Introduction à MATLAB

Didacticiel pour les étudiants de INF1005A
édition 1.0, 27 juillet 2005

Audrey Girouard
École Polytechnique de Montréal

Table des matières

1 Introduction et ouverture d'une session Matlab

Matlab est un logiciel permettant l'optimisation des calculs scientifiques. Développé au départ pour le calcul matriciel, d'où l'abréviation *MATrix LABoratory*, Matlab permet la résolution de problèmes grâce à des algorithmes, des graphiques, des simulations, etc.

Le didacticiel suivant est une brève introduction au monde de Matlab. Elle présentera l'essentiel des notions nécessaires pour se débrouiller avec le logiciel Matlab 7.0.

Note : les exemples dans ce didacticiel ont été réalisés dans des fichiers-m. Ceci explique que les instructions sont regroupées et séparées des résultats qui s'inscrivent dans l'affichage.

1.1 Introduction

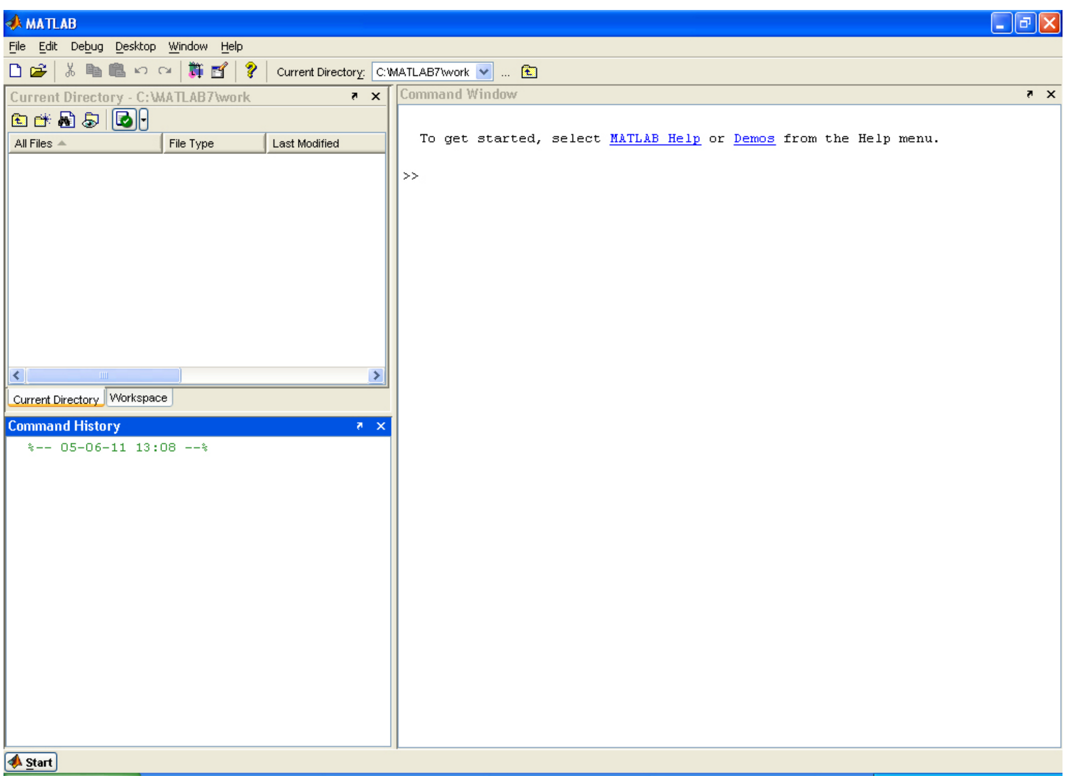
Pour lancer Matlab, il suffit de cliquer sur le raccourci Matlab situé sur le Bureau ou dans le menu Démarrer, selon l'ordinateur. Matlab est un gros programme. Il est donc normal qu'il prenne un peu de temps à démarrer, en fonction des autres programmes déjà ouverts sur votre ordinateur.

EXERCICE :

Démarrez Matlab.

1.2 Fenêtres

L'interface de Matlab correspond à celle-ci.



Par défaut, on retrouve 3 fenêtres. La fenêtre en haut à gauche contient le Workspace. En dessous, on retrouve le Current Directory ainsi que le Command History. Enfin, à droite, il y a la Command Window. Pour "sortir" une fenêtre de l'interface, il suffit de cliquer sur le bouton représentant une flèche en haut à droite de chacune des fenêtres.

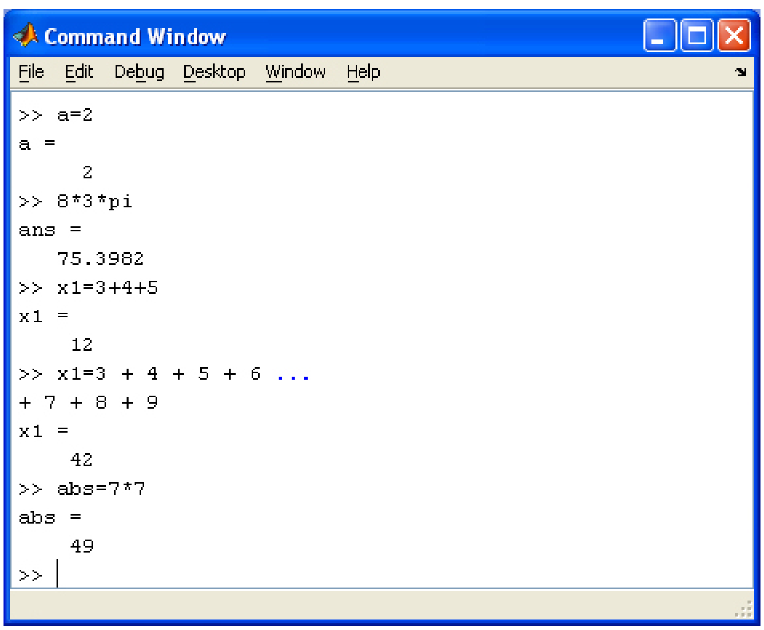
L'interface contient aussi des barres de menus. Voici les commandes accessibles dans les menus :

Edit → Clear Command Window	Permet d'effacer les instructions et/ou les résultats visibles dans la Command Window
Edit → Clear Command History	Efface les commandes précédentes mises en mémoire
Edit → Clear Command Workspace	Efface de la mémoire les variables stokées
View	Détermine les aspects visuels des différentes fenêtres.

La barre de boutons permet l'accès rapide à certaines commandes. On peut déterminer le Current Directory par cette barre sans avoir à passer par la fenêtre du même nom. En cliquant-glissant les barres horizontales et verticales séparant les différentes fenêtres de l'interface, il est possible d'agrandir ou de rapetisser ces fenêtres.

1.2.1 Command Window

L'une des plus importantes fenêtres de Matlab, la Command Window traite des instructions données. C'est après l'invite ("prompt") » qu'il faut entrer les instructions demandées. Les résultats s'afficheront dès le retour de ligne.



On peut éviter de retaper une instruction en pesant sur la flèche vers le haut (↑), ce qui affichera l'instruction précédente. Continuez à peser ↑ jusqu'à l'instruction recherchée.

Il est facile d'observer sur l'image les instructions données ainsi que les réponses obtenues :

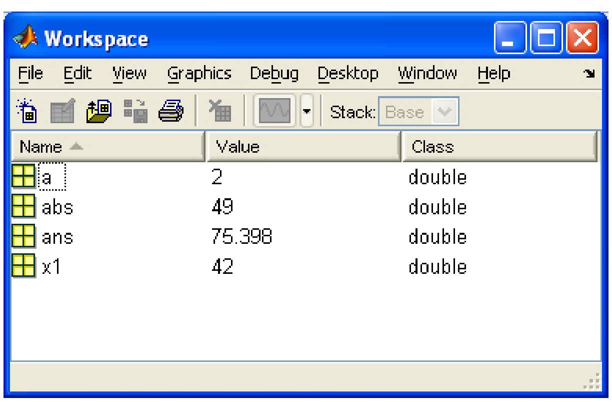
Instructions	Affichage
`>> a=2`	`a =` `2`
`>> 8*3*pi`	`ans =` `75.398`
`>> x1=3+4+5`	`x1 =` `12`
`>> x1=3 + 4 + 5 + 6 ...` `+ 7 + 8 + 9`	`x1 =` `42`
`>> abc=7*7`	`abc =` `49`

1.2.2 Boutton Start

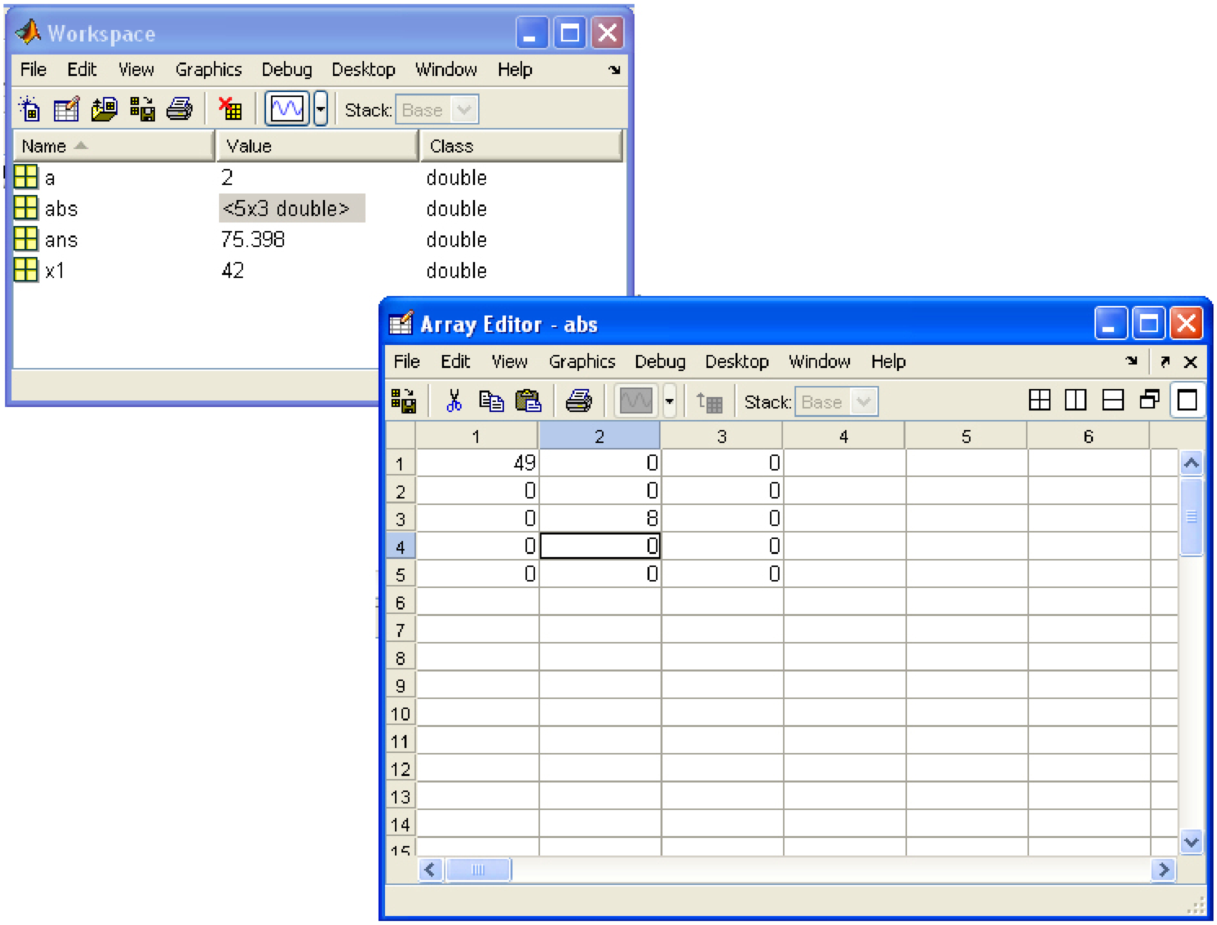
Le boutton *Start* est un outil pour ouvrir rapidement certaines fonctions de Matlab (outils, démonstrations et documentation). Pour plus de renseignements, consultez sa rubrique dans l'aide de Matlab.

1.2.3 Workspace

La fenêtre nommée Workspace permet de visualiser les variables mises en mémoire. On y retrouve leur nom, leur dimensions ainsi que le type de variable. Matlab étant basé sur les matrices, toutes les variables sont constituées de plusieurs dimensions : un scalaire est une matrice 1×1 et un vecteur est une matrice $1 \times n$ ou $n \times 1$, etc. Il est possible d'effacer certaines variables ainsi que de les éditer. Pour toutes les effacer, utilisez la commande Clear Workspace dans le menu Edit.



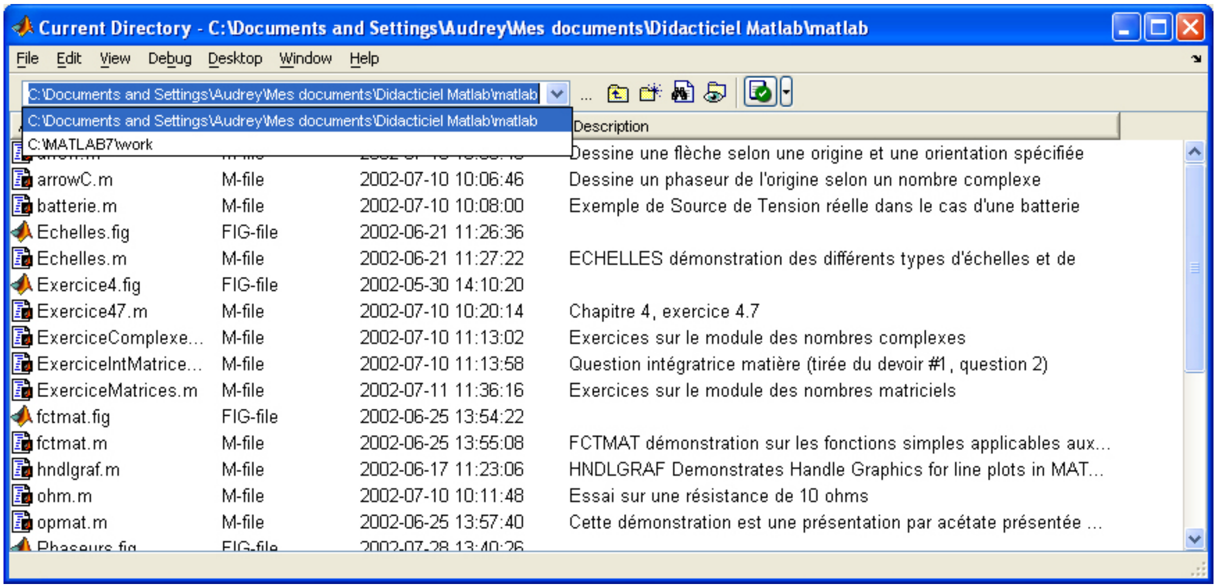
En double-cliquant sur une variable, la fenêtre Array Editor (Éditeur de tableau) apparait. Cette fenêtre contient les valeurs des variables et permet de les modifier. Dans l'exemple suivant, la variable initiale est une matrice 1×1. En ajoutant des valeurs dans les cases adjacentes, on a transformé la matrice pour qu'elle devienne de dimensions 5×3. Dans l'exemple, la case (3,2) a eu comme valeur 8 plutôt que 0.



1.2.4 Current Directory

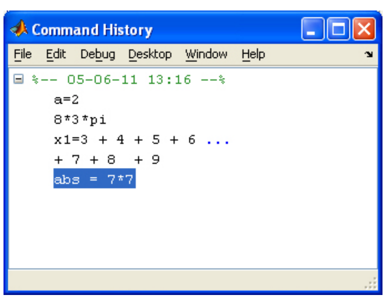
Le Current Directory est le répertoire courant où sont enregistrés les fichiers-M (voir section 1.3). Il est fortement conseillé de se créer un répertoire autre que celui fournit par Matlab afin de mieux gérer les fichiers contenus dedans. Bien que vous l'apprendrez au cours du didacticiel, quelques principes généraux par rapport au Current Directory sont nécessaires :

– Pour compiler un fichier-M, il doit être enregistré dans le répertoire courant.
– Si un fichier-M appelle une fonction autre qu'une fonction de Matlab (i.e. une fonction crée par l'utilisateur), le fichier-M appelant la fonction ainsi que le fichier-M définissant la fonction doivent être dans le même répertoire courrant.



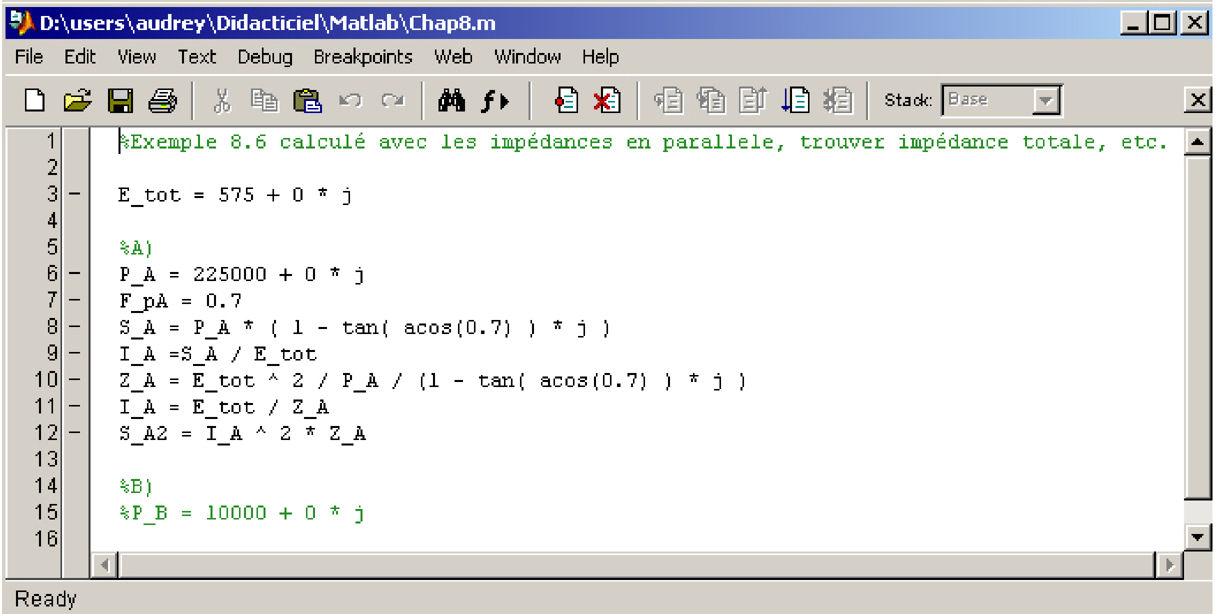
1.2.5 Command History

Le Command History inscrit les commandes au fur et à mesure qu'elles sont appellées dans le Command Window. Elle garde en mémoire ces commandes, ainsi que la date et l'heure d'ouverture de chacune des sessions de Matlab.



1.2.6 Editor/Debugger

Pour traiter des fichiers-M (voir section 1.3), il faut utiliser l'éditeur/débogueur de Matlab. Cette fenêtre ne fait pas partie de l'interface de base de Matlab et s'ouvre lorsqu'on ouvre un fichier-M ou lorsqu'on crée un nouveau fichier-M.



Sur la barre des boutons, un bouton est essentiel : le bouton RUN compile le programme, i.e. exécute les commandes du programme. Il peut aussi être exécuté avec F5.

Un autre élément intéressant de l'Éditeur consiste à laisser le curseur sur une variable, et sa valeur apparait (si le programme a été compilé au moins une fois).

Démonstrations :

Pour des démonstrations pas à pas sur l'environnement de Matlab, ouvrez Matlab, puis entrez `demo` dans la fenêtre de commande. Sélectionnez *Matlab - Desktop Tools and Development Environment*. Vous trouverez une demi-douzaine de démonstrations.

1.3 Les fichiers-M

Afin d'éviter d'avoir à retaper une série de commandes, il est possible de créer un programme MATLAB, connu sous le nom de «fichier-M» («M-file»), le nom provenant de la terminaison « `.m` » de ces fichiers. Il s'agit, à l'aide de l'éditeur de MATLAB, de créer un fichier en format texte qui contient une série de commandes MATLAB. Pour créer un fichier-M, il faut aller dans le menu File → New → M-File ou cliquez sur le bouton de la page blanche. L'enregistrement se fait normalement dans le répertoire courant. Une fois le fichier sauvegardé (sous le nom `nomdefichier.m` par exemple), il s'agit de l'appeler dans MATLAB à l'aide de la commande :

```
>> nomdefichier
```

Les commandes qui y sont stockées seront alors exécutées et les résultats s'afficheront dans la fenêtre de commande (Command Window). Si vous devez apporter une modification à votre série de commandes, vous n'avez qu'à modifier la ligne du fichier-M en question et à réexécuter le fichier-M en entrant le nom du fichier dans MATLAB à nouveau (essayez la touche ↑).

D'autres manières d'exécuter le fichier-M est de cliquer sur le bouton Run , ou aller dans le menu Debug→Run dans l'Éditeur/débogueur, ou peser F5.

Les fichiers-M vous évitent d'avoir à retaper une série de commandes à répétition et vous permettent de conserver vos instructions, commandes et calculs grâce à l'enregistrement. C'est la procédure recommandée pour vos travaux pratiques.

EXERCICE :

Création d'un fichier-M - Procédure
- Créez un fichier par le bouton ou par le menu
- Inscrivez quelques instructions. Par exemple, inscrivez les instructions suivantes :
```
A=8
B=7
C=A+B
```
- Enregistrez le fichier-m dans le répertoire courant. Si vous ne le faîtes pas, Matlab vous demandera de l'enregistrer avant de le compiler.
- Retournez dans Matlab : les réponses devraient être apparues dans la fenêtre de commande. Ces réponses devraient être :
```
>> A =
     7
B =
     9
C =
    16
```

1.4 Le " ;" et le "..."

Le point-virgule à la fin d'une ligne signale à MATLAB de ne pas retourner le résultat de l'opérétion à l'écran. Une pratique courante est de mettre des « ;» à la fin de toutes les lignes et d'enlever certains de ceux-ci lorsque quelque chose ne tourne pas rond dans notre programme, afin de voir ce qui se passe.

Dans l'éditeur/débogueur tout comme dans la fenêtre de commande, l'utilisation de «...» est utile pour continuer sur la ligne d'en dessous l'instruction en cours.

Les deux instructions sont identiques à l'exception du « ;» à la fin de la 2e.

```
>> B = pi * 3^2
B =
       28.274
>> B = pi * 3^2;
>>
```

Les deux instructions suivantes donnent le même résultat

```
>> A = 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9
A =
    45
>> A = 1 + 2 + 3 + 4 + 5 ...
 + 6 + 7 + 8 + 9
A =
    45
```

1.5 Aide

1.5.1 Aide

En plus des fonctionnalités de base de MATLAB, une vaste bibliothèque de fonctions (les « toolbox » en langage MATLAB) sont à votre disposition. Pour avoir une liste des familles de fonctions qui sont disponibles, entrez la commande `help`. Pour voir la liste des fonctions d'une famille de fonctions, on peut entrer `help matfun` par exemple, afin de voir la liste des fonctions matricielles. Pour obtenir de l'information sur une fonction en particulier, il s'agit d'utiliser la commande `help` avec le nom de la fonction, soit `help sin` pour avoir de l'information sur la fonction `sin`. Si la fonction n'a pas été compilée afin de gagner de la vitesse d'exécution, il est possible de voir le code source en entrant `type arrow`, par exemple.

1.5.2 Trouver une fonction

La commande `lookfor` permet de trouver dans les répertoires de documentation toutes les fonctions contenant le mot cherché. Par exemple, pour voir la liste des fonctions liées au hasard, on entre `lookfor random` et on obtient 13 fonctions utilisant le hasard.

1.5.3 Variables utilisées

La commande qui indique les variables utilisées est `who`. Elle donne simplement le nom de toutes les variables utilisées. La commande `whos` quant à elle donne le nom, les dimensions, l'espace mémoire utilisé ainsi que le type de la variable.

```
>> who

Your variables are:

DATAE  DATAG  DATAI  DATAR  E      I

>> whos
  Name        Size           Bytes  Class

  DATAE       1x7               56  double array
  DATAG       7x7              392  double array
  DATAI       1x7               56  double array
  DATAR       7x7              392  double array
  E           7x1               56  double array
  I           7x1               56  double array

Grand total is 126 elements using 1008 bytes
```

Nous voyons que tous les objets créés sont des matrices avec des composantes stockées comme des réels en double précision, même ce que nous avons entré sous la forme d'entiers.

2 Calculs simples

2.1 Entrée de données

Comme expliqué dans le chapitre 1, il exite 2 manières d'entrer et d'exécuter des instructions : par le Command Window, qui les exécute immédiatement après le retour de chariot, ou par un fichier-m qui les exécute lorsque le bouton RUN est cliqué. Peu importe la manière dont les instructions sont compilées, la réponse s'affichera dans le Command Window.

2.2 Format de données

Les réponses numériques données suites aux instructions peuvent être exprimées dans plusiques formats. Pour changer le format, on peut le modifier dans l'éditeur de tableau (voir section 1.2.3). C'est aussi possible de modifier le format en allant dans le menu File → Preferences. . ., puis dans la section Command Window. Il y a plusieurs choix dans le Numeric Format, ainsi que 2 dans le Numeric display. Le premier régit le format des donées alors que le second détermine s'il y aura des lignes vides entre les instructions et entre les différentes lignes des réponses.

Dans le Numeric Format, le choix suivant sont offerts (les exemples sont tous produit à partir de l'équation y = pi) :

+	Met un + si la donnée est positive, un - si elle est négative et un blanc si elle est nulle	+
bank	Mets en dollard et en sous	3.14
hex	Hexadecimal	400921fb54442d18
long	15 nombres après la virgule	3.14159265358979
long e	15 nombre après la virgule en notation scientifique	3.141592653589793e+000
long g	Ce qui s'applique le plus entre le long et le long e	3.14159265358979
rational	Fraction réduite	2973/355
short	5 chiffres après la virgule	3.1416
short e	5 chiffres après la virgule en notation scienfique	3.1416e+000
short g	Ce qui s'applique le plus entre le short et le short e	3.1416

2.3 Entrée d'équations

Pour associer une instruction à une variable, on indique le nom de la variable, suivi d'un signe =, puis de l'instruction en question. Si on n'indique pas de variable, la réponse sera associé à la variable `ans` (pour answer). Les variables peuvent être constitués d'au maximum 31 lettres ou chiffres. Les majuscules et les minuscules produisent deux variables différentes. Ainsi, `A` et `a` sont différents. Enfin, pour obtenir la valeur des données d'une variables, il suffit simplement de taper son nom.

2.4 Fonctions de base

Il est aussi possible de créer ses propres fonctions MATLAB. Cette section n'est cependant pas élaborée dans ce didacticiel.

En tout temps, les paranthèses () indiquent la priorité des opérations.

Pour le reste, il suffit d'entre les calculs à effecter simplement. L'addition, la soustraction, la multiplication, la division sont les opérateurs habituels (respectivement +, -, *, /).

Les puissances s'indiquent avec l'opérateur $\wedge$. On calcul les puissances de e avec la fonction `exp(x)` où x est la puissance de e. Pour calculer la racine carrée, on utilise la fonction `sqrt(x)`. Pour les autres racines, la fonction est `roots(x)` qui produira une matrice (voir section 3.1) contenant les racines.

Pour calculer des logarithmes, il existe 3 fonctions : la fonction `log(x)` calcule les logarithmes naturels (ln), la fonction `log2(x)` calcule les logarithmes en base 2, et la fonction `log10(x)` calcule les logarithmes en base 10.

Le nombre π s'indique `pi`, les lettres pour les nombres complexes `i` et `j` s'indiquent ainsi aussi. Une variable retournée pourrait être l'infinit, décrit par `inf`, lors d'une division par zero. La variable `NaN` (Not a Number) est retournée pour des opérations tel les 0/0 ou Inf/Inf.

Les fonctions trigonométriques principales s'inscrivent simplement ainsi : `sin(x)`, `cos(x)`, `tan(x)`, `asin(x)`, `acos(x)`, `atan(x)`, `atan2(x)`. Dans la série des hyperboliques, il y a `sinh(x)`, `cosh(x)`, `tanh(x)`.

Les fonctions de fonctions sont permises et on les écrit par l'entremise de paranthèses.

Pour plus de fonctions, veuillez consulter le catalogue à la fin du didacticiel. Sinon, consultez l'aide.

Instructions	**Affichage**

`a = (pi^2 + 3)/sin(8)`
`b = sinh(3*9*exp(2))`
`c = sqrt(16) - tan(.5)`
	`>> a =`
	`      13.0080`
	`b =`
	`  2.2013e+086`
	`c =`
	`      3.4537`

`%Exemple en électrotechnique`
`P = 4000;`
`Fp = 0.8;`
`Q = tan(acos(Fp))*P`
	`>> Q =`
	`        3000`

3 Calcul matriciel

3.1 Matrices

3.1.1 Création

Tous les types de variabled de MATLAB sont basés autour de la notion de matrice. Un scalaire est une matrice de taille 1×1, un vecteur est une matrice de taille $n \times 1$ ou $1 \times n$, etc. On définit une matrice par deux crochets [ ]. L'espace permet l'identification des différentes données dans la matrice dans la dimention horizontale. La virgule effectue le même travail. Le point virgule indique un changement de dimention verticale.

Instructions	Affichage

`A=[1 2]`
`B=[1,2]`
`C=[1;2]`
`D=[1 2;3 4]`
	`>> A =`
	`     1     2`
	`B =`
	`     1     2`
	`C =`
	`     1`
	`     2`
	`D =`
	`     1     2`
	`     3     4`

3.1.2 Opérateur deux points (:)

Un outil puissant de MATLAB est l'opérateur « :». Il permet de décomposer les nombres compris entre le premier et le dernier, avec un intervalle du nombre du milieu. Ainsi,

J :K	est le même que [J, J+1, ..., K]
J :K	est vide si J>K
J :D :K	est le même que [J, J+D, ..., J+m*D] où m = $\left\lfloor \frac{K-J}{D} \right\rfloor$
J :D :K	est vide si D>0 et J>K ou D<0 et J<K

Instructions	Affichage

`x=1:1:10`
`y=10:-1:1`
`z=0:.1:.5`
	`>> x =`
	`     1     2     3     4     5     6     7     8     9    10`
	`y =`
	`    10     9     8     7     6     5     4     3     2     1`
	`z =`
	`     0   0.1   0.2   0.3   0.4   0.5`

3.1.3 Indices d'une matrice

Pour modifier un seul élément de la matrice, on utilise les indices pour identifier chacun des éléments. Les indices sont décrits comme suit : `A(m,n)` où *m* est le numéro de la ligne et *n* le numéro de la colone. Ainsi, `A(1,2)` indique l'élément placé à la 1e ligne de la 2e colonne. En égalant un indice à un nombre, on modifie l'élément à l'indice.

Instructions	Affichage

`A=[1 2 3; 4 5 6; 7 8 9];`
`A(3,3)=A(2,1)+A(1,3)`
	`>> A =`
	`1 2 3`
	`4 5 6`
	`7 8 7`

Toujours sous le principe des indices, il est possible de modifier plusieurs éléments à la fois dans une matrice ou un vecteur grâce au « :» qui désignera une colonne ou une ligne complète, ou une partie de ligne ou de colonne.

Instructions	Affichage

`A(1:2,3)`
`A(2:3,2:3)`
`A(2,:)`
	`>> ans =`
	`3`
	`6`
	`ans =`
	`5 6`
	`8 7`
	`ans =`
	`4 5 6`

Si l'incrément des colonnes ou des lignes à traiter est différente de l'unité, on doit mettre les différents indices entre crochets ([ ]). L'ordre dans lequel on met les indices dans les crochets détermine l'ordre dans lesquels les éléments sortiront.

Instructions	Affichage

`A([1 3],:)`	`>> ans =`
	`1 2 3`
	`7 8 7`

3.2 Opérations matricielles

Dans le tableau suivant, la mention d'une opération «élément par élément» signifie que chaque élément réagira avec l'élément correspondant à la même position dans la seconde matrice. Par exemple pour l'addition, on obtiendra que $c(i,j) = a(i,j) + b(i,j)$.

Opération	**Opérateur**	**Commentaire**
Addition	a + b	Addition élément par élément.
Soustraction	a - b	Division élément par élément.
Multiplication matricielle	a * b	Multiplication matricielle standard où $c(i,j) = \sum_{k=1}^{n} a(i,k)b(k,j)$. (1)
Multiplication de tableau	a .* b	Multiplication élément par élément. (2)
Division matricielle de droite	a / b	Division matricielle de droite définie par a * inv(b) où inv(b) est l'inverse de b. (1)
Division de tableau de droite	a ./ b	Division de a par b élément par élément.(2)
Division de matricielle de gauche	a \ b	Division matricielle de gauche définie par inv(a) * b où inv(a) est l'inverse de a. (2)
Division de tableau de gauche	a .\ b	Division de b par a élément par élément. (1)
Puissance	a .∧ b	a exponentiel de b, élément par élément. (2)

(1)Le nombre de colonnes de la matrice a doit être égale au nombre de rangées de la matrice b.

(2)Les dimensions des deux matrices doivent être identiques.

<table>
<tr><th>Instructions</th><th>Affichage</th></tr>
<tr><td><pre>A=[1 2;3 4];
B=[3 3;6 6];
A+3
A+B
A-B</pre></td><td><pre>>> ans =
     4   5
     6   7
ans =
     4   5
     9   10
ans =
    -2   -1
    -3   -2</pre></td></tr>
<tr><td><pre>A*B
A.*B</pre></td><td><pre>>> ans =
    15   15
    33   33
ans =
     3   6
    18   24</pre></td></tr>
</table>

<table>
<tr><th>Instructions</th><th>Affichage</th></tr>
<tr><td><pre>A/B
A./B
A\B
A.\B</pre></td><td><pre>>> Warning: Matrix is singular to working precision.
ans =
   Inf   Inf
   Inf   Inf
ans =
      0.33333   0.66667
          0.5   0.66667
ans =
  2.9606e-016 2.9606e-016
          1.5   1.5
ans =
            3   1.5
            2   1.5</pre></td></tr>
<tr><td><pre>A.^B</pre></td><td><pre>>> ans =
           1   8
         729   4096</pre></td></tr>
</table>

Démonstration sur les opérations matricielles. Une fois le fichier-m ouvert, cliquez sur RUN. (fichier : opmat.m ou taper `opmat`)

3.3 Matrices spéciales

La fonction `zeros(m,n)` produit une matrice $m \times n$ où toutes les données sont 0. La fonction `ones(m,n)` produit une matrice $m \times n$ où toutes les données sont 1. La fonction `eye(n)` produit une matrice identité $n \times n$. Enfin, la fonction `rand(m,n)` produit une matrice $m \times n$ dont les données sont choisies aux hasards.

<table>
<tr><th>Instructions</th><th>Affichage</th></tr>
<tr><td><pre>Z=zeros(3,5)
Un=ones(2,2)
Huit=8*ones(2,2)
I=eye(3)
R=rand(2,4)</pre></td><td><pre>>> Z =
     0   0   0   0   0
     0   0   0   0   0
     0   0   0   0   0
Un =
     1   1
     1   1
Huit =
     8   8
     8   8
I =
     1   0   0
     0   1   0
     0   0   1
R =
      0.95013   0.60684   0.8913   0.45647
      0.23114   0.48598   0.7621   0.018504</pre></td></tr>
</table>

3.4 Fonctions matricielles

Dans tous les exemples suivant, ce sont les matrices 3×3 A=[1 2 3 ;4 5 6 ;7 8 9] et B=[1 3 3 ; 5 1 2 ; 6 6 6] qui seront employées.

3.4.1 Transposé

La transposé d'une matrice s'effectue avec la fonction "apostrophe", c'est à dire qu'on effectue la transposée de `A` en indiquant `A'`.

Instructions	Affichage

`A'`	`>> ans =`
	`1 4 7`
	`2 5 8`
	`3 6 9`

En fait, la transposé simplement avec l'apostrophe est une transposé couplée à un conjugué (d'un nombre complexe). Or puisque les matrices que nous utilisons jusqu'à maintenant sont des nombres réels, il n'y a pas vraiment de différence. Par contre, lorsque nous utilisons les nombres complexes, il est important d'effectuer la transposée désirée (avec ou sans conjugué). La transposé sans conjuguée d'inscrit en mettant un point devant l'apostrophe : si `B` est une matrice complexe, sa transposée sans conjuguée devient `A.'`.

Instructions	Affichage

`B = [3+8i 4-7i; -4 + 8i -1-1i];`
`avec = B'`
`sans = B.'`
	`>> avec =`
	
	`3.0000 - 8.0000i -4.0000 - 8.0000i`
	`4.0000 + 7.0000i -1.0000 + 1.0000i`
	
	`sans =`
	
	`3.0000 + 8.0000i -4.0000 + 8.0000i`
	`4.0000 - 7.0000i -1.0000 - 1.0000i`

3.4.2 Inverse

L'inverse d'une matrice s'obtient grâce à la fonction `inv(A)`. Les matrices singulières n'ont pas d'inverse. Observez l'exemple suivant pour voir le message d'erreur qui apparaît lorsque vous tentez d'inverser ce genre de matrice.

Instructions	Affichage

`inv(A)`	`>> Warning: Matrix is close to singular or badly scaled.`
	`Results may be inaccurate. RCOND = 1.541976e-018.`
	`ans =`
	`-4.5036e+015  9.0072e+015 -4.5036e+015`
	`9.0072e+015 -1.8014e+016  9.0072e+015`
	`-4.5036e+015  9.0072e+015 -4.5036e+015`

`inv(B)`	`ans =`
	`-1  0  0.5`
	`-3  -1  1.8333`
	`4  1  -2.1667`

3.4.3 Concaténation

La concaténation consiste à former une grande matrice à partir de petites matrices. Il suffit de placer des matrices à la place des données pour donner une plus grande matrice.

Instructions	Affichage

`A=ones(2,2);`
`B=[A A-1; A+3 A+7]`
	`>> B =`
	`1  1  0  0`
	`1  1  0  0`
	`4  4  8  8`
	`4  4  8  8`

3.4.4 Déterminant

La fonction `det(A)` permet de trouver le déterminant de la matrice A.

Instructions	Affichage
`det(A)`	`>> ans =` `0`

3.4.5 Somme

La fonction `sum(A)` permet la somme des colonnes de la matrice A. En utilisant la transposé (voir section 3.4.1), on peut calculer la somme des rangées grâce à l'équation `sum(A')'` .

Instructions	Affichage

`SommeColonnes = sum(A)`
`SommeRangees = sum(A')'`
	`>> SommeColonnes =`
	`12 15 18`
	`SommeRangees =`
	`6`
	`15`
	`24`

Pour faire la somme d'une seule colonne, on peut utiliser l'opérateur « :» comme dans l'exemple suivant :

3.4.6 Réduction

Il est possible de réduire une matrice avec la fonction `rref(A)`. À l'aide de cette opération, il est facile de voir si la matrice est singulière (si une de ses lignes est composée de 0).

Instructions	Affichage

`rref(A)`
`rref(B)`
	`ans =`
	`1 0 -1`
	`0 1 2`
	`0 0 0`
	`ans =`
	`1 0 0`
	`0 1 0`
	`0 0 1`

Instructions	Affichage
`sum(A(:,3))` `%somme de la 3e colonne`	`>> ans =` `18`

3.4.7 Rang

Le rang d'une matrice s'obtient avec la fonction suivante : `rank(A)`. Cette fonction produit une évaluation du nombre de colonnes et de lignes linéairement indépendantes.

Instructions	Affichage
`rank(A)`	`>> ans =` `2`

3.4.8 Norme

La norme d'une matrice ou d'un vecteur est définie dans Matlab par la fonction `norm(A)`.

Instructions	Affichage
`norm(A)`	`>> ans =` `16.848`

3.4.9 Diagonalisation

La fonction nécessaire pour effectuer la diagonalisation d'une matrice est `diag(A)`

Instructions	Affichage

`diag(A)`	`>> ans =`
	`1`
	`5`
	`9`

3.4.10 Ajout et retrait de colonnes et rangées

Pour ajouter des colonnes ou des rangées, on peut procéder de 2 manières : on peut simplement ajouter chacun des éléments à la matrice un à fois. On peut aussi effectuer la concaténation de la matrice initiale avec un vecteur colonne ou rangée à ajouter.

Instructions	Affichage

`A=[1 2;3 4]`
`A(1,3)=5`
`A(2,3)=6`
	`>> A =`
	`1 2`
	`3 4`
	`A =`
	`1 2 5`
	`3 4 0`
	`A =`
	`1 2 5`
	`3 4 6`

`A=[1 2;3 4];`
`B=[5;6]`
`C=[A B]`
`A(:,3)=B`
 	
`% les deux dernières instructions sont deux méthodes équivalentes`
	`>> B =`
	`5`
	`6`
	`C =`
	`1 2 5`
	`3 4 6`
	`A =`
	`1 2 5`
	`3 4 6`

Pour le retrait, on utilise la concaténation pour introduire une matrice vide qui aura pour effet d'enlever l'élément (colonne ou rangée) désiré.

Instructions	Affichage

A=[1 2 3; 4 5 6]
A(:,3)=[]
	`>> A =`
	`    1     2     3`
	`    4     5     6`
	`A =`
	`    1     2`
	`    4     5`

Démonstration sur les fonctions matricielles. Une fois le fichier-m ouvert, cliquez sur RUN. (fichier : fct-mat.m ou taper `fctmat`)

3.5 Exercices

1. Créez une matrice 3 par 5 possédant des valeurs au choix.
2. On désire produire une série de nombres entre 4 et 6 en sautant de .5 chacun des nombres.
3. Remplacez la série créée au no. 2 dans la deuxième rangée de la matrice du no.1
4. Soit les matrices suivantes : A=[1 1 ; 2 2] et B=[1 0 ; 3 0]. Effectuez toutes les opérations matricielles décrites et observez les différences.
5. Voici quelques matrices. À l'aide des fonctions décrites dans les matrices spéciales, reproduisez les. Vous avez le droit à la multiplication par un scalaire, mais pas d'entrer les données simplement !

```
- X =      5     3     3     3
           3     5     3     3
           3     3     5     3
           3     3     3     5
- Y =    2.48598
         2.8913
         2.7621
- Z =    2     1
         1     2
```

6. A = [3 5 9 ;4 2 1 ; 13 3 8 ; 2 8 1] ; À partir de la matrice 4 par 3 précédente, créer une matrice 5 par 4 qui contiendra la matrice intiale en haut à gauche, la somme des colonnes sous ces colonnes, ie la dernière rangée, la somme des rangées à droite des rangées, ie à la dernière colonne. La matrice est singulière ou non. Si possible, trouver l'inverse, la norme, le rang et le déterminant de la matrice. Expliquez pourquoi il y a impossibilité, s'il y a lieu.

   Voici un diagramme illustrant la nouvelle matrice. On note où se place la matrice initiale par un A, la somme des colonnes par un C la somme des rangées par un R et la somme de la diagonalisation de la matrice intiale par un D.

```
|A A A R|
|A A A R|
|A A A R|
|A A A R|
|C C C D|
```

ExerciceMatrices.m

4 Courbes et figures

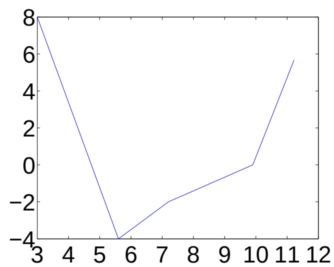
À la différence des autres programmes permettant de dessiner des graphiques, Matlab est spécialisé dans les graphiques tracés à partir de matrices de données. Puisque les graphiques en 3D ne sont pas très utiles lors du cours d'ELE1400, ils ne seront pas traités dans ce didacticiel. Pour plus de détails, consultez l'aide de Matlab.

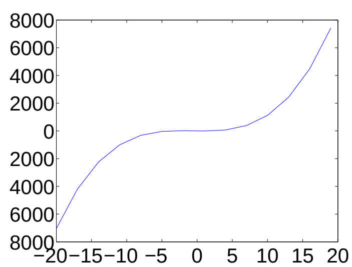
4.1 Tracer des courbes

Pour tracer des courbes, il existe deux fonctions principales : `plot` et `fplot`.

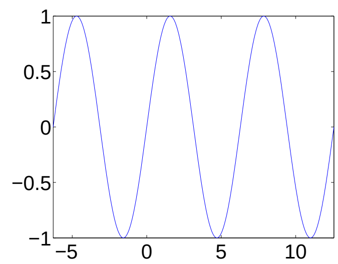
La fonction `plot(X,Y)` trace la courbe dont les valeurs de la variable indépendante sont indiqués dans `X` et les valeurs de la variable dépendante dans `Y`. Notons que les valeurs peuvent être décrites par une fonction. C'est avec cette fonction qu'il faut tracer des courbes lorsque nous avons des données expérimentales. Il suffit de les mettre dans une matrice.

Instructions	Affichage

`% données expérimentales`
 	
`X=[3 5.6 7.2 9.9 11.22];`
`Y=[8 -4 -2 0 5.67];`
`plot(X,Y)`
	

`% fonction`
 	
`X1=-20:3:20;`
`Y1=X1.^3+2*X1.^2-8*X1+1;`
`plot(X1,Y1)`
	

Quant à ;a fonction `fplot`, elle s'écrit de la manière suivante : `fplot(fon,lim)` où `fon` indique le nom de la fonction à tracer sous la forme d'une chaîne de caractères (entre guillemets) et `lim` défini les limites des échelles. Pour les limites, il faut les écrire entre crochet avec la syntaxe suivante : `lim = [Xmin Xmax Ymin Ymax]`.

Instructions	**Affichage**
`f='sin';` `fplot(f,[-2*pi 4*pi])`	

Lorsque vous avez plusieurs graphiques à tracer dans un même fichier, pour que tous s'affichent en même temps, il faut indiquer l'instruction `figure` avant chaque graphique à tracer afin de créer une nouvelle figure pour chacun d'eux.

Instructions

`x = 0 : .05 : 3*pi;    y1 = sin(x);    y2 = cos(x);`
 	
`figure; plot(x,y1);`
`figure; plot(x,y2);`

4.2 Identification des axes et des points, légendes, grilles

Il est possible d'identifier les axes par les fonctions suivantes : `xlabel(texte)`, `ylabel(texte)`. On peut titre le graphique grâce à `title(texte)` et ajouter une légende avec `legend(fonction1,fonction2,...,pos)` où `fonctionX` est le texte à indiquer sur la fonction désirée et où `pos` est la position où placer la légende.

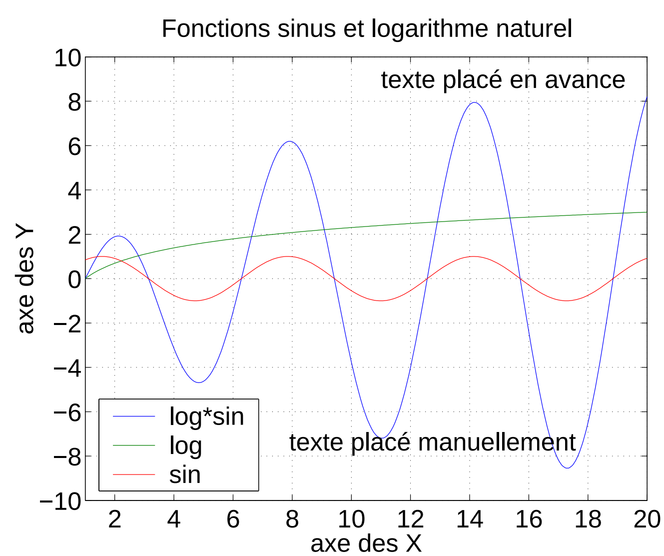
POSITION

0 = Meilleur placement automatique (où il y a le moins de conflits avec les données)
1 = Coin supérieur à droite (défaut)
2 = Coin supérieur à gauche
3 = Coin inférieur à gauche
4 = Coin inférieur à droite
-1 = À droite du graphique

Il est aussi possible de quadriller le graphique avec la fonction `grid`. Cette fonction s'applique à tous les types d'échelles. Il est aussi possible d'ajouter du texte sur le graphique de 2 manières : la fonction `text(x,y, 'text')` où `x` et `y` sont les coordonnées où placer le texte et la fonction `gtext('text')` où le texte sera placé manuellement avec la sourie lors de l'exécution du graphique.

De plus, on peut modifier les axes à l'aide des fonctions suivantes :

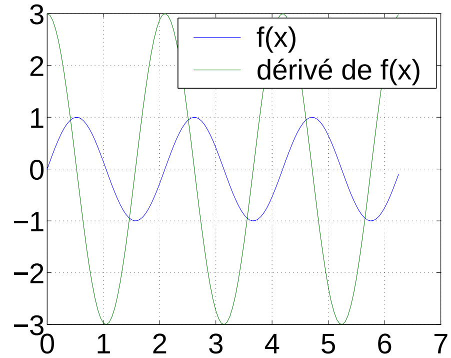
`Axis([xmin xmax ymin ymax])`	Déterminer les valeurs maximales et minimales des axes
`Axis equal`	Rend l'incrémentation des axes identiques
`Axis square`	Donne la même incrémentation ainsi que les mêmes valeurs minimales et maximales aux axes
`Axis off`	Annule l'identification des axes, le fond du graphique ainsi que les marques

<table>
<tr><td>Instructions</td><td><pre>figure
fplot('[log(x)*sin(x)*3 log(x) sin(x)]',[1 20 -10 10])
title('Fonctions sinus et logarithme naturel')
xlabel('axe des X')
ylabel('axe des Y')
gtext('texte placé manuellement')
text(11,9,'texte placé en avance')
legend('log*sin','log','sin',0)
grid</pre></td></tr>
<tr><td>Affichage</td><td>
</td></tr>
</table>

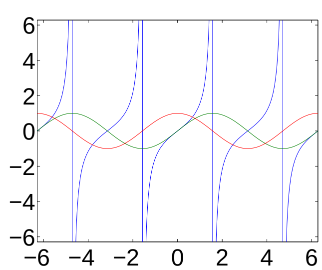
4.3 Tracés multiples superposés et côte à côte

Pour obtenir plusieurs tracés superposés, il suffit de les inscrire dans la fonction `plot(x1,y1,x2,y2, ...)`. On peut aussi utiliser la fonction `hold on` pour superposer des graphiques.

Instructions	Affichage

Les deux séries d'instructions suivantes produisent le même graphique :
 	
`x=0:.05:2*pi;`
`y1=sin(3*x);`
`y2=3*cos(3*x);`
 	
1. En utilisant la fonction `plot(x1,y1,x2,y2,...)`
 	
`plot(x,y1,x,y2)`
`legend('f(x)','dérivé de f(x)')`
`grid on`
 	
2. En utilisant la fonction `hold`
 	
`plot(x,y1)`
`grid on`
`hold on`
`plot(x,y2)`
`legend('f(x)','dérivé de f(x)')`
`hold off`
	

Pour les fonctions superposées tracées à l'aide de la fonction `fplot`, il s'agit de créer une matrice avec les noms des fonctions désirées.

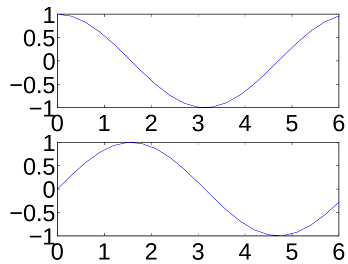
Instructions	Affichage
`f='[tan(x),sin(x),cos(x)]';` `fplot(f,2*pi*[-1 1 -1 1])`	

Pour obtenir plusieurs graphiques côte à côte, il suffit d'utiliser la fonction `subplot(m,n,p)`, ce qui divise la figure en une matrice de *m* par *n* carreaux rectangulaires. `P` indique la position dans la matrice que prendra la fonction `plot` qui suit dans les instructions.

Si on désire créer une matrice 2x2 pour les graphiques, la fonction `subplot` à inscrire avant chaque fonction à afficher se défini ainsi :

1 `subplot(2,2,1)`	2 `subplot(2,2,2)`
3 `subplot(2,2,3)`	4 `subplot(2,2,4)`

Instructions	**Affichage**

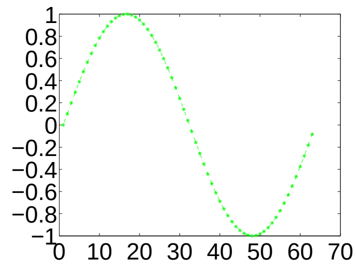
`t = 0:0.3:2*pi;`
`x = cos(t);`
`y = sin(t);`
 	
`figure`
`subplot(2,1,1);`
`plot(t,x)`
`subplot(2,1,2);`
`plot(t,y)`
	

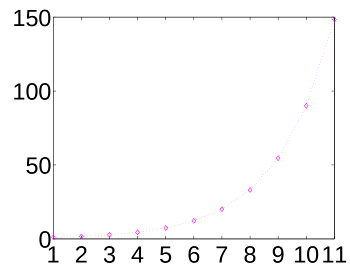
4.4 Options visuelles

Pour tracer les courbes, plusieurs options sont offertes. L'usager peut fait le choix entre le type de traits et de point utilisé, ainsi que la couleur. On utilise la fonction `plot(X,Y,S)` où `S` indique le choix d'option entre guillemets.

COULEUR	POINT	TRAIT
b bleu	. point	- continu
g vert	o cercle	: pointillé
r rouge	x x	-. alterné
c cyan	+ plus	– interrompu
m magenta	* étoile	
y jaune	s carré	
k noir	d diamant	
	$\vee$ triangle (bas)	
	$\wedge$ triangle (haut)	
	< triangle (gauche)	
	> triangle (droite)	
	p pentagram	
	h hexagram	

Instructions	**Affichage**

`figure;`
`X1 = 0:.1:2*pi;`
`plot(sin(X1),'g*-.');`
	

`X2 = 0:.5:5;` `plot(exp(X2),'md:');`	

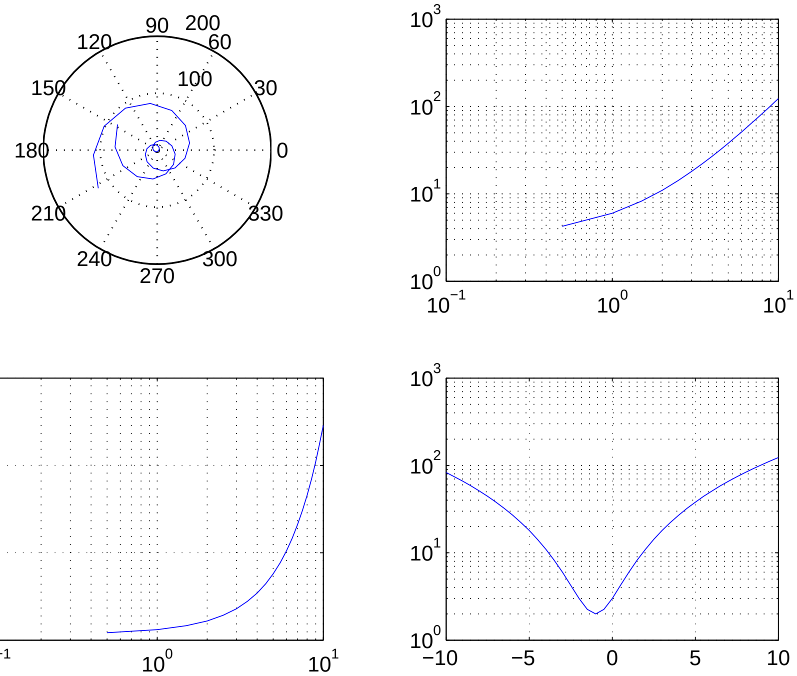
Démonstration sur les options des graphiques. Une fois le fichier-m ouvert, cliquez sur RUN. (Fichier : hndlgraf.m ou taper `hndlgraf` ou taper `demo`, aller dans Matlab - Graphics et choisir Line Plotting)

4.5 Échelles

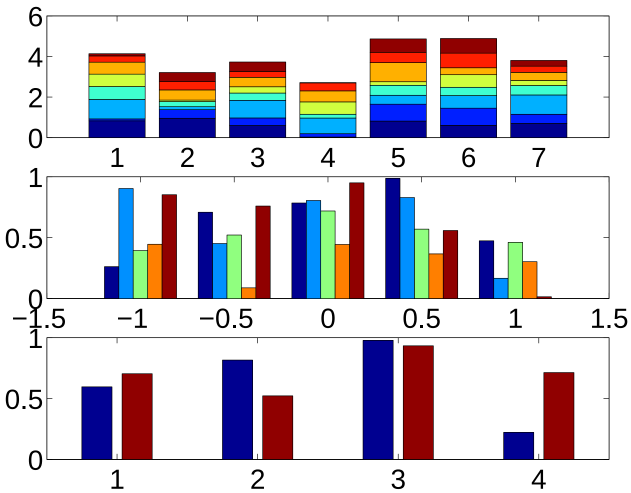
Plusieurs fonctions pour tracer des graphiques sont disponibles pour les différentes échelles.

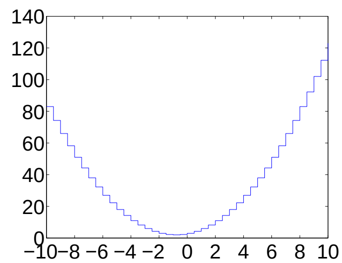
`plot(x,y)`	linéaire x-y	`polar(theta,r)`	polaire
`loglog(x,y)`	logarithmique x-y	`bar(x,y)`	barres
`semilogx(x,y)`	logarithmique x, linéraire y	`stairs(x,y)`	escalier
`semilogy(x,y)`	linéaire x, logarithmique y		

Instructions	`x=-10:.5:10;`
	`y=x.^2+x.*2+3;`
	
	`subplot(2,2,1), polar(x,y), grid on`
	`subplot(2,2,2), loglog(x,y),grid on`
	`subplot(2,2,3), semilogx(x,y), grid on`
	`subplot(2,2,4), semilogy(x,y), grid on`

Affichage	

Pour tracer des graphiques à barres, on utilise la fonction `bar(x,y)`. Elle trace, selon la matrice m par n où m représente la quantité de groupes de barres, et n le nombre de barres par groupe. De plus, on peut utiliser la fonction `bar(...,'stacked')` pour empiler les différentes barres du groupe. On peut aussi définir la largeur des bandes avec `bar(...,largeur)`. Quant aux escaliers, on utilise la foncton `stairs(x,y)` où `x` indique la largeur des marches et `y` la position selon l'axe des ordonnées.

<table>
<tr><td>Instructions</td><td><code>subplot(3,1,1), bar(rand(7,8),'stacked')</code>
<code>subplot(3,1,2), bar(-1:.5:1,rand(5),1)</code>
<code>subplot(3,1,3), bar(rand(4,2),.75)</code></td></tr>
<tr><td>Affichage</td><td></td></tr>
</table>

<table>
<tr><td>Instructions</td><td><code>x=-10:.5:10;</code>
<code>y=x.^2+x.*2+3;</code>
<code>stairs(x,y)</code></td></tr>
<tr><td>Affichage</td><td></td></tr>
</table>

Démonstration sur les opérations matricielles. Une fois le fichier-m ouvert, cliquez sur RUN. (Fichier : echelles.m ; ou taper `echelles`)

4.6 Transfert des figures dans un fichier doc.

L'instruction `print` sans option envoie la figure à l'imprimante par défaut sur l'ordinateur.

La fonction d'impression comprend une grande quantité d'options. Pour toutes les voir, entrer l'instruction `help print`. Voici les instructions les plus utiles.

Option	Explication	Exemple
<nomdufichier>	Sauve la figure sous le nom défini dans le répertoire courant (Current directory). Le format de sauvegarde par défaut est Postscript. Il est possible de rajouter une extension au nom du fichier qui déterminera le format dans lequel il sera sauvé.	print exp3
-f<nomdelafigure>	Indique le nom de la figure à traiter (imprimer si aucun autre option spécifiée)	print -f2
-dbitmap	Copie la figure ouverte dans le presse-papiers en format bitmap. Le format bitmap produit une copie-photo du graphique en question.	print -dbitmap
-dsetup	Appelle la fenêtre d'impression sans lancer l'impression	print -dsetup
-dmeta	Copie la figure ouverte dans le presse-papiers en format metafile. Le format metafile produit une copie du graphique, qui, une fois collée dans Word (voir procédure suivant le tableau), pourra être modifé (diposition, texte, etc).	print -dmeta
-djpeg<nn>	Sauve dans le format jpeq avec une qualité nn.	print -djpeg90

Pour copier un graphique metafile dans Word, il faut aller dans le menu Édition → Collage Spécial.... Une fenêtre apparaitera. Si vous avez dans votre presse-papier une image métafile, il suffira de cliquer sur Image(métafichier amélioré) puis sur OK pour que le fichier apparaisse. Une fois l'image dans le fichier Word, il faut cliquer-droit sur cette image et choisir Modifier l'image pour la modifier.

Pour utiliser plus d'une option à la fois, on procède selon cet ordre :
`print -outil -options nomdufichier`
La liste des outils est disponible dans l'aide.

Instructions

```
print -dbitmap -f3
% copie dans le presse-papiers de la figure 3
```

4.7 Éditeur de figures

Lorsqu'on traite des données, il est parfois fastidieux d'entrer toutes les commandes pour modifier les options du graphique. Parfois, il faut aussi en essayer plusieurs afin de déterminer laquelle produit un meilleur graphique. C'est pourquoi toutes les options abordées dans cette section possèdent une seconde manière d'être mises sur le graphique : on peut les indiquer par le biais des menus ainsi que des boutons sur la figure.

Ainsi, on peut les modifier et observer le résultat à notre guise sans avoir à créer une nouvelle figure à chaque fois.

Il suffit d'enregistrer la figure à la fin afin de conserver ces options. Par défaut, l'enregistrement de la figure se fait en format `.fig`, le format des figures de Matlab, mais il est possible de le sauver dans un autre format tel `bitmap (.bmp)`.

Il est aussi possible de copier la figure une fois qu'elle est ouverte sans avoir à passer par les instructions. Il suffit d'aller dans le menu Edition → Copy figure. Pour choisir dans quel format la figure sera enregistrée, allez dans le menu Edition → Copy Options. Dans cette fenêtre de préférences, sélectionner le format désiré. Voir le tableau de la section précédente pour connaître les propriétés de chacun des formats et pour connaître comment coller cette copie dans Word.

À partir des boutons, on peut créer une nouvelle figure, ouvrir une figure, enregistrer la figure actuelle, ou l'imprimer.

Il est aussi possible de modifier le graphique grâce à l'icone de la flèche. Une fois ce bouton enfoncé, il suffit de double-cliquer sur un item du graphique pour que la fenêtre de l'éditeur des propriétés (Property Editor) s'ouvre. Grâce à cet éditor, il est possible de modifier l'objet en question grâce aux différentes options qui nous sont offertes. Voici quelques exemples des propriétés intéressantes d'objets qui nous sont possible de modifier par le biais de cet éditeur.

Axes	– Limites – Échelles – Grille – Style (Bordure, Couleur, Police, ...) – Titre du graphique – Identification des axes
Lignes	– Données – Style de ligne et de point – Nom de la ligne (Tag)
Texte	– Police – Texte – Position, Allignement – Nom du texte (Tag)
Légende	– Police – Texte – Position, Style – Nom de la légende (Tag)
Figure	– Couleur de fond – Présence ou non du menu et des boutons – Titre de la figure – Nom de la figure (Tag)

On peut ajouter des éléments sur le graphique sans passer par l'éditeur des propriétés. Le menu Insert offre la possibilité d'ajouter un élément (Titre, Identification des axes, Légende, Flèche, texte, etc.). Il y a 3 boutons sur la barre qui permettent l'ajout de lignes, flèches et de textes.

Si vous sélectionnez un objet, vous pouvez aussi cliquer-droit pour avoir accès à certaines de ses propriétés.

Dans n'importe quelle fenêtre de Matlab possédant un menu (Fenêtre principale, Éditeur/débogueur, Figure), il est possible d'accéder aux préférences de Matlab par le biais de File → Preferences. Dans cette fenêtre, il est possible de modifier des éléments égalements. Voici quelques options pouvant être utiles. Pour le reste, explorez par vous-même toutes les options offertes dans Matlab.

Command Window	Text Display	: Numeric Format Numeric Display
Figure Copy Template	Text	: Change font size
FCT → Copy Options	Clipboard Format :	

4.8 Approximation de données

4.8.1 Fonction polyfit

Une fonction très utile dans les laboratoires est la fonction `polyfit(x,y,n)` qui donne un polynôme qui approxime les données. Dans l'équation, on donne les donnes respectives dans le `x` et le `y`, puis on indique le degré du polynôme qui approximativera les données dans le `n`.

`f = polyfit(x,y,n)` retourne la variable `f` qui est un vecteur rangées de n+1 dimention contenant les coefficients des puissances descendantes. Ainsi, la fonction p(x) produite sera sous la forme :

$p(x) = p_1x^n + p_2x^{n-1} + \ldots + p_nx^1 + p_{n+1}$

où p_1 est contenu dans `f(1)`, p_2 dans `f(2)`,...

Ainsi, en appelant les instructions suivantes, on obtiendras

`f = polyfit(x,y,1)` $\Rightarrow p(x) = f(1) * x + f(2)$
`f(1)` est la pente de la droite et `f(2)` est la valeur à l'origine
`f = polyfit(x,y,2)` $\Rightarrow p(x) = f(1) * x^2 + f(2) * x + f(3)$
`f(1)` est le coefficient a de l'équation quadratique, `f(2)` est b, `f(3)` est c.

Pour tracer la courbe par la suite, il suffit d'utiliser les coefficients du polynôme pour le recréer et le tracer à l'aide de `plot` ou de `fplot`.

4.8.2 Basic Fitting

Si on connais préalablement le degré du polynôme que sont supposés former les données, cette méthode est excellente. Par contre, si on ne connais pas exactement le degré à employé, une seconde manière d'approximer sur le graphique les données s'impose.

Cette manière consiste à tracer les données désirées. Une fois la figure produite, allez dans le menu Tools → Basic Fitting. Par la fenêtre appaissant, il est possible de sélectionner le groupe de données désirées (si par exemple il y en a plusieurs sur un même grapique), de sélectionner la ou les approximations désirées selon le degré du polynome. De plus, il est possible de montrer l'équation de chaque approximation. On peut aussi indiquer sur un graphique la différence entre chaque donnée et sa valeur calculée avec l'approximation.

En cliquant sur la flèche → en bas en gauche, la fenêtre s'étant et les résultats numériques sont disponibles. Dans ces résultats, on trouve la fonction théorique du polynôme, ses coefficients, ainsi que la norme des différence entre chaque donnée et sa valeur calculée avec l'approximation. De plus, on peut enregistrer ces données dans des variables disponibles dans le Workspace.

En étendant la fenêtre une seconde fois avec la flèche, on peut calculer f(x) d'une valeur ou d'un groupe de valeurs selon l'équation sélectionnée dans le "menu du centre".

Enfin, si cette méthode a été employé et que vous désirez conserver le graphique, n'oubliez pas de l'enregistrer afin de conserver tous les changements que vous venez d'apporter.

4.9 Exercices

1. À partir des données suivantes, produisez un graphique comprenant les données expérimentales, la courbe approximant ces données et analysez le résultat produit.

```
%           NbL.    E       I
%           ----------------------
Donnees=[   0       0       1.57
            50      6.7     1.54
            45      7.28    1.52
            40      8.14    1.50
            35      8.88    1.505
            30      10.02   1.502
            25      11.40   1.48
            20      14.1    1.490
            15      17.32   1.468
            10      22.9    1.430
            5       40.4    1.33  ] ;
```

   SourceCourant.m

2. À partir des données suivantes, produisez un graphique comprenant les données expérimentales, la courbe approximant ces données et analysez le résultat produit.

```
%           NbL.    E       I
%           ----------------------
Donnees = [ 1       126     1.11
            2       125.8   2.05
            5       126     5.4
            7       124     7.45
            10      124.5   10.22
            12      126     12.3
            15      127     15.74
            18      124.5   18.9    ] ;
```

   SourceTension.m

5 Calcul complexe

5.1 Fonction complexe

Les mathématiciens présentent les nombres complexes sous la forme de $A = a + b * i$, alors que les électriciens le font sous la forme de $A = a + b * j$. Matlab ne fait pas la différence entre i et j. On peut aussi utiliser la fonction `sqrt(-1)` à la place de i ou de j. Par contre, tout nombre mis sous une forme complexe (voir section 5.3 pour les différentes formes) sera redonné sous la forme cartésienne $a + b * i$. Pour obtenir des formes différentes il faut utiliser les fonctions spécifiques décrites ci-dessous. La fonction `complex(a,b)` peut être utilisée.

Instructions	Affichage

`a = 2 + 4i`
`b = 2 + 4j`
`c = 2 + 4*sqrt(-1)`
`d = complex(2,4)`
	`>> a =`
	`2 + 4i`
	`b =`
	`2 + 4i`
	`c =`
	`2 + 4i`
	`d =`
	`2 + 4i`

5.2 Opérations de base

Les opérations de bases telles l'addition, la soustraction, la multiplication ou la division sur des nombres complexes sont effectuées aussi simplement que si la variable était réelle. Il n'y a aucune transformation de forme nécessaire.

Les différentes opérations de base pouvant être effectuées sur un nombre complexe sont les suivantes :

`real(x)`	Partie réelle
`imag(x)`	Partie imaginaire
`conj(x)`	Conjugué
`abs(x)`	Norme
`angle(x)`	Angle

L'angle produit est en radian. Pour l'utiliser en degré ou pour le transformer en degré, utiliser la régle de proportion suivante : 180°= π rad.

La fonction `unwrap(x)` ajoute ou soustrait des multiples de 2π afin que la valeur de l'angle soir situé entre $-\pi$ et π.

Instructions	Affichage

`nombre = 8+4j`
`reel = real(nombre)`
`imaginaire = imag(nombre)`
`conjugue = conj(nombre)`
`norme = abs(nombre)`
`angle = angle(nombre)`
`angle2 = unwrap(7.0686)`
	`>> nombre =`
	`8 + 4i`
	`reel =`
	`8`
	`imaginaire =`
	`4`
	`conjugue =`
	`8 - 4i`
	`norme =`
	`8.9443`
	`angle =`
	`0.46365`
	`angle2 =`
	`7.0686`

5.3 Transformation entre les différentes formes (cartésienne, polaire, trigonométrique)

Forme cartésienne : $A = a + bj$
Forme polaire : $A = D\angle\theta = D * exp(angle * j)$
(cette forme vient de la forme exponentielle)
Forme trigonométrique : $A = D * \cos(angle) + j * D * \sin(angle)$

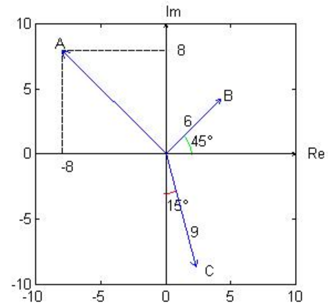
Attention, l'angle utilisé doit être en radian.

Instructions	Affichage

`A = 28+9j`
`NormeA = abs(A)`
`AngleA = ANGLE(A)`
`AngleAdegre = AngleA*180/pi`
 	
`B = NormeA * exp(AngleA*j)`
`C = NormeA * (cos(AngleA) + i*sin(AngleA))`
	`>> A =`
	`28 + 9i`
	`NormeA =`
	`29.411`
	`AngleA =`
	`0.311`
	`AngleAdegre =`
	`17.819`
	`B =`
	`28 + 9i`
	`C =`
	`28 + 9i`

5.4 Exercices

1. Soit A = 4 + 8i et B = - 2 - 7i. Calculez A+B, A-B, A*B, A/B.
2. Les trois vecteurs présentés dans la figure Exercice4.fig sont des nombres complexes. Écrivez ces nombres sous la forme polaire, cartésienne et trigonométrique.



3. Tracez les nombres complexes suivants sur un meme graphique : A = 4+8i, B=-2-7i, C : norme de 8, angle de 125˚, D= 6-j6.
4. Soit les nombres complexes suivants : S=7-3i, F=5e∧(πi/3) et K=8angle(90˚). Calculez F*K, F/S, F+K, S-K. Exprimez vos réponses sous forme polaire et cartésienne. Représentez à l'aide de vecteurs, sur un meme diagramme, les nombres S, F et K.
5. Écrivez un bout de programme permettant de tracer sous la forme temporelle et vectorielle une fonction sous la forme A(t) = A *cos(omega*t+phi). Par exemple, prenez la fonction A(t) = 8*cos(40t+199)

ExerciceComplexes.m

Exercice4.fig.

6 Liens vers des fichiers-m

6.1 Exercices

Calcul matriciel	**Exercice** (fichier : ExerciceMatrices.m)
Courbes et figures	**Exercice sur la source de courant** (fichier : SourceCourant.m) **Exercice sur la source de tensions** (fichier : SourceTension.m)
Calcul complexe	**Exercice** (fichier : ExerciceComplexes.m) **Figure associé à l'exercice 4** (fichier : Exercice4.fig)

6.2 Démonstrations

Opération matricielles	**Démonstration** (fichier : opmat.m) Tapez `opmat`
Fonctions matricielles	**Démonstration** (fichier : fctmat.m) Tapez `fctmat`
Options de graphique	**Démonstration** (fichier : hndlgraf.m) Tapez `hndlgraf` Tapez `demo`, aller dans Matlab - Graphics, et choisir Line Plotting
Échelles de graphique	**Démonstration** (fichier : echelles .m) Tapez `echelles`

7 Catalogue de commandes

Nom	Fonction	Description ou Commentaire
Fonctions générales		
Aide	help	Aide de Matlab
Démonstrations	demo	Liste des démonstrations disponible dans Matlab quant à son fonctionnement
Variables	who whos	Nomme les variables ainsi que donne la grandeur de l'espace mémoire occupé
Recherche	lookfor	Trouver toutes les fonctions contenant le mot recherché
Dimensions matrices	size	Dimensions de la matrice
Clear	clear	Efface l'espace de travail courant
Quitter	quit, exit	Met fin à Matlab
Calculs simples		
Addition	-	Addition
Soustraction	+	Soustraction
Multiplication	*	Multiplication
Division	/	Division
Puissances	$\wedge$	Puissance
Puissances de e	exp(x)	Puissance de e
Logarithme naturel	log(x)	Logarithme naturel (ln)
Logarithme en base 2	log2(x)	Logarithme en base 2
Logarithme en base 10	log10(x)	Logarithme en base 10 (commun)
Racine carrée	sqrt(x)	Racine carrée
Racines	roots(x)	Produit une matrice contenant les racines de x
Sinus	sin(x)	Sinus
Inverse du sinus	asin(x)	Inverse du sinus
Sécante	sec(x)	Sécante
Sinus hyperbolique	sinh(x)	Sinus hyperbolique
Inverse du sinus hyperbolique	asinh(x)	Inverse du sinus hyperbolique
Cosinus	cos(x)	Cosinus
Inverse du cosinus	acos(x)	Inverse du cosinus
Cosécante	csc(x)	Cosécante
cosinus hyperbolique	cosh(x)	cosinus hyperbolique
Inverse du cosinus hyperbolique	acosh(x)	Inverse du cosinus hyperbolique
Tangente	tan(x)	Tangente
Cotangente	cot(x)	Cotangente
Tangente hyperbolique	tanh(x)	Tangente hyperbolique

Nom	Fonction	Description ou Commentaire
Inverse de tangente	atan(x)	Inverse de tangente
Inverse de tangente	atan2(x)	Inverse de tangente dans le 4e quadrant (-$pi \leq$ ATAN2(Y,X) $\leq <=$ pi)
π	pi	le nombre pi
Infinit	inf	Infinit
Matrices		
Addition	a + b	Addition élément par élément
Soustraction	a - b	Division élément par élément
Multiplication matricielle	a * b	Multiplication matricielle standard où $c(i,j) = \sum_{k=1}^{n} a(i,k)b(k,j)$
Multiplication de tableau	a .* b	Multiplication élément par élément
Division matricielle de droite	a / b	Division matricielle de droite définie par a * inv(b) où inv(b) est l'inverse de b
Division de tableau de droite	a ./ b	Division de a par b élément par élément
Division de matricielle de gauche	a \ b	Division matricielle de gauche définie par inv(a) * b où inv(a) est l'inverse de a
Division de tableau de gauche	a .\ b	Division de b par a élément par élément
Puissance	a .∧ b	a exponentiel de b, élément par élément
Identité	eye(m)	Produit une matrice identitée $m \times m$
Zeros	zeros(m,n)	Produit une matrice $m \times n$ où toutes les données sont 0
Un	ones(m,n)	Produit une matrice $m \times n$ où toutes les données sont 1
Hasard	rand(m,n)	Produit une matrice $m \times n$ où toutes les données sont choisies au hasard
Déterminant	det(A)	Donne le déterminant de la matrice
Transposé	A'	Transposé la matrice
Réduction	rref(A)	Réduit la matrice par la méthode de Gauss-Jodan (Attention aux matrices singulières)
Inverse	int(A)	Inverse de la matrice A (Attention aux matrices singulières)
Somme	sum(A)	Somme des colonnes de la matrice A
Rang	rank(A)	Évalue le nombre de rangée et de colonnes linéairement indépendantes
Norme	norm(A)	Produit la norme de la matrice
Diagonalisation	diag(A)	Diagonalise la matrice
Matrice triagulaire inférieure	tril(A)	Permet d'obtenir la partie triangulaire inférieure d'une matrice
Matrice triagulaire supérieure	triu(A)	Permet d'obtenir la partie triangulaire supérieure d'une matrice
Logarithme d'une matrice	logm(A)	Effectue le logarithme de la matrice A
Exponentiel d'une matrice	expm(A)	Met la matrice comme puissance de la constante e

Nom	Fonction	Description ou Commentaire
Racine carrée d'une matrice	sqrtm(A)	Effectue la racine carrée de la matrice
Fonction d'une matrice	funm(A, fonction)	Effectue la fonction de la matrice
Valeurs propres	eig(A)	Renvoie les valeurs propres (eigenvalues) de la matrice carrée A
Coefficient polynome caractéristique	poly(A)	Renvoie les coefficients du polynôme caractéristique associé à la matrice A
	Graphiques	
Tracé d'une courbe	plot(x,y)	Permet de tracer une courbe selon la matrice X par Y
Tracé d'une courbe	fplot(fon, lim)	Permet de tracer une courbe selon la fonction fon avec les limites d'axes lim
Identification axe X	xlabel(texte)	Produit une étiquette de l'axe des X
Identification axe Y	ylabel(texte)	Produit une étiquette de l'axe des Y
Titre	title(texte)	Produit un titre au graphique
Légende	legend (fon1,fon2,pos)	Met une légende où fon1, fon2, . . .représentes les fonctions et pos la position où la légende se place dans le graphique
Grille	grid	Quadrille le graphique selon l'échelle choisie
Texte positionné d'avance	text(x,y,'mots')	Positionne en x et y le texte 'mots'
Texte positionné manuellement	gtext('mots')	Positionne le texte 'mots' manuellement lors de l'exécution du graphique
Axes	axis	Plusieurs options d'axes sont offertes
Tracés superposés	hold on ; hold off	Permet de superposer les graphiques avec hold on, puis arrête la superposition avec hold off
Tracés côte à côte	subplot(m,n,p)	Place sur une même figure plusieurs graphiques
Échelle logarithmique	loglog(x,y)	Trace un graphique avec une échelle logarithmique
Échelle semi-logarithmique en X	semilogx(x,y)	Trace un graphique avec une échelle semi-logarithmique en x
Échelle semi-logarithmique en Y	semilogy(x,y)	Trace un graphique avec une échelle semi-logarithmique en y
Échelle polaire	polar(theta,r)	Trace un graphique avec une échelle polaire
Barres	bar(x,y)	Trace un graphique avec des barres
Escalier	stairs(x,y)	Trace un graphique en escalier
	Nombres complexes	
Nombre complexe	a+bi a+b*j	Définit un nombre complexe a+bi
Fonction complexe	complex(a,b)	Produit un nombre complexe a+bi
Partie réelle	real(x)	Renvoie la partie réelle d'un nombre
Partie imaginaire	imag(x)	Donne la partie imaginaire du nombre x

Nom	Fonction	Description ou Commentaire
Norme	abs(x)	Renvoie la norme du nombre x
Angle	angle(x)	Donne l'angle du nombre x en radian
Conjugué	conj(x)	Produit le conjugué de x
Forme polaire	norme* exp(angle*j)	Produit un nombre complexe à partir de sa norme et de son angle
Flèche	arrow(p0,v)	Permet de tracer une flèche à partir de p0, selon un vecteur v.