g
x =
     2.3333  1.2345e-006

>> format short eng
x =
    2.3333e+000     1.2345e-006

>> get(0,'format') %Format courant
ans =
    short eng
```

Écritures et caractéristiques

- 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 ; 54  2  -5];     %Matrice : dimension 2x3
>>
>> whos
```

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

Pas besoin d'initialiser le type ou les dimensions.

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

Point-virgule pour la ligne suivante dans la matrice

Crochets pour définir les matrices

```
>> U = [6,-13,sqrt(31),3,-12]
U =
    6.0000  -13.0000    5.5678    3.0000  -12.0000
```

```
>> V = [2;43;-5]
V =
     2
    43
    -5
```

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
```

Accéder aux éléments d'une matrice.

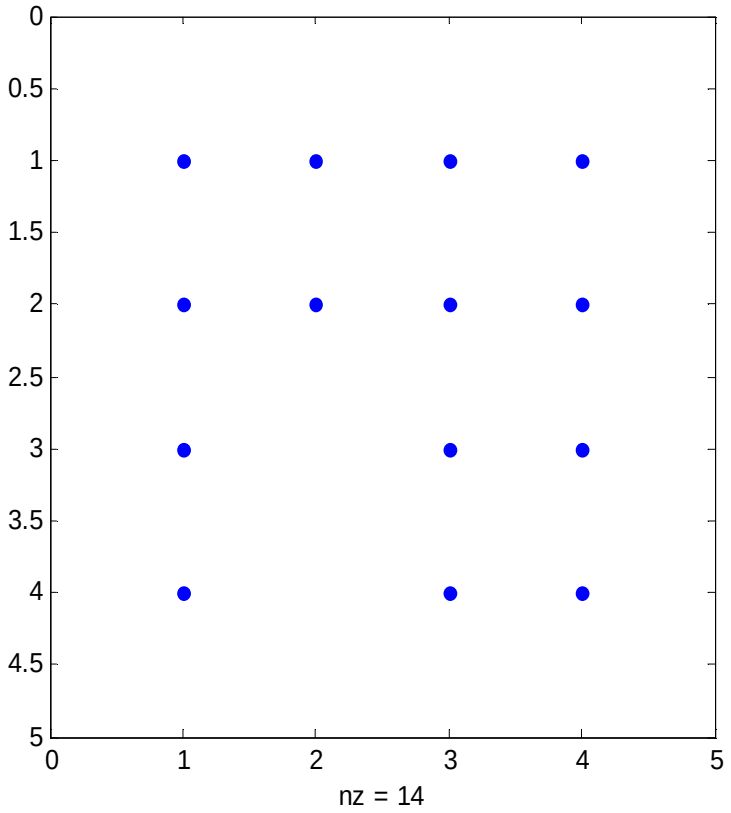
```
>> B = [1 2 7 -3 ; 2 -6 8 -4 ; 9 0 8 1 ; 2 0 7 5]
B =
     1     2     7    -3
     2    -6     8    -4
     9     0     8     1
     2     0     7     5

>> 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
     5
```

Accéder aux éléments d'une matrice.

```
>> B

B =

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

```
>> C = sparse(B)
C =
   (1,1)        1
   (2,1)        2
   (3,1)        9
   (4,1)        2
   (1,2)        2
   (2,2)       -6
   (1,3)        7
   (2,3)        8
   (3,3)        8
   (4,3)        7
   (1,4)       -3
   (2,4)       -4
   (3,4)        1
   (4,4)        5
```

```
>> spy(C) %Les éléments qui existent, mais pas leurs
          % valeurs, aussi pas d'éléments nuls
```



Accéder aux éléments d'une matrice.

```
>> whos
```

Name	Size	Bytes	Class	Attributes
B	4x4	128	double	
C	4x4	188	double	sparse

```
>> full(C)
ans =
    1     2     7    -3
    2    -6     8    -4
    9     0     8     1
    2     0     7     5
```

Calculs et opérations.

```
>> 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
```

Calculs et opérations.

```
>> 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

>> D = [1 2 3 ; 2 4 1 ; -1 -2 2]
D =
     1     2     3
     2     4     1
    -1    -2     2

>> E = D(:,[3,1,2])
E =
     3     1     2
     1     2     4
     2    -1    -2

>> F = D([3,1,2],:)
F =
    -1    -2     2
     1     2     3
     2     4     1

>> E = D([3,1,2],:) - repmat(A,1,3)
E =
    -2    -3     1
    -1     0     1
    -1     1    -2
```

Calculs et opérations.

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

>> size(2)
ans =
    2     2

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

```
>> B = [1   2   3
 4   5   6]

 B = 1  2  3
     4  5  6

>> size(2)
ans =
    2     2

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

```
>> magic(3) %Matrice magique: même  somme par tous
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

>> rand(1,7) %7 nombres aléatoires compris entre 0 et 1
ans =
    0.8147    0.9058    0.1270    0.9134    0.6324    0.0975    0.2785

>> sort(rand(1,7)) %Classement croissant
ans =
0.1576   0.4854   0.5469   0.9572   0.9575   0.9649   0.9706
```

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, uniformoment)
v =
   0    0.5236    1.0472    1.5708    2.0944    2.6180    3.1416
```

```
>> w = logspace(1,2,7) %Valeur initiale=0, Valeur finale=pi, Nombre de valeurs (7 en log)
w =
   10.0000    14.6780    21.5443    31.6228    46.4159    68.1292   100.0000
```

Opérations sur les vecteurs et les matrices

```
>> 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
```

Opérations sur les vecteurs et les matrices

```
>> 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
```

Opérations sur les vecteurs et les matrices

```
>> 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 matricies.

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.

inv(M) : inverse d'une matrice M.

size(M) : nombre de lignes et de colonnes de la matrice M.

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.

Opérations sur des matrices particulières

```
>> 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

>> diag(M+2)
     7    -4
```

Opérations sur des matrices particulières

```
>> 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
```

Opérations sur les matrices

```
>> 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
```

```
>> I = triu(M)
I =
    2   -5    7   -11
    0   -4    0    23
    0    0    9    14

>> S = tril(M)
S =
    2    0    0    0
    6   -4    0    0
    7   -8    9    0
```

Opérations sur les matrices

```
>>N = [eye(3)+4*ones(3)+floor(25*rand(3))]

N =
   27    25    11
    5    25    24
    8     8    13

>> N(:)
ans =
   27
    5
    8
   25
   25
    8
   11
   24
   13

>> N(:)'
ans =
   27    5    8   25   25    8   11   24   13

>> reshape(N,5,2)
??? Error using ==> reshape
To RESHAPE the number of elements must
not change.
```

```
>> M = [1:6;-3:4:9 floor(10*rand(1,2));3*ones(1,3) 12,-6,13]

M =
    1    2    3    4    5    6
   -3    1    5    9    5    4
    3    3    3   12   -6   13

>> reshape(M,1,18) %remodeler la matrice M
ans =
1  -3   3   2   1   3   3   5   3   4   9  12   5   5  -6   6   4  13

>> reshape(M,2,9)
ans =
    1    3    1    3    3    9    5   -6    4
   -3    2    3    5    4   12    5    6   13

>> reshape(M,6,3)
ans =
    1    3    5
   -3    5    5
    3    3   -6
    2    4    6
    1    9    4
    3   12   13
```

Résolution d'un système d'équations linéaire (1/5)

Soit le système d'équations suivant :

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

Ce système peut être écrit sous la forme :

A * X = B

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

La résolution de ce système d'équations est :

- Première méthode : ***X = inv(A) * B***

- Deuxième méthode : ***X = A \ B***

Résolution d'un système d'équations linéaire (2/5)

```
>> 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'équations linéaire (3/5)

Soit le système d'équations suivant :

$$\begin{cases} 2x - 4y + 2z = 8 \\ 3x - y - 2z = 32 \\ 5x + 3y + 4z = -12 \end{cases}$$

Ce système d'équations peut se mettre sous la forme : **A * X = B**

$$A = \begin{pmatrix} 2 & -4 & 2 \\ 3 & -1 & -2 \\ 5 & 3 & 4 \end{pmatrix} \qquad X = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \qquad B = \begin{pmatrix} 8 \\ 32 \\ -12 \end{pmatrix}$$

On peut vérifier déjà que :

```
>> Exig1 = rank(A)
Exig1 =
     3
```

```
>> Exig2 = det(A)
Exig2 =
   120
```

```
>> Exig3 = inv(A)
Exig3 =
    0.0167    0.1833    0.0833
   -0.1833   -0.0167    0.0833
    0.1167   -0.2167    0.0833
```

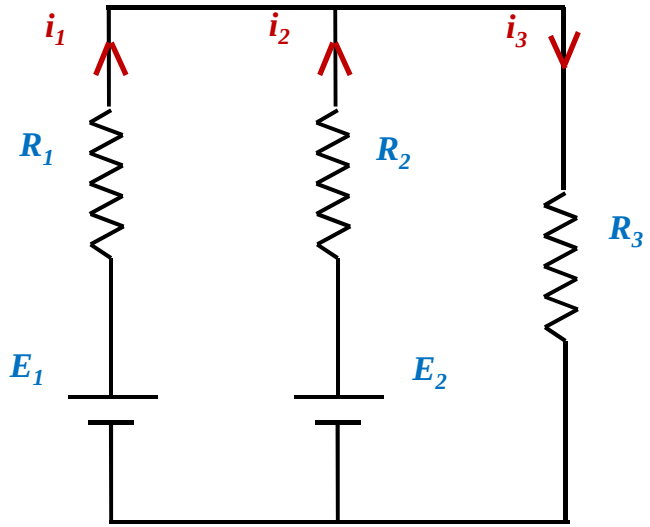
La résolution de ce système est :

- <u>Première méthode</u> : ***X = inv(A) * B***
- <u>Deuxième méthode</u> : ***X = A \ B***

$$X = \begin{pmatrix} 5 \\ -3 \\ -7 \end{pmatrix}$$

Résolution d'un système d'équations linéaire (4/5)

Soit le circuit électrique suivant :



Le système d'équations relatif à ce circuit électrique 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'équations linéaire (5/5)

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 i1 dans le circuit électrique (électrocinétique)
```

Diagonalisation d'une matrice : (1/5)
Valeurs propres, Vecteurs propres

$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}_{\{\vec{i},\vec{j},\vec{k}\}} \xrightarrow{\text{Diagonalisation}} \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix}_{\{\vec{u},\vec{v},\vec{w}\}}$$

Diagonalisation ➔ il faut trouver les valeurs propres λ_1, λ_2, λ_3 et les vecteurs propres u, v, w.

$$P = \begin{pmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \\ z_1 & z_2 & z_3 \end{pmatrix} \quad (u \downarrow, v \downarrow, w \downarrow)$$

$$D = \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{pmatrix}$$

```
>> 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:

$$\det(A - \lambda.I) = 0;$$ où I 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 λ_1 et λ_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 λ_1 et à λ_2 respectivement:

$$\vec{u} = \begin{pmatrix} 0{,}8944 \\ 0{,}4472 \end{pmatrix} \qquad \vec{v} = \begin{pmatrix} 0{,}7071 \\ 0{,}7071 \end{pmatrix}$$

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

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

Diagonalisation d'une matrice : (3/5)
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

>> rank(A)
ans =
    2

>> rank(P)
ans =
    2
```

```
>> A^12
ans =
     1058786      -1054690
      527345      -523249

>> P*D^12*inv(P)
ans =
     1058786      -1054690
      527345      -523249

>> P*D^12*inv(P) - A^12
ans =
     0     0
     0     0
```

Diagonalisation d'une matrice : (4/5)
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
```

Diagonalisation d'une matrice : (5/5)
Valeurs propres, Vecteurs propres

$$\begin{pmatrix} 3 & 2 & 4 \\ -1 & 3 & -1 \\ -2 & -1 & -3 \end{pmatrix}_{\{\vec{i},\vec{j},\vec{k}\}} \xrightarrow{\text{Diagonalisation}} \begin{pmatrix} -1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix}_{\{\vec{u},\vec{v},\vec{w}\}}$$

Repère des vecteurs propres

```
>> 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
```

%Le polynôme caractéristique de M :

P(λ) = λ ^3 – λ^2 + 4

```
>> [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
```

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
```

Université Mohammed Premier
École Nationale des Sciences Appliquées d'Oujda

Cours d' *Informatique 2 : MATLAB*

Version 3.0 (Janvier 2023)

MATLAB POUR L'INGÉNIEUR

STPI-1

ENSAO, 2022 – 2023

Partie 1

Prof. **Kamal GHOUMID**

Université Mohammed Premier
École Nationale des Sciences Appliquées d'Oujda

Cours d' ***Informatique 2 : MATLAB***

Version 3.0 (Janvier 2023)

MATLAB POUR L'INGÉNIEUR

Chapitre 3

Représentations Graphiques

Partie 1

Prof. **Kamal GHOUMID**

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, …

- La commande **plot** permet de tracer un ensemble de points de coordonnées (x_i , y_i), $i=1, ..., N$.

- 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 dépond 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) ;

Quelques spécifications et/ou attributs pour la commande plot

'y' : jaune	**'.'** : point
'm' : magenta	**'o'** : cercle
'c' : cyan	**'x'** : marque x
'r' : rouge	**'+'** : plus
'g' : vert	**'*'** : étoile
'b' : bleu	**'s'** : carré
'w' : blanc	**'d'** : losange
'k' : noir	**'p'** : pentagone
'^' : triangle (haut)	**'h'** : hexagone
'v' : triangle (bas)	**'-'** : trait plein
'<' : triangle (gauche)	**':'** : pointillé court
'>' : triangle (droit)	**'-'** : pointillé long
'-.' : pointillé mixte (tiret point)	

- 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 currant 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.

- 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 reconnaitre 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.

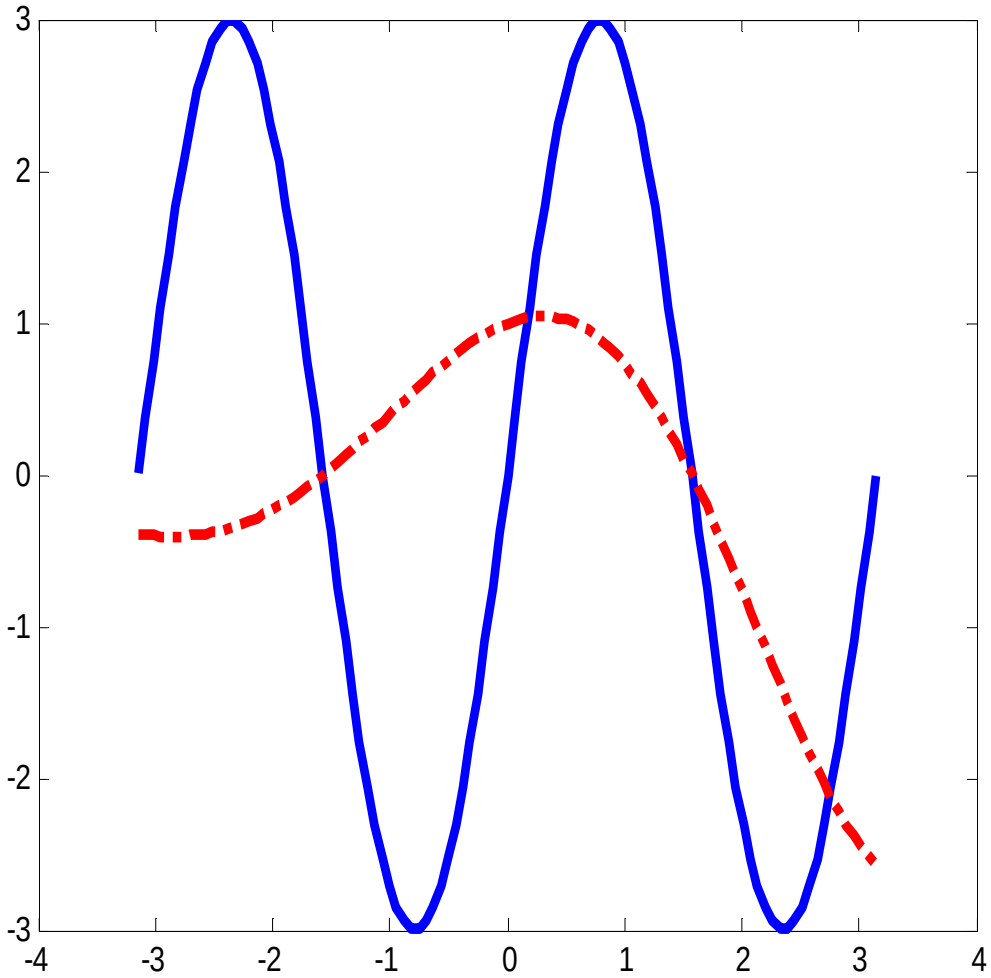
- La commande **title** permet de donner un titre à la figure.

- 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)

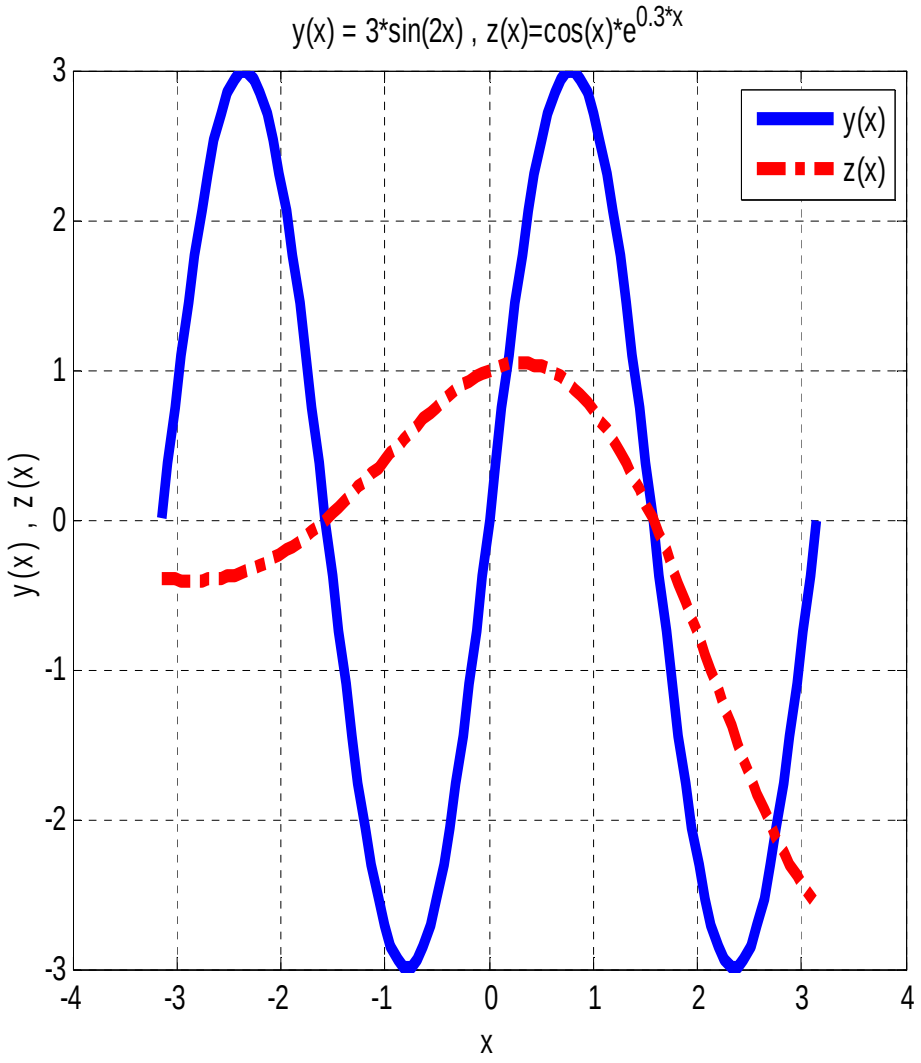
Exemple

```
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)
```



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
```

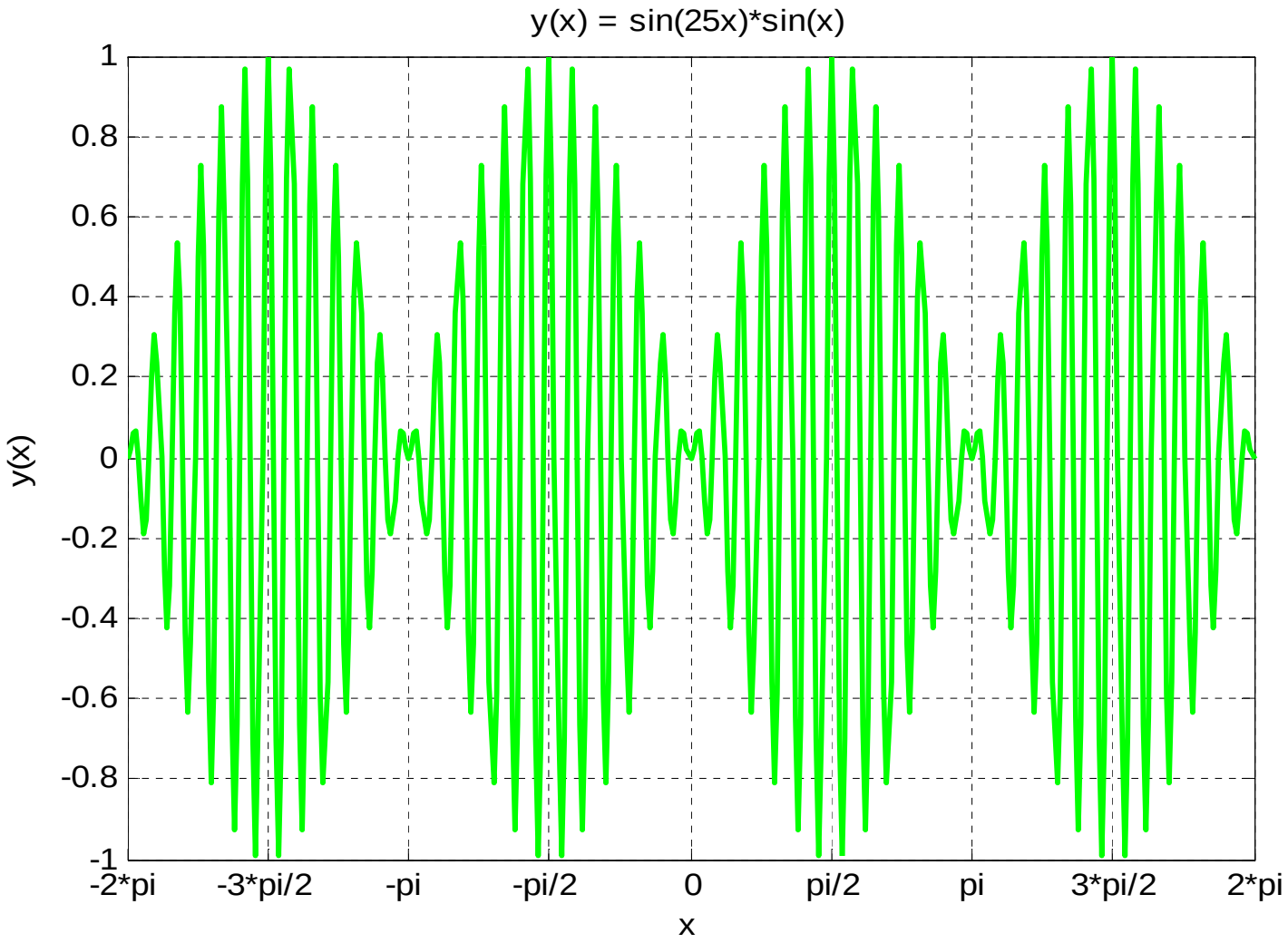


- L'instruction **axis** est utilisée pour préciser les plages de la représentation graphique. Sa syntaxe est **axis([x_{min} x_{max} y_{min} y_{max}])**.

- Il est parfois souhaitable d'utiliser l'instruction **set** pour changer l'écriture des abscisses.

```
x = -2*pi : pi/100 : 2*pi;
y = sin(25*x).*sin(x);
plot(x,y,'g','linewidth',2);
axis([-2*pi   2*pi   -1  1])
set(gca,'XTick',-2*pi:pi/2:2*pi);
set(gca,'XTickLabel',{'-2*pi','-3*pi/2','-pi','-pi/2','0','pi/2','pi','3*pi/2','2*pi'});
grid on
xlabel('x');
ylabel('y(x)');
title('y(x) = sin(25x)*sin(x)');
```

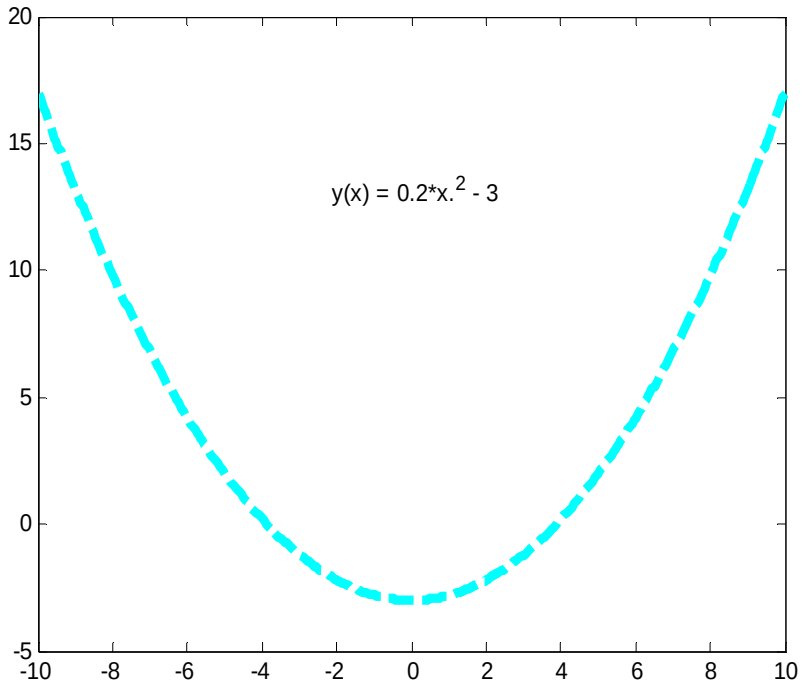
Exemple : Exécution du code

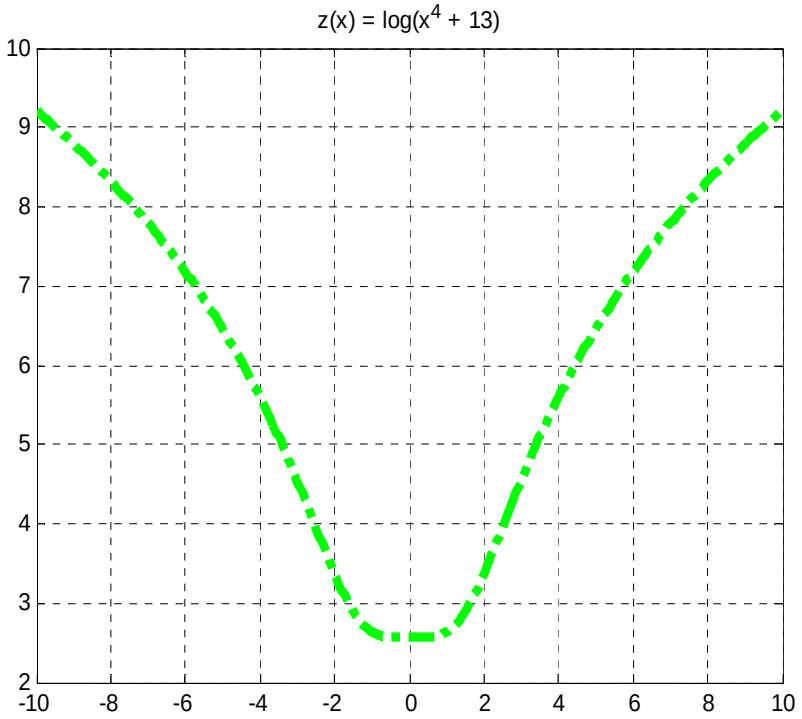


Représentation 1D (9/21)

Exemple

```
x = -10 : 0.05 : 10;
y = 0.2*x.^2 - 3;
z = log(x.^4 + 13);
figure (1);
plot(x,y,'--c','linewidth',4);
gtext('y(x) = 0.2*x.^2 - 3');
figure (2);
plot(x,z,'-.y','linewidth',4);
grid on
title('z(x) = log(x^4 + 13)');
```





Représentation 1D (10/21)

Exemple

```
x = -2 : 0.05 : 2;
y = sin(3*x);
z = sin(3*x + pi/4);
plot(x,y,'r',x,z,'k');
grid on
legend('y(x)','z(x)');
```

➡ L'instruction **subplot** permet de décomposer une fenêtre en sous-fenêtres et d'afficher sur chacune de ces sous-fenêtres une figure différente. Cette commande à la syntaxe suivante :

subplot*(m,n,i)*

où

m : est le nombre de lignes (encarts verticaux);

n : est le nombre de colonnes (encarts horizontaux);

i : pour spécifier dans quelle sous-fenêtre doit s'effectuer l'affichage. L'ordre de numérotation des sous-figures est de gauche à droite et de haut en bas.

subplot(2,1,1)
subplot(2,1,1)

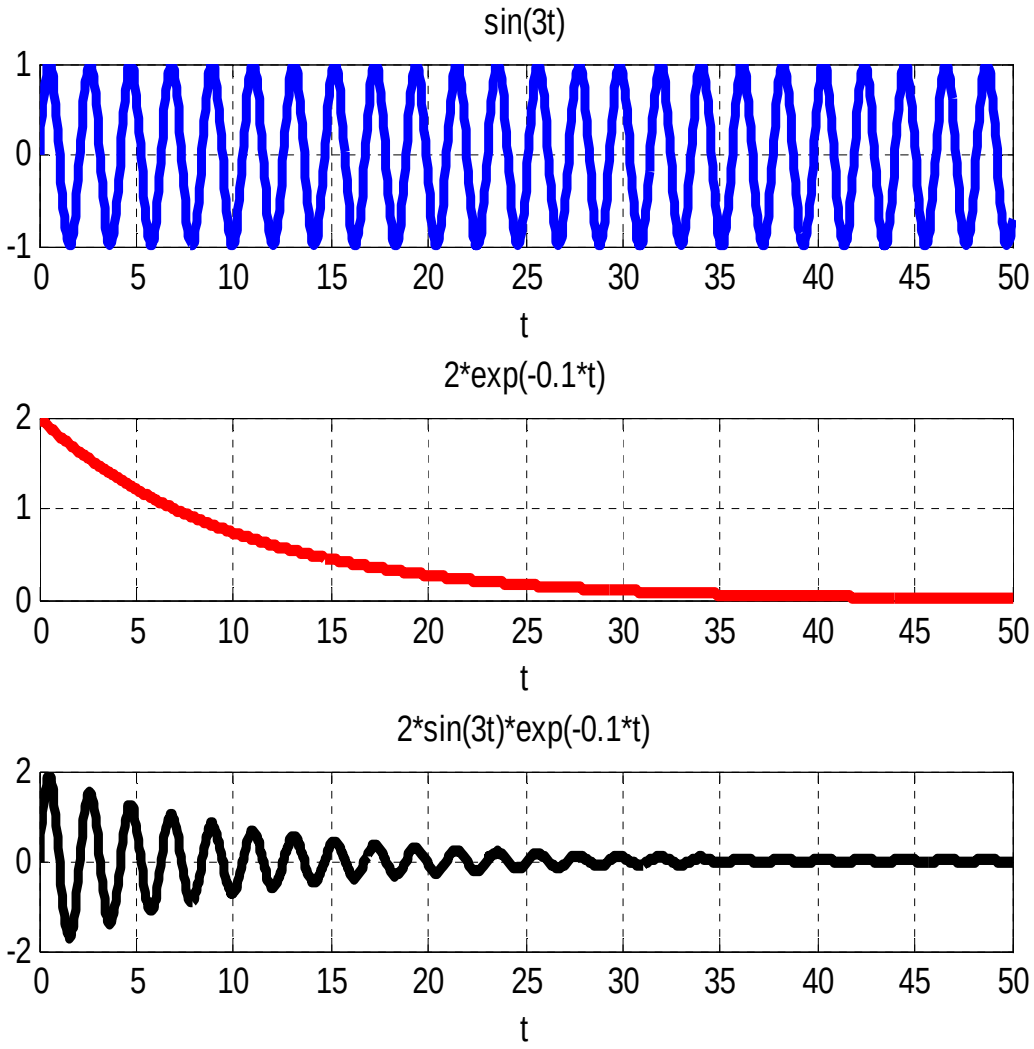
subplot(3,2,1)	**subplot(3,2,2)**
subplot(3,2,3)	**subplot(3,2,4)**
subplot(3,2,5)	**subplot(3,2,1)**

subplot(1,2,1)	**subplot(1,2,2)**

Représentation 1D (12/21)

Exemple

```
t = 0 : 0.05 : 50;
x = sin(3*t);
y = 2*exp(-0.1*t);
subplot(3,1,1)
plot(t,x,'linewidth',3);
xlabel('t');
title('sin(3t)');
grid on
subplot(3,1,2)
plot(t,y,'r','linewidth',3);
xlabel('t');
title('2*exp(-0.1*t)');
grid on
subplot(3,1,3)
plot(t,x.*y,'k','linewidth',3);
xlabel('t');
title('2*sin(3t)*exp(-0.1*t)');
grid on
```



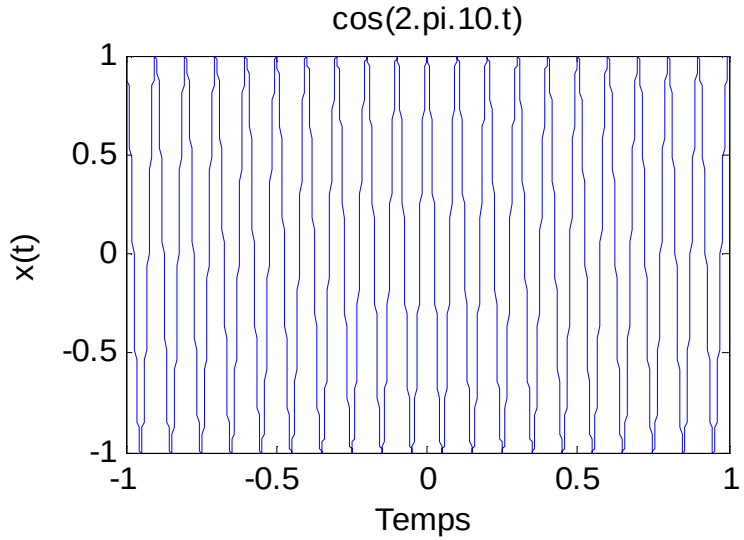
Représentation 1D (13/1)

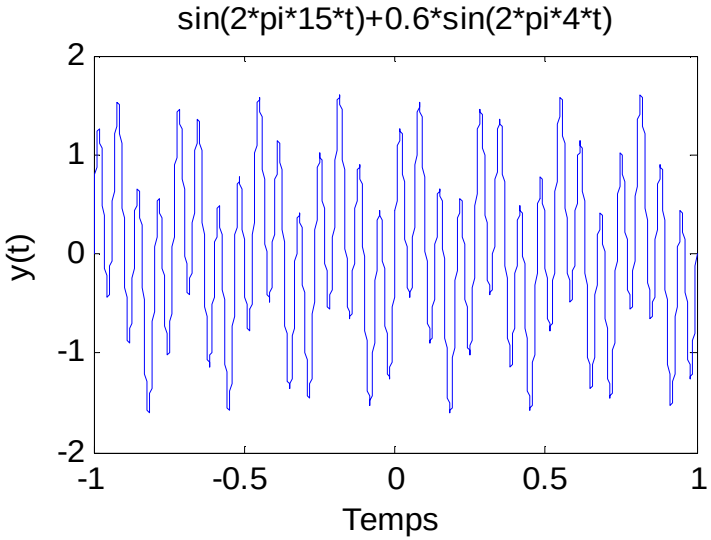
Exemple

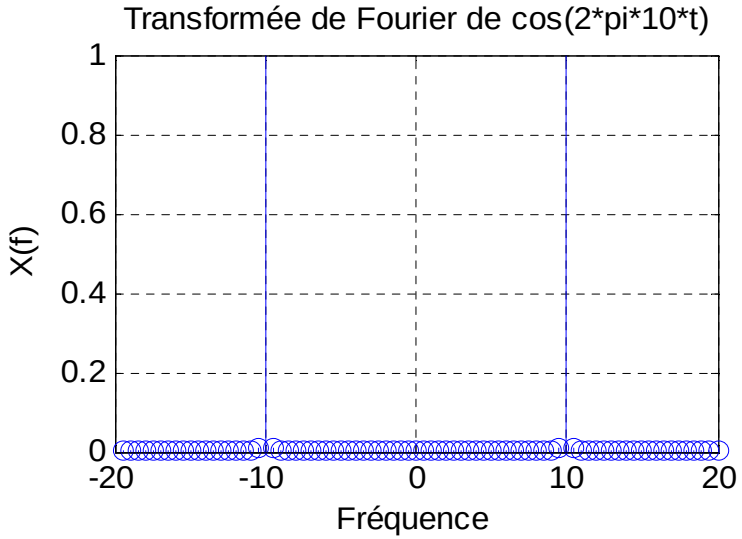
```
fe=1000;
te=1/fe;
t=-1:te:1;
x=cos(2*pi*10*t);
subplot(2,2,1);
plot(t,x);
xlabel('temps');
ylabel('x(t)');
title('cos(2.pi.f.t)');

% Transformée de Fourier
subplot(2,2,3);
f=linspace(-fe/2,fe/2,length(t));
Xf=fftshift(fft(x)/fe);
stem(f,abs(Xf));
axis([-20,20,0,1]);
grid on;
xlabel('fréquence');
ylabel('X(f)');
title('Transformée de Fourier de cos(2*pi*10*t)');
```

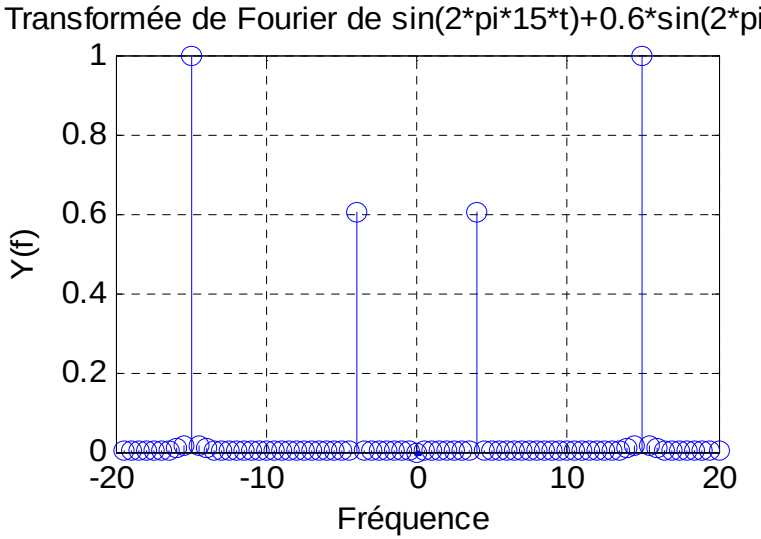
```
y=sin(2*pi*15*t)+0.6*sin(2*pi*4*t);
subplot(2,2,2);
plot(t,y);
xlabel('temps');
ylabel('x(t)');
title('sin(2*pi*15*t)+0.6*sin(2*pi*4*t)');


% Transformée de Fourier
subplot(2,2,4);
f=linspace(-fe/2,fe/2,length(t));
Yf=fftshift(fft(y)/fe);
stem(f,abs(Yf));
axis([-16,16,0,1]);
grid on;
xlabel('fréquence');
ylabel('X(f)');
title('Transformée de Fourier de sin(2*pi*15*t)+0.6*sin(2*pi*4*t)');
```

Représentation 1D (15/21)









➡ La commande **fplot** permet elle ausside tracer le graphe d'une fonction sur un intervalle donné. Sa syntaxe est :

fplot('nomf', [xmin , xmax])

où :

*** nomf** est soit le nom d'une fonction MATLAB incorporée, soit une expression définissant une fonction de la variable x, soit le nom d'une macro (voir plus loin).

*** [x_{min} , x_{max}]** est l'intervalle pour lequel est tracé le graphe de la fonction.

➡ Il est possible de tracer plusieurs fonctions sur la même figure. Il faut pour cela utiliser la commande **fplot** de la manière suivante:

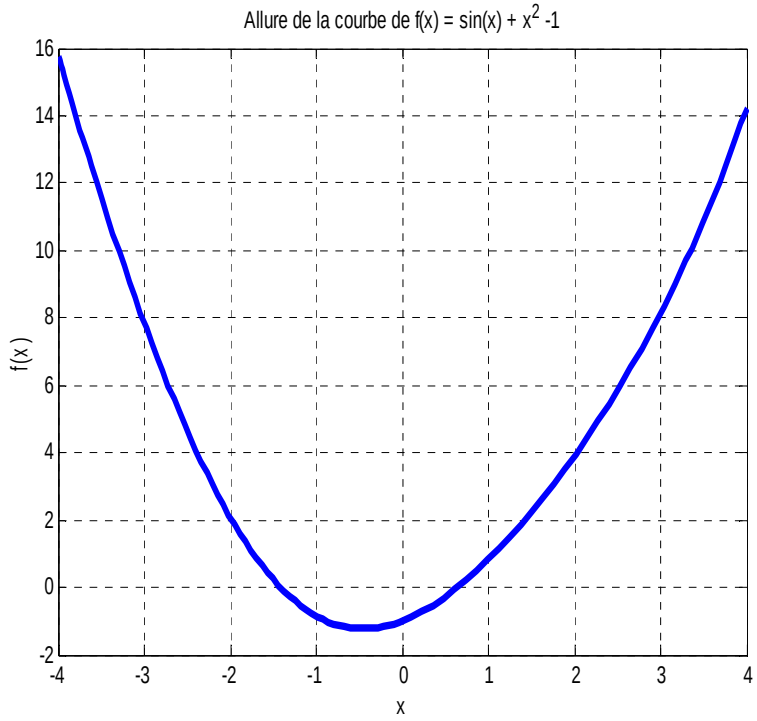
fplot('[nomf_1 , nomf_2 , nomf_3]', [xmin , xmax])

où **nomf_1**, **nomf_2**, **nomf_3** sont soit les noms des fonctions MATLAB incorporées, soit des expressions qui définissent une fonction de la variable x, soit des macros.

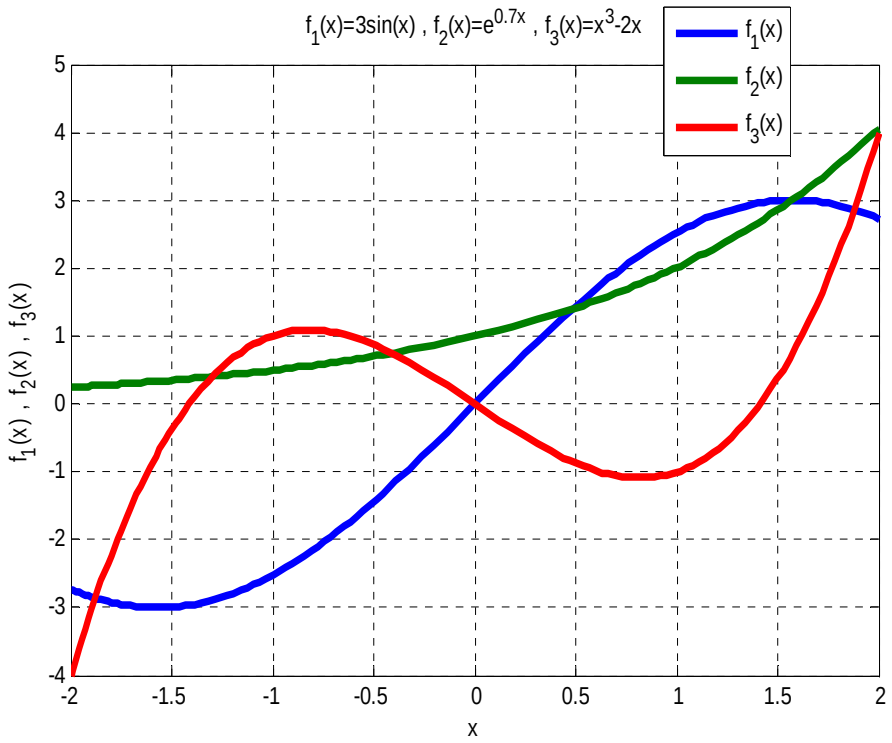
Représentation 1D (17/21)

Exemples

```
fplot('sin(x) + x^2-1',[-4  4])
xlabel('x');
ylabel('f(x)');
title ('Allure de la courbe de f(x) = sin(x) + x^2 -1');
grid on
```




```
fplot('[3*sin(x) , exp(0.7*x) , x^3-2*x]', [-2 , 2])
xlabel('x');
ylabel('f_1(x) , f_2(x) , f_3(x)');
title ('f_1(x)=3sin(x) , f_2(x)=e^{0.7x} , f_3(x)=x^3-2x');
grid on
legend('f_1(x)','f_2(x)','f_3(x)');
```



- Autres types de graphique plus adaptés aux représentations des données de types listes (statistiques) : **hist**, **pie**, **bar**, **stem**, **boxplot**, …

- La fonction **hist** permet de tracer un histogramme d'un vecteur ***x***, en répartissant ces valeurs (éléments du vecteur) en ***n*** classes. Sa syntaxe est **hist(x,n)**.

- La commande **[N,X] = hist(x,n)** retourne dans ***N*** l'effectif de chacune des classes et dans ***X*** l'abscisse du centre de chaque classe.

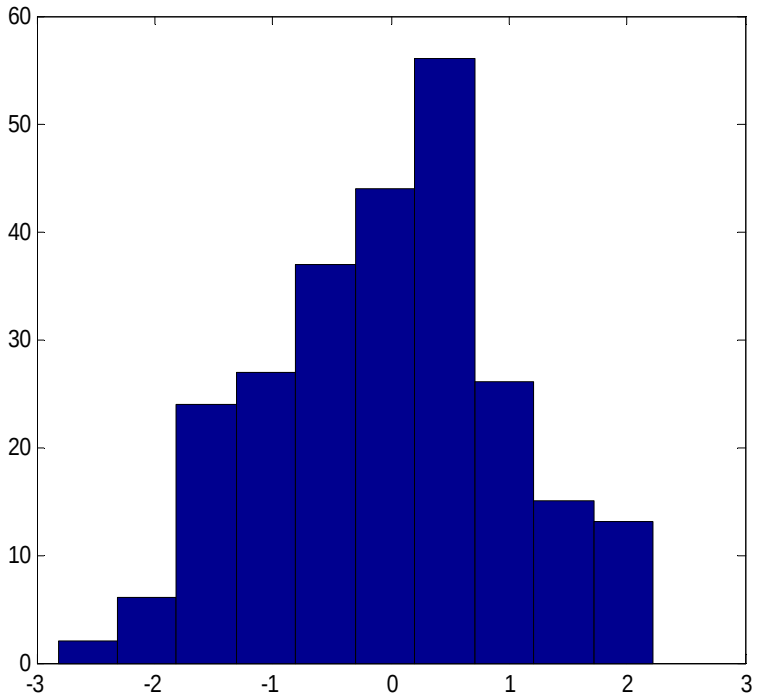
- La fonction **pie(x)** permet de dessiner un diagramme en cercle de valeurs de **x** normalisées.

- L'instruction **bar(x,y)** dessine un diagramme sous forme de barres des valeurs de ***y*** en fonction de celles de ***x***.

Représentation 1D (19/21)

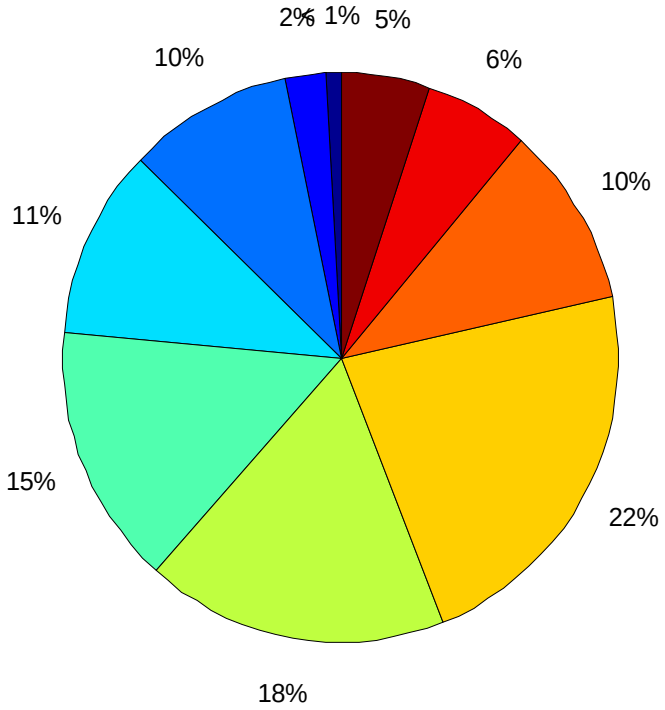
Exemples

```
A = randn(1,250);
figure (1)
hist(A);
[N,X] = hist(A);
figure (2)
pie(N)
```



Après exécussion, dans l'espace de commande :

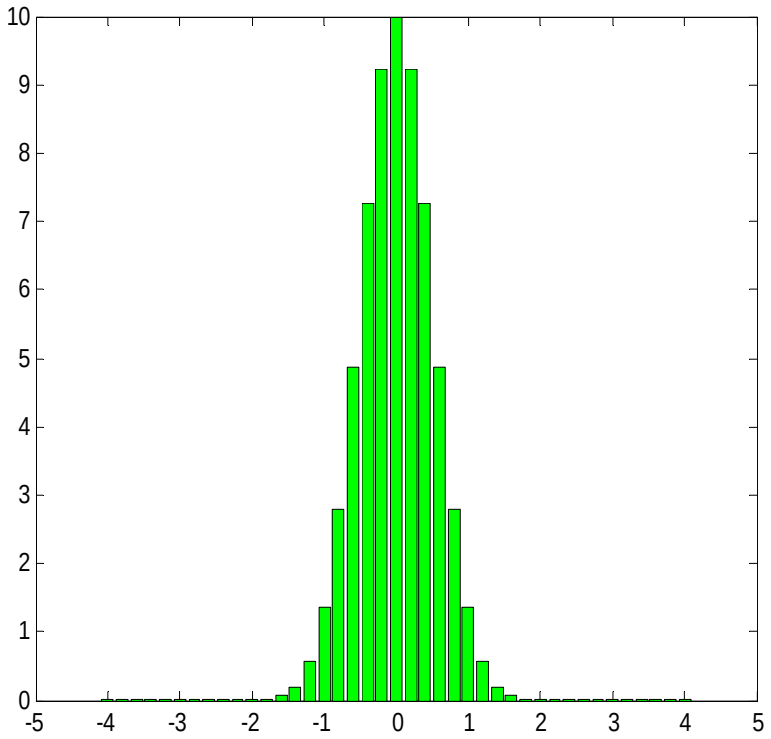
```
>> N =
     2   6   24   27   37   44   56   26   15   13

>> sum(N)
   ans =
        250
```



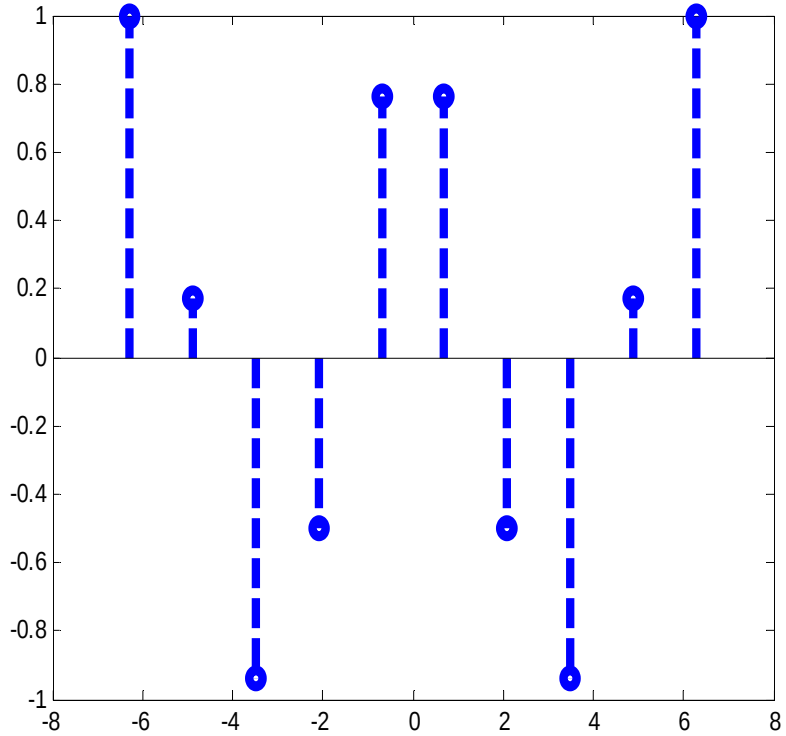
Représentation 1D (20/21)

Exemples

```
x = -4:0.2:4;
bar(x,10*exp(-2*x.^2),'g')
```

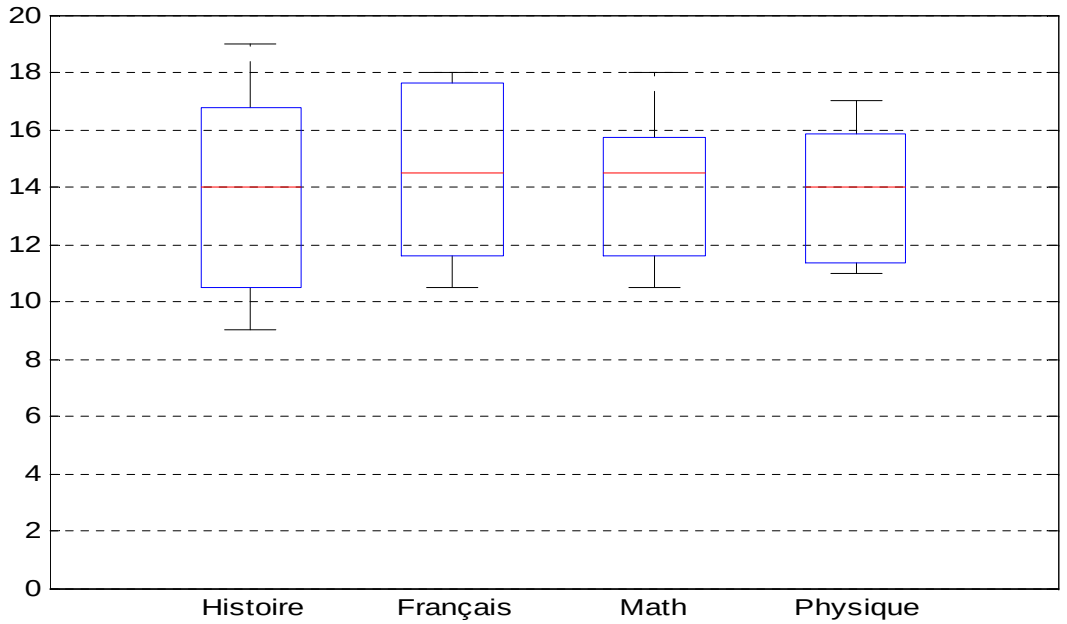



```
t = linspace(-2*pi,2*pi,10);
h = stem(t,cos(t),'--','linewidth',4);
```



Exemple

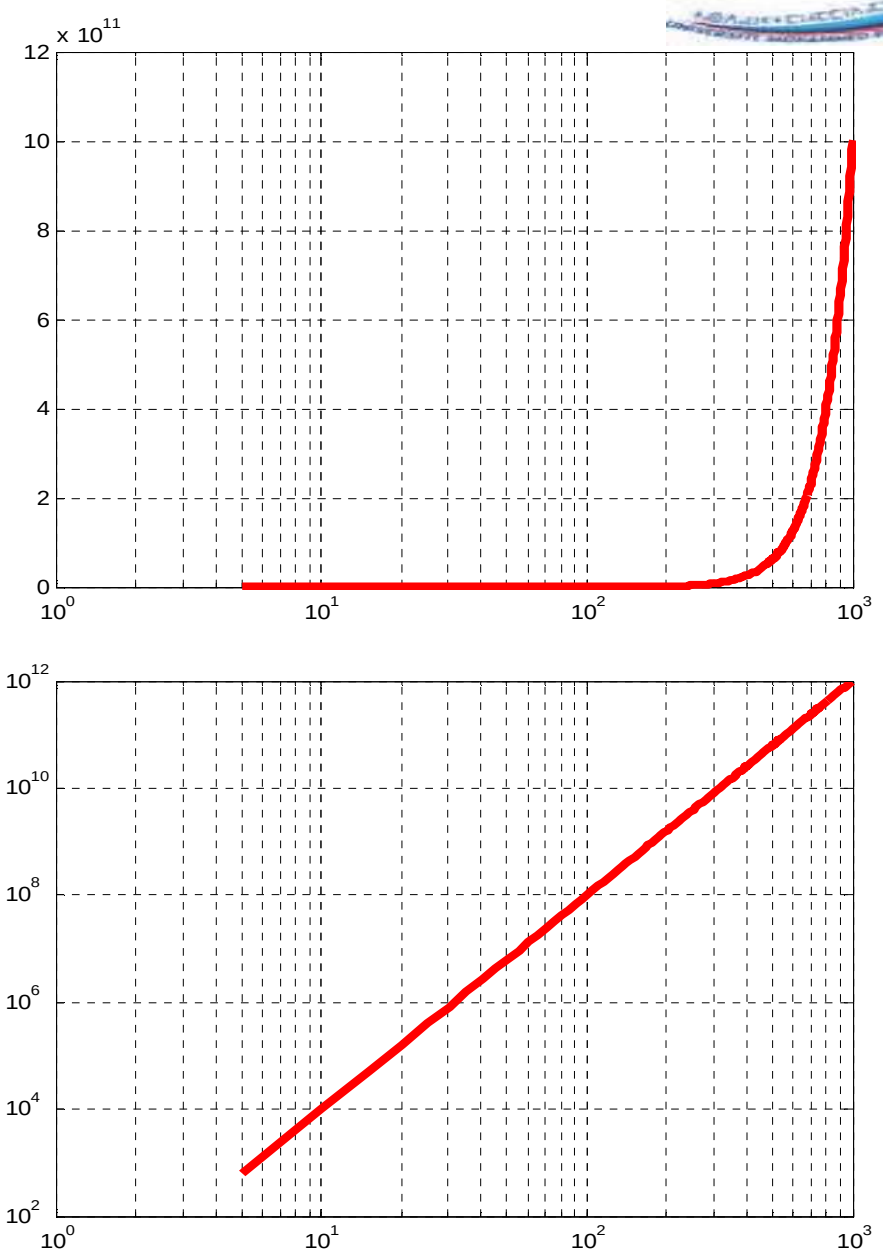
```
%Cinq étudiants ont eu les notes suivantes dans quatre matières différentes :
>> N = [16,18,14.5,15.5 ; 9,12,10.5,11 ; 19,17.5,18,17 ; 14,14.5,15,14 ; 11,10.5,12,11.5]
>> N =
        16.0000   18.0000   14.5000   15.5000
         9.0000   12.0000   10.5000   11.0000
        19.0000   17.5000   18.0000   17.0000
        14.0000   14.5000   15.0000   14.0000
        11.0000   10.5000   12.0000   11.5000
boxplot(N,'labels',{'Histoire','Français','Math','Physique'});
axis([0  5  0  20]);
grid on
```



Représentation 1D (21/21)

Exemple : Représentation en échelle log

```
>> x = linspace(0,1000,200);
>> y = x.^4 + 2*x;
>> figure(1);
>> semilogx(x,y,'r','LineWidth',4)
>> grid on
>> figure(2);
>> loglog(x,y,'r','LineWidth',4)
>> grid on
```



Représentation 1D (21/21)

Sauvegarde d'une figure

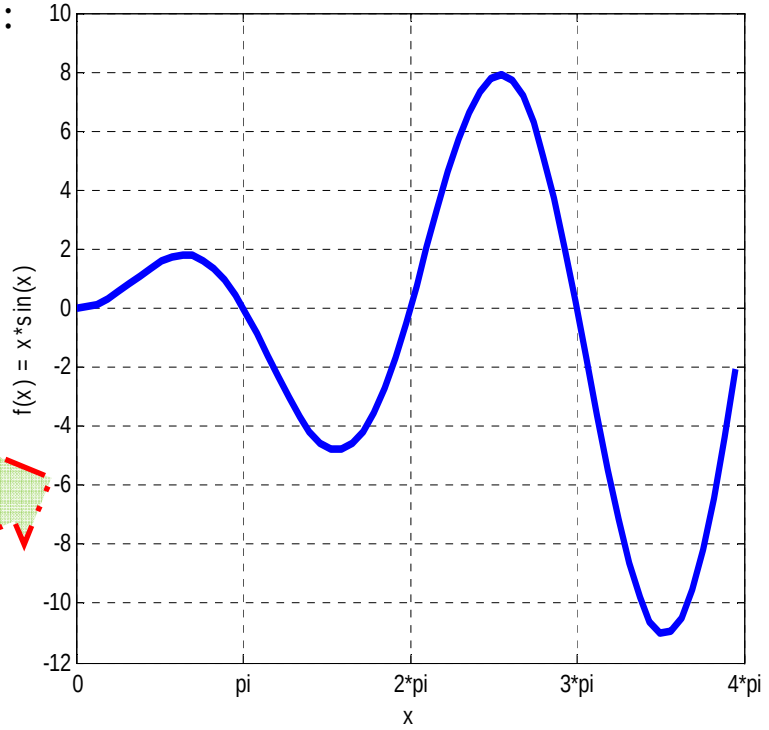
- L'instruction **saveas** (ou "**print**") permet de sauvegarder une figure d'une fenêtre graphique dans un fichier sous divers formats d'images (ps, psc, eps, epsc, jpeg, tiff, …).

- La syntaxe de la commande **saveas** est :

saveas(gcf , nom du fichier , type de format)

- Sauvegarde de la figure de l'exemple ci-après :
  >> saveas(gcf,'FigureSave.png')

```
x=0:pi/100:4*pi;
plot(x,x.*sin(x),'LineWidth',3)
set(gca,'XTick',0:pi:4*pi);
set(gca,'XTickLabel',{'0','pi','2*pi','3*pi','4*pi'});
axis([0 4*pi -12 10]);
grid on
xlabel('x');
ylabel('f(x) = x.*sin(x)');
ylabel('f(x) = x*sin(x)');
```

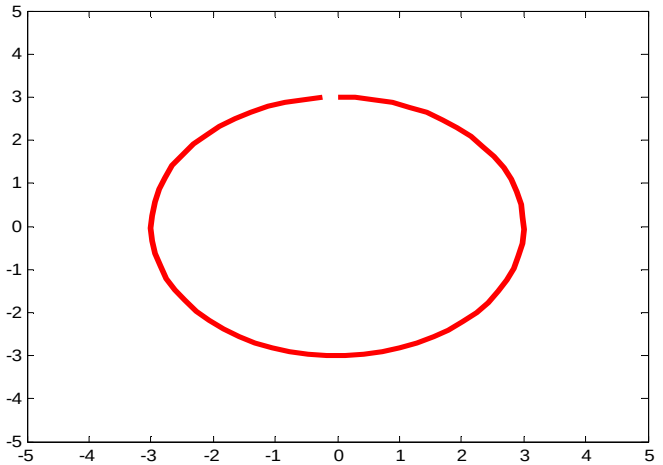


Représentation 2D & 3D (1/4)

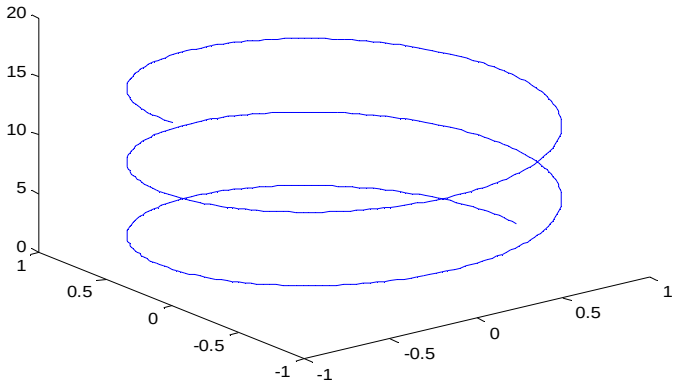
Cette partie ne sera pas trop détailler vu les chapitres qui restent à aborder (parties importantes du cours). On va se contenter de quelques exemples. Sachant que Matlab propose beaucoup de choses concernant les graphismes **2D, 3D et l'animation**.

Exemples

```
%Courbe paramétrique : le cercle
t = [0 :0.1: 2*pi];
rayon = 3;
X = rayon * sin(t);
Y= rayon * cos(t);
plot (X , Y , 'r','linewidth',3);
axis([-5  5  -5  5]);
```



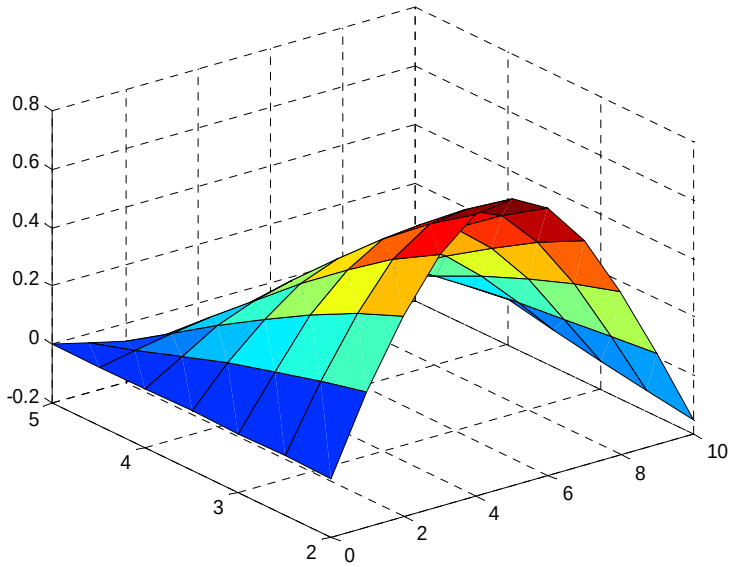

```
%Courbe paramétrique : le cercle
t = [0 :0.1: 2*pi];
x = cos (t);
y = sin (t);
z = t;
t = 0 : pi/100 : 5*pi;
plot3 (cos(t),sin(t),t);
```

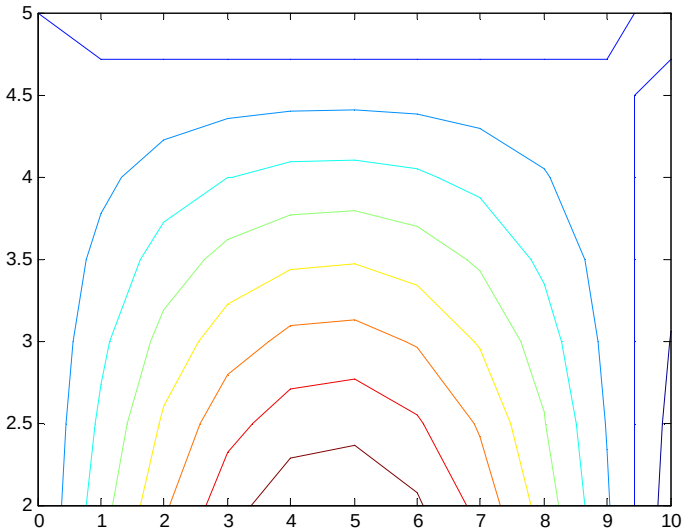


Représentation 2D & 3D (2/4)

Exemple

```
x=0:1:10;
y=2:0.5:5;
for k=1:length(x) % parcours de la grille, colonne après colonne
    for l=1:length(y)  % parcours de la grille, ligne après ligne
        z1(l,k)= sin(x(k)/3)*cos(y(l)/3); % calcul de la matrice z, élément après élément
    end
end
figure (1) ;
surf(x,y,z1);   % visualisation de la surface
figure (2) ;
contour(x,y,z1);
```

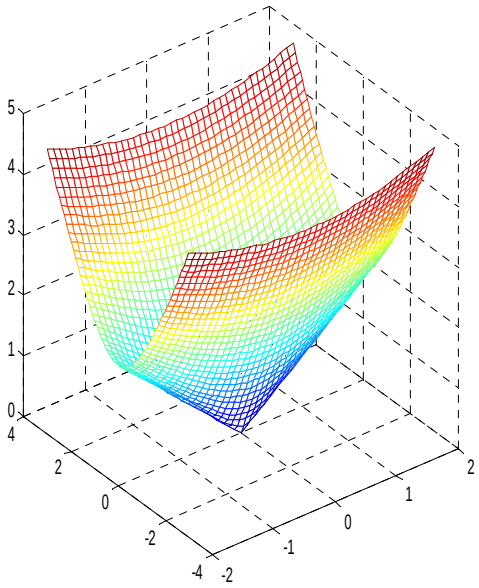


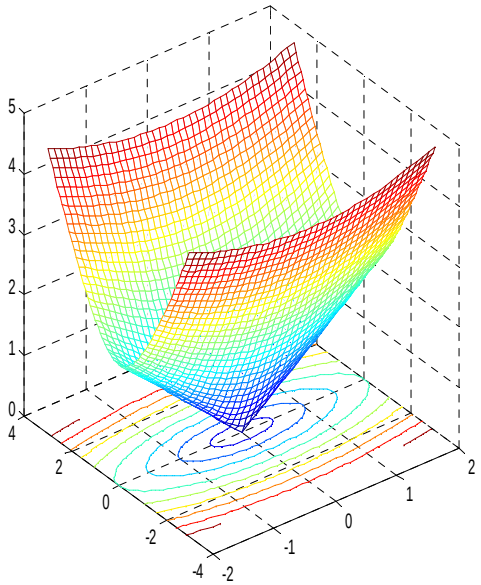


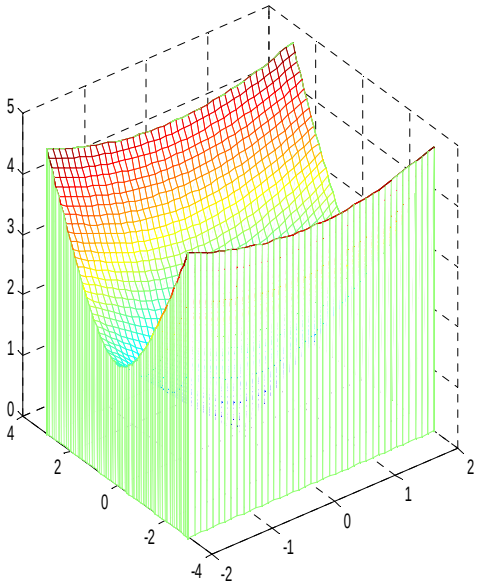
Représentation 2D & 3D (3/4)

Exemple

```
x = -2 : 0.1 : 2;
y = -3 : 0.1 : 3;
[X , Y] = meshgrid(x , y) ;
Z = sqrt (X.^2 + 2.*Y.^2) ;
figure (1)
mesh (X,Y,Z) ;
figure (2)
meshc (X,Y,Z);
figure (3)
meshz (X,Y,Z);
```



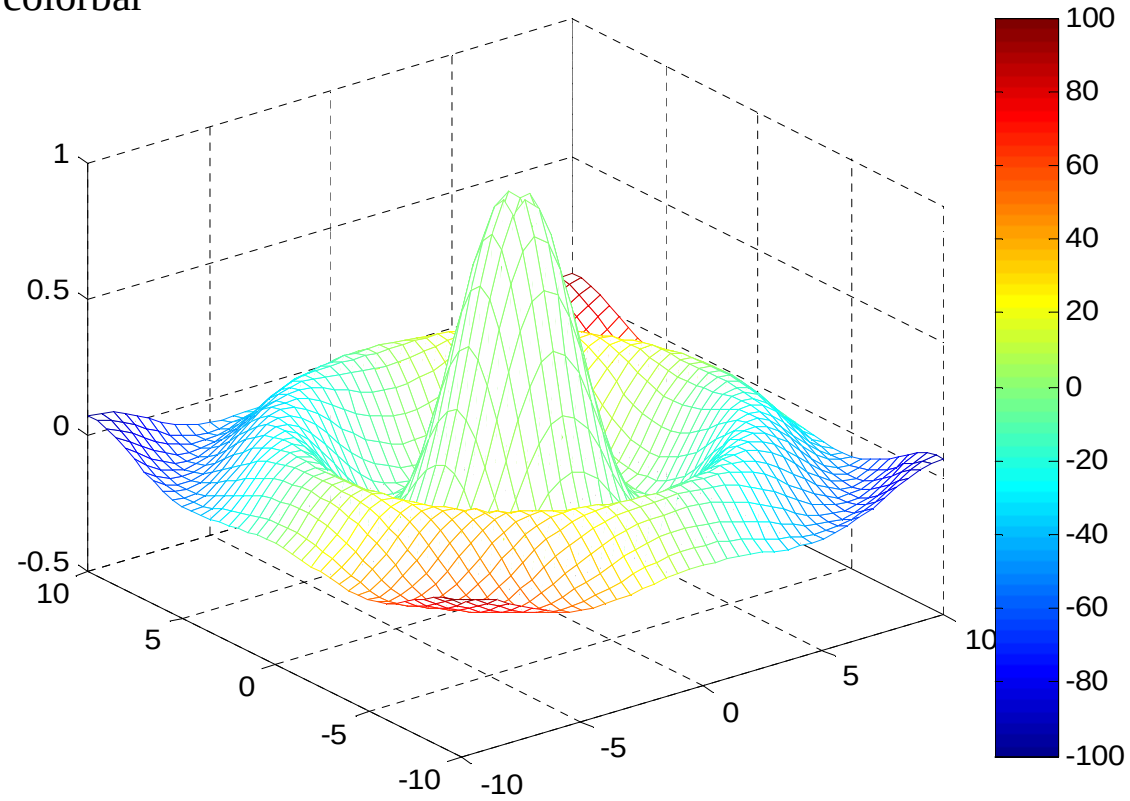




Représentation 2D & 3D (4/4)

Exemple

```
[X,Y] = meshgrid(-10:0.5:10);
R = sqrt(X.^2 + Y.^2);
Z = sin(R)./R;
C = X.*Y;
mesh(X,Y,Z,C)
colorbar
```



Université Mohammed Premier
École Nationale des Sciences Appliquées d'Oujda

Cours d' *Informatique 2 : MATLAB*

Version 3.0 (Janvier 2023)

MATLAB POUR L'INGÉNIEUR

STPI-1

ENSAO, 2022 – 2023

Partie 1

Prof. **Kamal GHOUMID**

Université Mohammed Premier
École Nationale des Sciences Appliquées d'Oujda

Cours d' ***Informatique 2 : MATLAB***

Version 3.0 (Janvier 2023)

MATLAB POUR L'INGÉNIEUR

Chapitre 4

Manipulations des Polynômes

Partie 1

Prof. **Kamal GHOUMID**

Un polynôme peut être représenté par un vecteur, selon les puissances décroissantes, dont le nombre d'éléments est égale au degré du polynôme + 1.

$P(x) = x^5 + 2x^4 - 7x^3 + 12x^2 - 4x + 31$

```
>> P = [1   2   -7   12   -4   31]

P =

     1     2    -7    12    -4    31
```

$Q(x) = -15x^3 + 7x + 3$

```
>>  Q = [-15, 0,7, 3]

Q =

    -15   0  7   3
```

Polynômes (2/24)

- Évaluation d'un polynôme

polyval(P,x)

- Racines d'un polynôme

roots(P)

- Polynôme à partir des racines

poly([racine1, racine2,…])

- Dérivation d'un polynôme

polyder(P)

- Intégration d'un polynôme

polyint(P)

Polynômes (3/24)

- Addition de polynômes

P + Q

- Soustraction de polynômes

P - Q

- Produit de polynômes

conv(P,Q)

- Division de polynômes

[q,r] = deconv(P,Q)

- Décomposition en éléments simples

[r,p,k] = residue(P,Q)

Polynômes (4/24)

```
>> P = [1 -2 -3];
>> Q = [1 -3];

>> polyval(P,1)
ans =
   -4

>> polyval(Q,-3)
ans =
   -6

>> P1 = polyder(P)
P1 =
     2    -2

>> P2 = polyder(P1)
P2 =
     2
```

```
>> polyval(P1,2)
ans =
     2

>> polyval(P2,5)
ans =
     2

>> polyval(P2,sqrt(3))
ans =
     2

>> polyval(P1,0)
ans =
    -2

>> polyval(P1,127)
ans =
   252
```

```
>> P = [1 -3 2];
>> Pd1 = polyder(P)

Pd1 =
     2    -3

>> Pd2 = polyder(Pd1)
Pd2 =
     2

>> Pi1 = polyint(P)
Pi1 =
    0.3333   -1.5000    2.0000         0

>> polyval(Pi1,0)
ans =
     0

>> polyval(Pi1,1)
ans =
    0.8333

>> polyval(Pi1,-2)
ans =
  -12.6667

>> Q = [2 3 -4];
>> Qi1 = polyint(Q)

Qi1 =
    0.6667    1.5000   -4.0000         0

>> polyder(Qi1)
ans =
     2     3    -4

>> polyder(polyder(Qi1))
ans =
     4     3
```

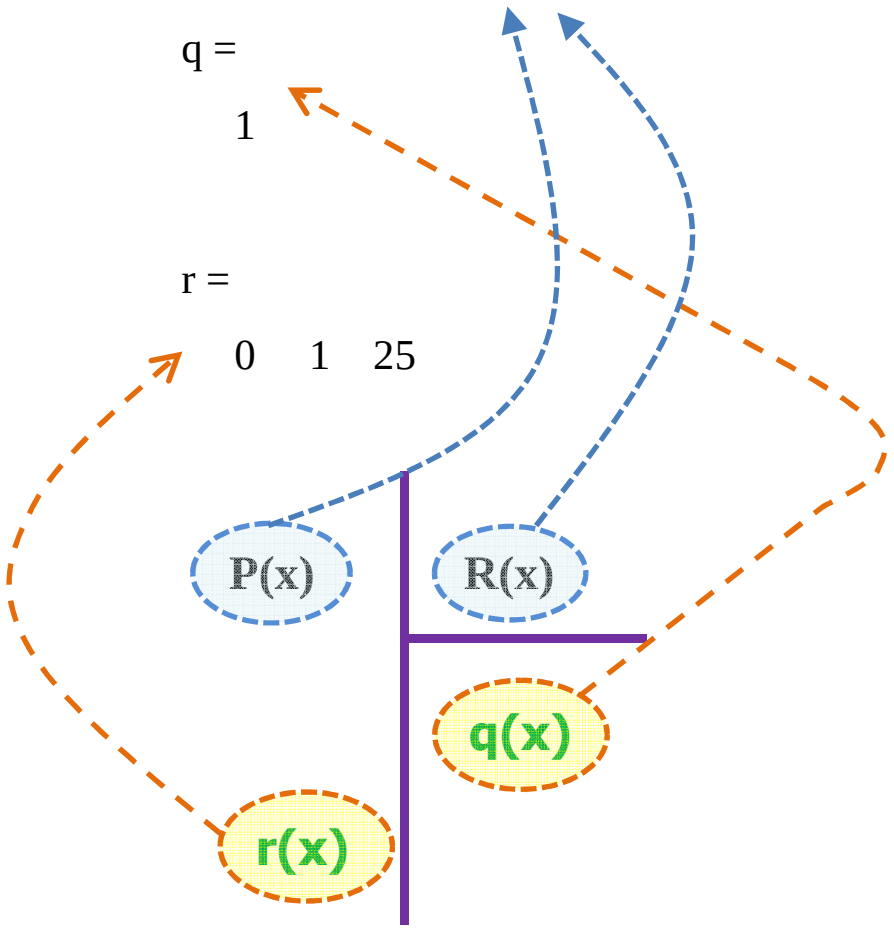
```
>> roots(P)

ans =

   3.0000

  -1.0000


>> roots(Q)

ans =

    3


>> R = poly([-4  7])

R =

    1    -3   -28
```

```
>> P + Q
??? Error using ==> plus
Matrix dimensions must agree.

>> P = [1  -2  -3];
>> Q = [0  1  -3];

>> P + Q
ans =
    1    -1    -6

>> P - Q
ans =
    1    -3     0

>> P - 2*R

ans =

   -1     4    53
```

Polynômes (7/24)

```
>> P = [1 -2 -3];
>> Q = [1 -3];

>> conv(P,Q)
ans =
    1    -5     3     9

>> conv(P,R)
ans =
    1    -5   -25    65    84

>> [q,r] = deconv(P,Q)
q =
    1     1

r =
    0     0     0
```

```
>> [q,r] = deconv(P,R)
q =
     1

r =
     0     1    25
```



Décomposition en élément simple

```
>> P = [1   13];  % C'est le polynôme P(x) = x + 13
>> Q = [1  -1  -2];       % C'est le polynôme Q(x) = x^2 - x - 2
>> [r,p,k] = residue(P,Q)    % décomposition en élément simple

r =
     5
    -4

p =
     2
    -1

k =
     []

% Donc P(x)/Q(x) = 5/(x - 2) – 4/(x + 1)
```

$$\frac{x-13}{x^2-x-2} = \frac{5}{x-2} - \frac{4}{x+1}$$

```
>> P = [-8  22  17  -43];  % C'est le polynôme P(x) = -8x^3 + 22x^2 + 17x - 43
>> Q = [1  -2  -5  6];      % C'est le polynôme Q(x) = x^3 - 2x^2 - 5x + 6
>> [r,p,k] = residue(P,Q)    % décomposition en élément simple

r =
   -1.0000
    5.0000
    2.0000

p =
    3.0000
   -2.0000
    1.0000

k =
    -8
```

% Donc P(x)/Q(x) = -1/(x - 3) + 5/(x + 2) + 2/(x - 1) - 8

$$\frac{-8x^3 + 22x^2 + 17x - 43}{x^3 - 2x^2 - 5x + 6} = \frac{-1}{x-3} + \frac{5}{x+2} + \frac{2}{x-1} - 8$$

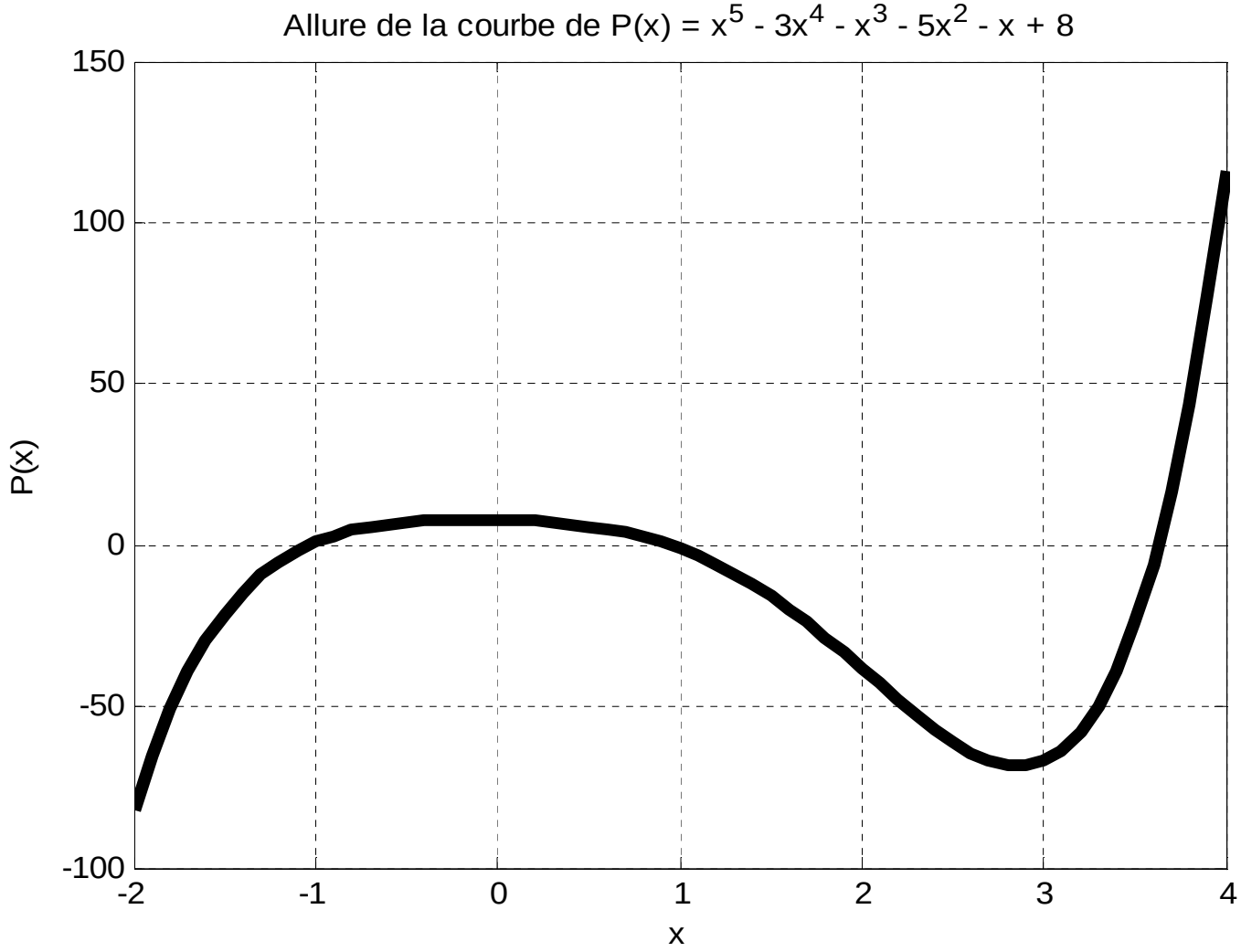
Représentation graphique d'un polynôme

```
P = [1  -3  -1  -5  -1  8];

x = -2:0.1:4;

y = polyval(P,x);

plot(x,y,'k','linewidth',4);
```

```
Px = 'x^5 -3*x^4 - x^3 -5*x^2 - x + 8';

fplot(Px,[-2,4]);
```

```
grid on

xlabel('x');

ylabel('P(x)');

title('Allure de la courbe de P(x) = x^5 -3x^4 - x^3 -5x^2 - x +8');
```

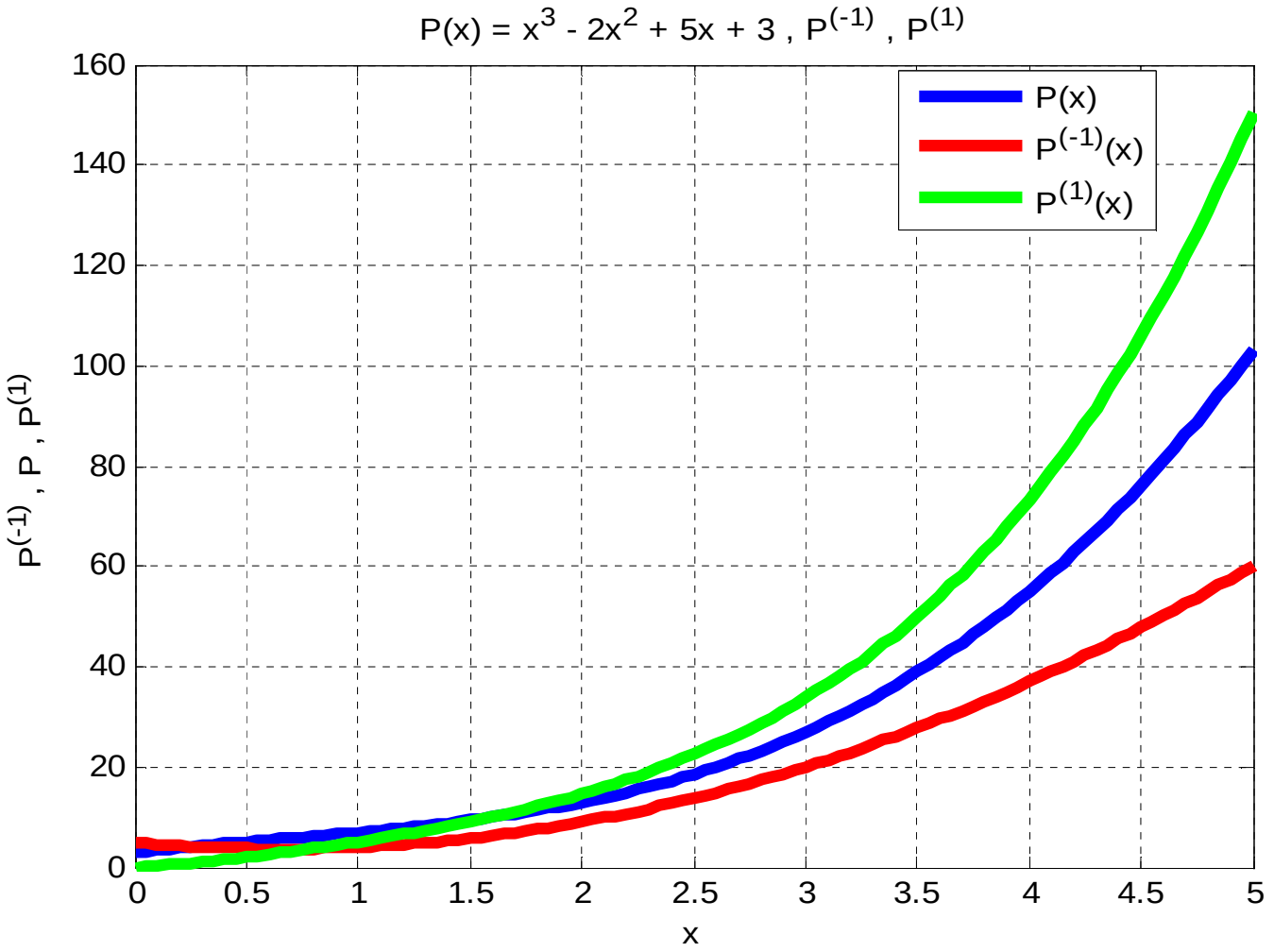
Représentation graphique d'un polynôme



Représentation graphique d'un polynôme

```
P = [1 -2 5 3];
Pd1 = polyder(P);
Pi1 = polyint(P);
x = 0:0.05:5;
U = polyval(P,x);
V = polyval(Pd1,x);
W = polyval(Pi1,x);
plot(x,U,'linewidth',4);
hold on
plot(x,V,'r','linewidth',4);
hold on
plot(x,W,'g','linewidth',4);
xlabel('x');
grid on
legend('P(x)','P^{(-1)}(x)','P^{(1)}(x)')
ylabel('P^{(-1)} , P , P^{(1)}');
title('P(x) = x^3 - 2x^2 + 5x + 3 , P^{(-1)} , P^{(1)}');
```

Représentation graphique d'un polynôme

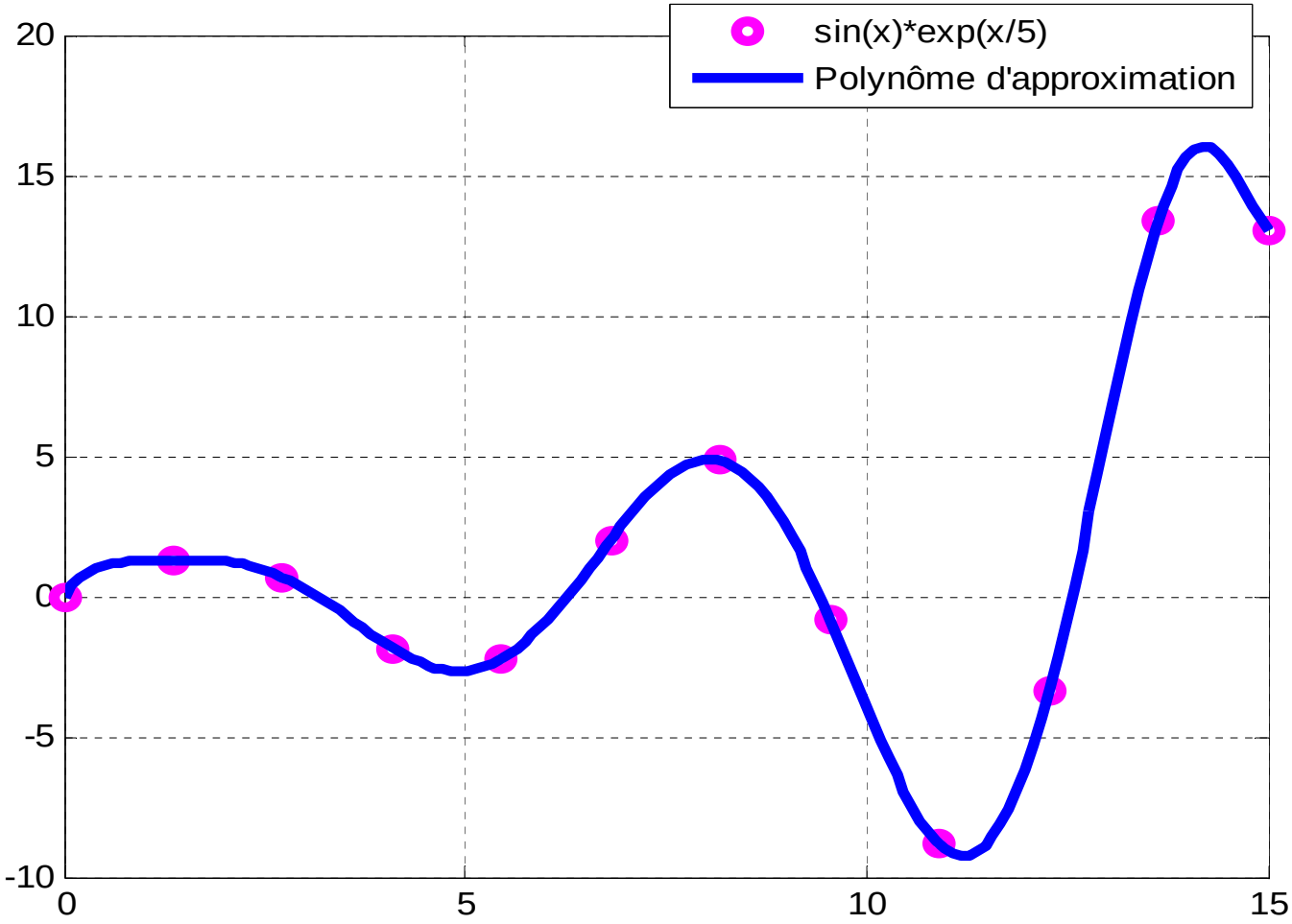


➡ **Approximation polynômiale d'une fonction** (voir l'exemple).

```
>> x1 = linspace(0,15,12);
>> y1 = sin(x1).*exp(x1/5);
>> x2 = linspace(0,15,150);
>> P = polyfit(x1,y1,10); %  polyfit approxime les points de y1 par le polynôme P de degré 10
>> y2 = polyval(P,x2);
>> plot(x1,y1,'om',x2,y2,'linewidth',3);
legend('sin(x).*exp(x/5)','Polynôme d''approximation');
```

```
>>  P =  %les coefficients du polynôme P(x) sont (dans l'ordre décroissant)
  Columns 1 through 11
    0.0000   -0.0000    0.0001    0.0002   -0.0272    0.3139   -1.6226    4.2346
  Columns 9 through 11
 -5.8780    4.2399   -0.0001
```

Approximation polynômiale d'une fonction

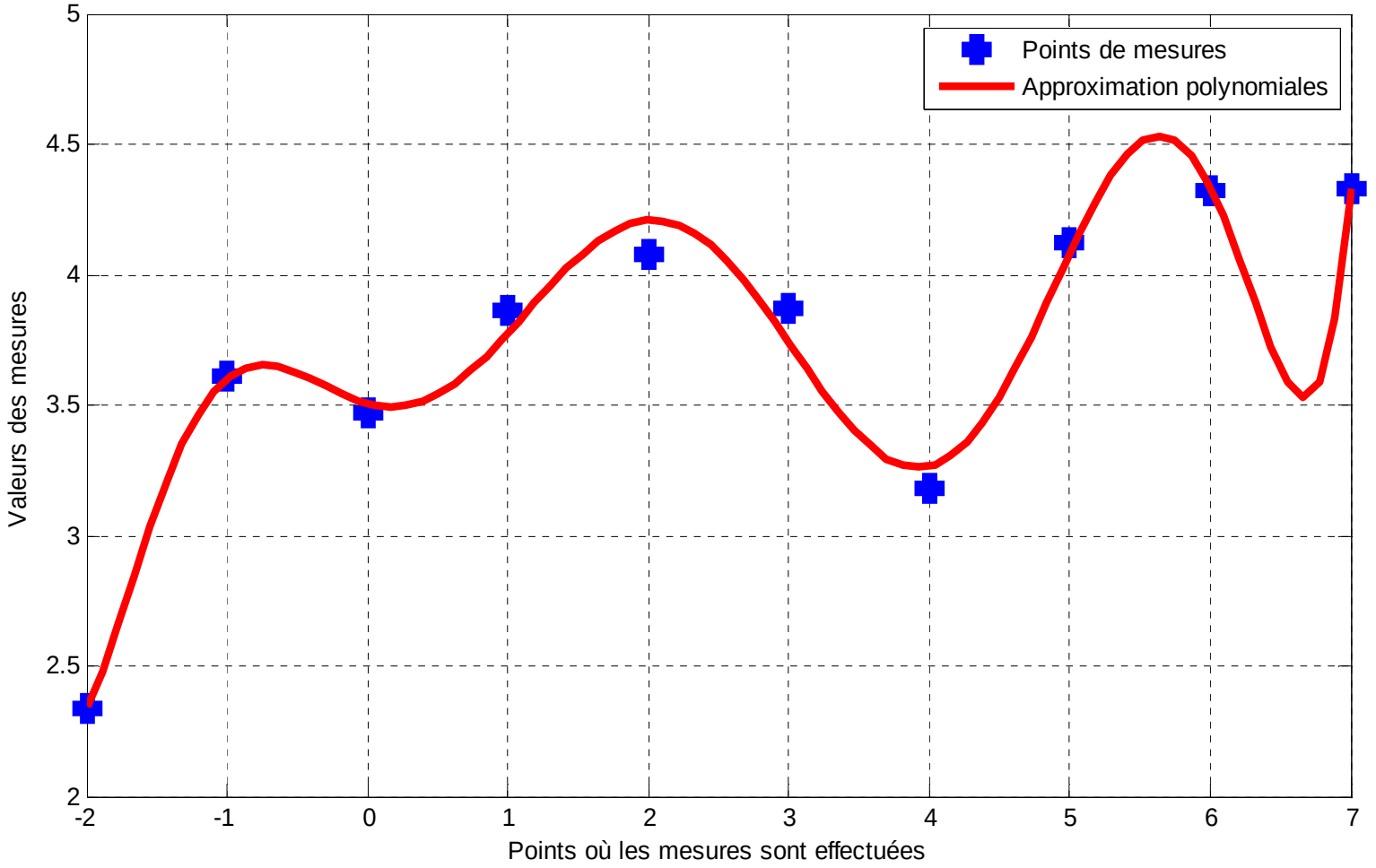


Approximation polynômiale des points de mesures

```
>> x1 = [-2 : 7];
>> M = [2.34 3.61 3.47 3.86 4.08 3.87 3.18 4.12 4.32 4.33];
>> x2 = linspace(-2,7,80);
>> P = polyfit(x1,M,8)
        0.0002   -0.0038    0.0200   -0.0075   -0.1842    0.2435    0.3447   -0.1456    3.5068
>> Approx = polyval(P,x2);
>> plot(x1,M,'+','linewidth',12);
>> hold on
>> plot(x2,Approx,'r','linewidth',3);
>> grid on
>> xlabel('Points où les mesures sont effectuées');
>> ylabel('Valeurs des mesures');
>> title('On a fitté les points M par le polynôme P(x) = 0.0002x^8 - 0.0038x^7 + 0.0200x^6 -
0.0075x^5 - 0.1842x^4 + 0.2435x^3 + 0.3447x^2 - 0.1456x + 3.5068');
legend('Points de mesures','Approximation polynomiales')
```

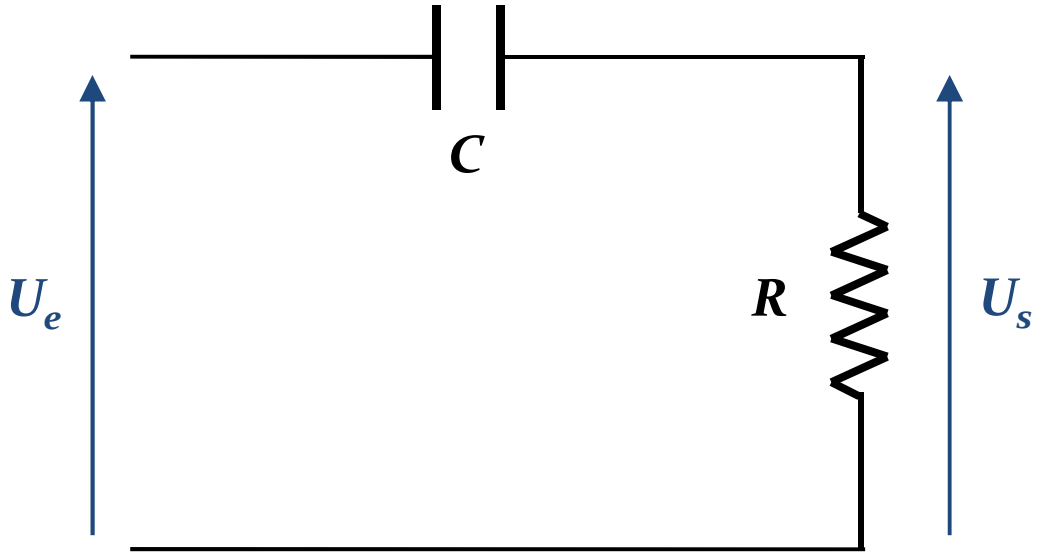
Approximation polynômiale des points de mesures

On a fitté les points M par le polynôme $P(x) = 0.0002x^8 - 0.0038x^7 + 0.0200x^6 - 0.0075x^5 - 0.1842x^4 + 0.2435x^3 + 0.3447x^2 - 0.1456x + 3.5068$



Tracés des fonctions de transferts

▪ Circuit CR en série : Filtre passe-haut



$$H(P) = \frac{U_s(P)}{U_e(P)} = \frac{RCp}{1 + RCp} \quad \xleftrightarrow{p = j\omega} \quad H(\omega) = \frac{jRC\omega}{1 + jRC\omega}$$

p : Variable de Laplace

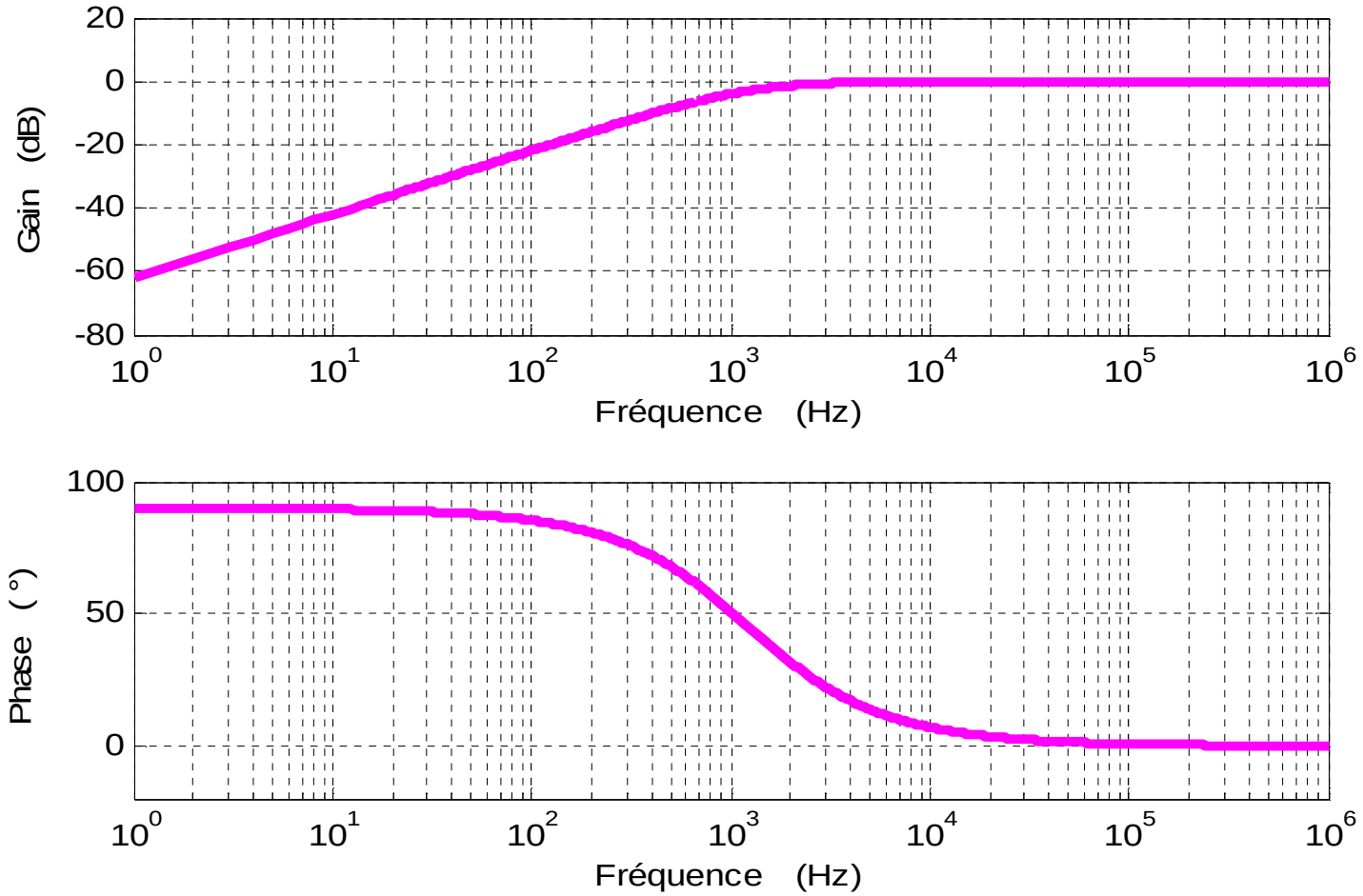
Tracés des fonctions de transferts

▪ Circuit CR en série : Filtre passe-haut

```
disp('%%%%%%%%---------- Cours : Circuit CR ----------%%%%%%%%');
R = 4000; %Résistance en Ohm
C = 0.2e-6; %Capacité en F
w = 0:1e6; %Plage de variation de la pulsation
Num = [R*C,0]; %Numérateur de H(p)
Denom = [R*C,1]; %Dénominateur de H(p)
H = polyval(Num,j*w)./polyval(Denom,j*w);
Feq_Coupure = 1/(2*pi*R*C);
disp(['Feq_Coupure = ',num2str(Feq_Coupure),' Hz']);

subplot(2,1,1);
semilogx(w,20*log10(abs(H)),'m','linewidth',3);
axis([0,1e6,-80 20]);
grid on
xlabel('Fréquence   (Hz)');
ylabel('Gain   (dB)');

subplot(2,1,2);
semilogx(w,angle(H)*180/pi,'m','linewidth',3);
axis([0,1e6,-20 100]);
grid on
xlabel('Fréquence   (Hz)');
ylabel('Phase   ( ° )');
```

Tracés des fonctions de transferts

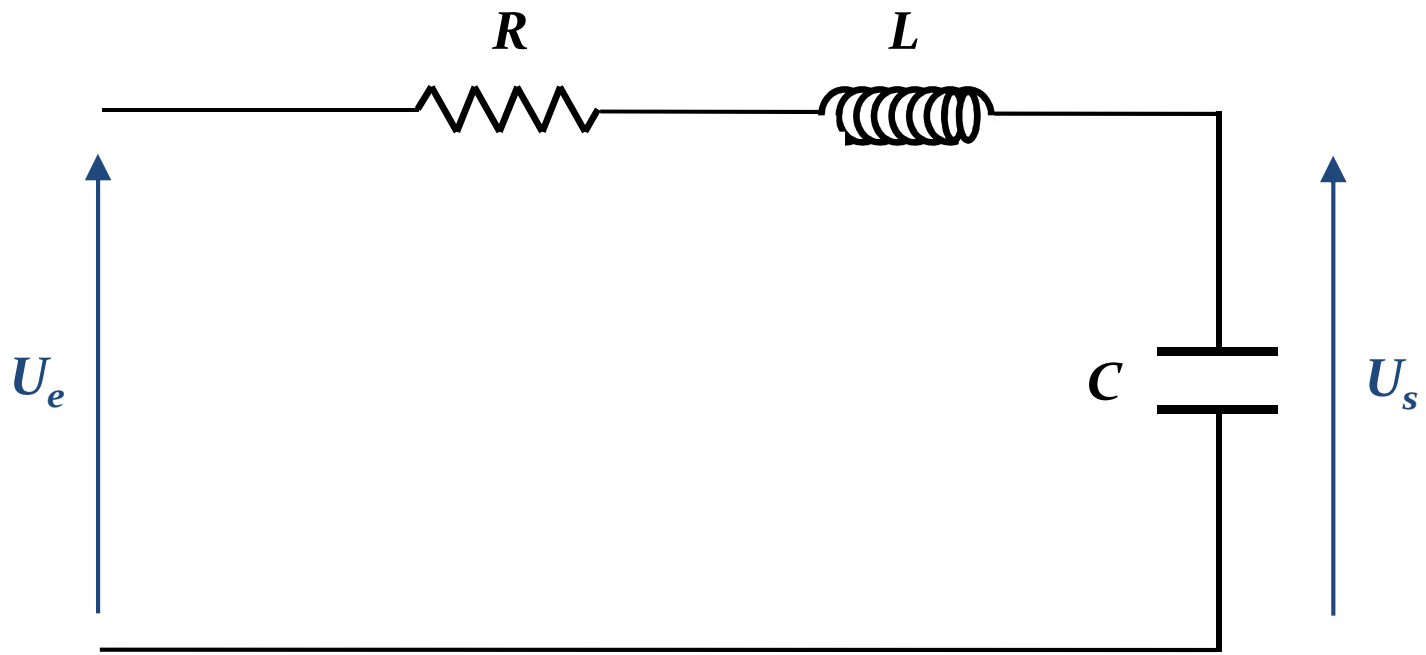
- *Circuit CR en série : Filtre passe-haut*



Feq_Coupure = 198.9437 Hz

Tracés des fonctions de transferts

▪ Circuit RLC en série : Filtre passe-bas d'ordre 2



$$H(P) = \frac{U_s(P)}{U_e(P)} = \frac{1}{1 + RCp + LCp^2} \quad \xleftrightarrow{p = j\omega} \quad H(\omega) = \frac{j}{1 - LC\omega^2 + jRC\omega}$$

p : Variable de Laplace

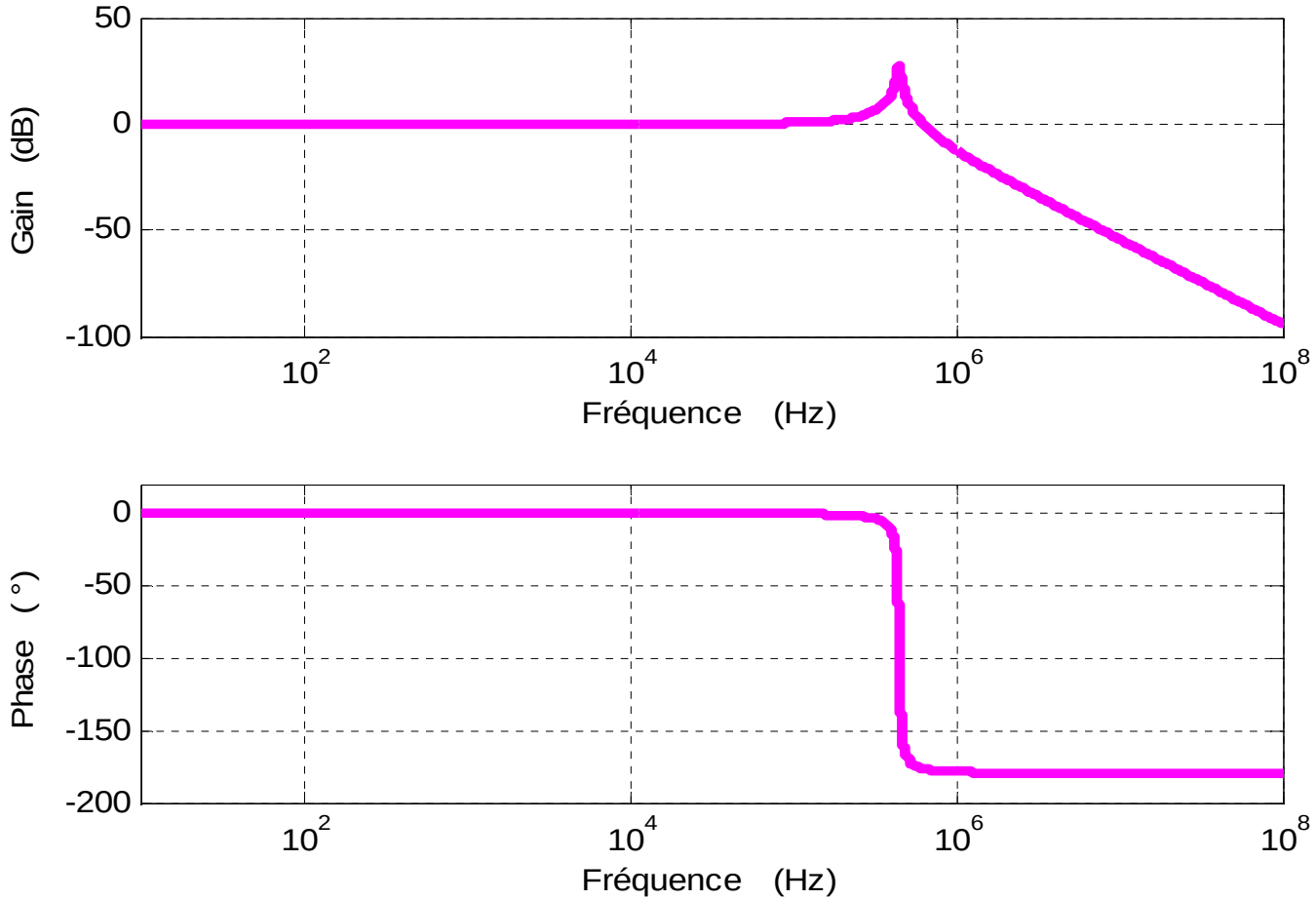
Tracés des fonctions de transferts

▪ Circuit RLC en série : Filtre passe-bas d'ordre 2

```
disp('%%%%%%%%---------- Cours : Circuit RLC ----------%%%%%%%%');
R = 10; %Résistance en Ohm
L = 0.5e-3; %Inductance en H
C = 10e-9; %Capacité en F
w = 0:10:1e8; %Plage de variation de la pulsation
Num = [1]; %Numérateur de H(p)
Denom = [L*C,R*C,1]; %Dénominateur de H(p)
H = polyval(Num,j*w)./polyval(Denom,j*w);
Feq_Coupure = 2*pi/sqrt(L*C);
Q = 1/(R*C*Feq_Coupure); %Facteur d'amortissement
disp(['Feq_Coupure = ',num2str(Feq_Coupure),' Hz']);
disp(['Facteur d''amortissement Q = ',num2str(Q)]);

subplot(2,1,1);
semilogx(w,20*log10(abs(H)),'m','linewidth',3);
axis([0,1e8,-100 50]);
grid on
xlabel('Fréquence   (Hz)');
ylabel('Gain   (dB)');

subplot(2,1,2);
semilogx(w,angle(H)*180/pi,'m','linewidth',3);
axis([0,1e8,-200,20]);
grid on
xlabel('Fréquence   (Hz)');
ylabel('Phase   ( ° )');
```

Tracés des fonctions de transferts

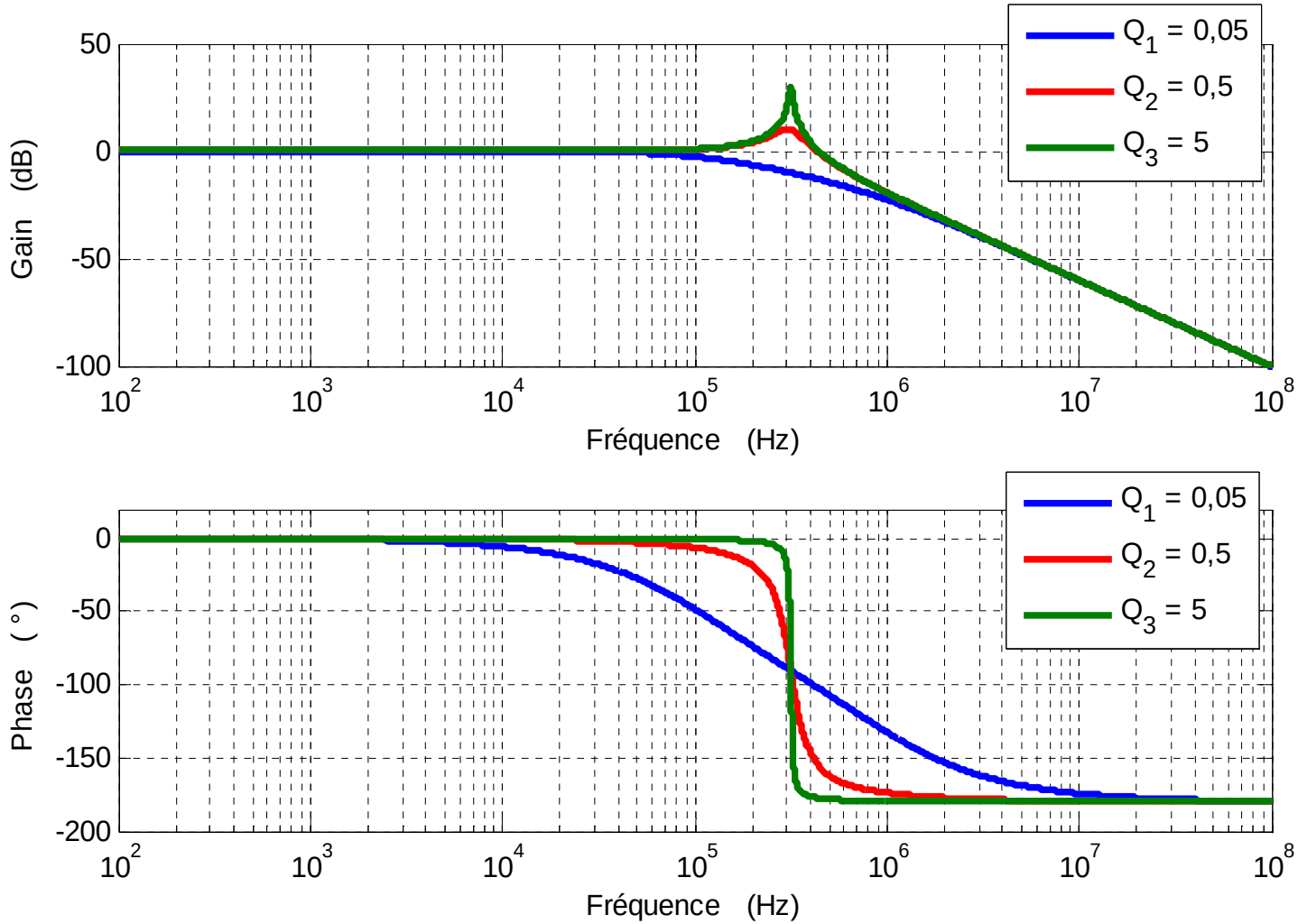
- *Circuit RLC en série : Filtre passe-bas d'ordre 2*



Feq_Coupure = 2809925.8924 Hz
Facteur d'amortissement Q = 3.5588

Tracés des fonctions de transferts

- *Circuit RLC en série : Filtre passe-bas d'ordre 2*



Université Mohammed Premier
École Nationale des Sciences Appliquées d'Oujda

Cours d' *Informatique 2 : MATLAB*

Version 3.0 (Janvier 2023)

MATLAB POUR L'INGÉNIEUR

STPI-1

ENSAO, 2022 – 2023

Fin de la Partie 1

Prof. **Kamal GHOUMID**