

Guide MATLAB®

Court aide-mémoire pour les étudiants de MTH2210
Édition 0.2, le 22 août 2004

Steven Dufour
École Polytechnique de Montréal

Copyright © 2002, 2004 STEVEN DUFOUR.

Permission est accordée de copier, distribuer et/ou modifier ce document selon les termes de la Licence de documentation libre GNU (GNU Free Documentation License), version 1.1 ou toute version ultérieure publiée par la Free Software Foundation; avec la Section invariable étant « Annexe: License de documentation libre GNU »; avec les Textes de première de couverture qui sont « Guide MATLAB: Court aide mémoire pour les étudiants de MTH2210 » et sans Textes de quatrième de couverture. Une copie de la présente licence est incluse dans la section intitulée « Licence de Documentation Libre GNU ».

Introduction

Ce *court* guide d'introduction à MATLAB a pour but de tenter de faciliter vos premiers contacts avec ce logiciel et de vous servir de guide de référence lorsque vous complèterez vos travaux pratiques dans le cadre du cours MTH2210. Le but n'est pas de vous fournir un guide complet des commandes de MATLAB, mais de vous donner les outils nécessaires, **sans plus**, dans le cadre des activités de ce cours. En d'autres mots, tout ce dont vous avez besoin est là. Sinon, des informations supplémentaires vous seront fournies dans l'énoncé des problèmes à résoudre.

Ce guide prend en ligne de compte que vous êtes des aspirants ingénieurs, i.e. que vous êtes déjà rompus à l'informatique et que vous êtes suffisamment curieux pour faire un effort personnel afin de devenir confortable à utiliser cet outil. La présentation est donc la plus directe possible et nous ne ferons aucune introduction à l'utilisation de l'interface graphique de MATLAB. Explorez!

Plusieurs caractéristiques de MATLAB en font un outil simple et puissant. MATLAB est un langage *interactif*, i.e. dès que vous entrez une commande, vous avez une réponse immédiate du logiciel. MATLAB s'occupe de déterminer le *type* et la *taille* des variables mises en mémoire, ce qui facilite la tâche du programmeur en lui permettant d'expérimenter et de se concentrer sur les problèmes à résoudre. Tous les types 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. Une autre idée de base de MATLAB est que si vous vous demandez quel est le calcul qu'une commande exécutera, en général il s'agit de l'opération la plus naturelle. Finalement, l'apprentissage de l'approche *vectorielle* de MATLAB sera utile pour ceux qui auront à utiliser des ordinateurs à architecture vectorielle ou parallèle.

La première section de ce guide est un court tutoriel, qui peut être vu comme votre première session MATLAB. Il couvre les commandes et la philosophie de base de MATLAB. Vous trouverez ensuite un court catalogue de commandes MATLAB, sous la forme d'un tableau. Ce format a pour but de vous permettre de trouver rapidement la commande dont vous avez besoin. Pour chaque commande, un exemple d'utilisation est donné, avec un commentaire ou une illustration du résultat. Une courte bibliographie commentée est donnée en annexe, pour ceux qui veulent aller plus loin dans leur apprentissage de MATLAB. Lorsque ce document est consulté en ligne à l'aide du logiciel *Acrobat*, les hyperrefs dirigeront votre fureteur vers les documents pertinents.

Ce guide est dynamique. Je suis ouvert à toute suggestion de correction ou de modification. Des mises-à-jour de ce document seront rendues disponibles sur une base régulière. Consultez le numéro de l'édition et la date du document pour voir s'il a été mis à jour. Dans le but de créer des textes académiques, techniques et scientifiques de qualité, ce document est protégé par la licence GFDL de la « Free Software Foundation ». Le code \LaTeX du document est donc disponible sur mon [site Internet](#).

Steven Dufour
École Polytechnique de Montréal
mathappl.polymtl.ca/steven/
Le 22 août 2004

Tutoriel

Les premiers pas

Une fois MATLAB lancé, nous sommes en présence de l'invite («*prompt*») de MATLAB:

```
>>
```

MATLAB est prêt à recevoir nos commandes:

```
>> a=1           % tout ce qui vient après le symbole % est un commentaire
a =
     1
```

Notons qu'il n'est pas nécessaire de déclarer le type ou la taille de *a*. Nous voyons aussi un des avantages de travailler avec un langage interactif: nous avons une réponse immédiate de MATLAB.

```
>> A=2           % un autre scalaire, i.e. une matrice de taille 1x1
A =
     2
>> a             % on peut vérifier ce que contient une variable
a =
     1
>> A             % a et A sont des symboles différents
A =
     2
```

Voici comment déclarer un vecteur et une matrice:

```
>> b=[1 2]       % un vecteur ligne de dimension 1x2
b =
     1     2
>> b=[1,2]       % la même chose
b =
     1     2
>> c=[3;4;5]     % un vecteur colonne de dimension 3x1
c =
     3
     4
     5
>> d=[0.1 pi sqrt(2) 2+2 eps] % laissez aller votre imagination
d =
    0.1000    3.1416    1.4142    4.0000    0.0000
>> A=[1 2;3 4]   % une matrice de dimension 2x2 (on écrase A=2)
A =
     1     2
     3     4
```

On aura une meilleure idée de ce que contient *d* en changeant le format d'impression:

```
>> format short e % on aurait aussi pu utiliser <<format long e>>
>> d
d =
    1.0000e-01    3.1416e+00    1.4142e+00    4.0000e+00    2.2204e-16
>> format          % on revient au format initial
```

On peut facilement faire la concaténation de tableaux:

```
>> B=[A;b]           % on peut joindre une matrice et un vecteur ligne
B =
     1     2
     3     4
     1     2
>> C=[c B]           % ou une matrice et un vecteur colonne
C =
     3     1     2
     4     3     4
     5     1     2
```

On va chercher les composantes des tableaux de la même façon qu'avec d'autres langages:

```
>> c(2)               % la deuxième composante de c
ans =
     4
>> A(1,2)             % la composante de A en première ligne, deuxième colonne
ans =
     2
>> A(3)               % ou bien, puisque les entrées sont stockées par colonne
ans =
     2
```

Si nous n'assignons pas le résultat d'un calcul à une variable, MATLAB met le résultat dans la variable ans («answer»), que nous pouvons utiliser au même titre que les autres variables:

```
>> ans(1)+sin(1)+pi   % MATLAB peut être utilisé comme une calculatrice
ans =
    5.9831
```

MATLAB nous donne plus de liberté que les autres langages pour accéder aux composantes des tableaux:

```
>> c([2,3])           % les deuxième et troisième composantes de c
ans =
     4
     5
>> A([2,3,4])         % les composantes (1,2), (2,1) et (2,2) de A
ans =
     3     2     4
```

L'opération précédente peut vous sembler plus intuitive si nous utilisons la commande sub2ind:

```
>> indices=sub2ind(size(A),[2,1,2],[1,2,2]);
>> A(indices)
ans =
     3     2     4
```

Une condition booléenne peut aussi être utilisée pour accéder aux composantes voulues:

```
>> A(A>3)             % les composantes de A supérieures à 3
ans =
     4
```

```
>> A(mod(A,2)~=0)    % on recherche les entrées impaires
ans =
     1
     3
```

Dans l'environnement MATLAB, une expression booléenne vraie est égale à 1 et une expression fausse est égale à 0. Les opérateurs booléens sont similaires à ceux rencontrés en C, soient « == », « ~ », « & » et « | » pour n'en nommer que quelques-uns.

Il est facile de modifier les matrices:

```
>> A(1,1)=0          % n'oubliez pas que les indices commencent à 1
A =
     0     2
     3     4
>> A(3,3)=9          % la matrice est redimensionnée automatiquement
A =
     0     2     0
     3     4     0
     0     0     9
```

Nous sommes aussi assurés que les autres nouvelles composantes seront initialisées à « 0 ». MATLAB nous donne aussi des outils puissants pour modifier les tableaux. Voici deux exemples:

```
>> A([6,8])=[7,5]    % puisque les entrées sont stockées par colonne
A =
     0     2     0
     3     4     5
     0     7     9
>> A(sub2ind(size(A),[1,3,1],[1,1,3]))=1    % quel est le résultat?
```

Nous verrons d'autres exemples à la section suivante. Si nous ne nous souvenons plus des variables que nous avons utilisées:

```
>> whos
  Name          Size          Bytes  Class
  A              3x3              72  double array
  B              3x2              48  double array
  C              3x3              72  double array
  a              1x1               8  double array
  ans            2x1              16  double array
  b              1x2              16  double array
  c              3x1              24  double array
  d              1x5              40  double array
  indices        1x3              24  double array

Grand total is 40 elements using 320 bytes
```

Nous voyons que tous les objets créés sont des matrices avec des composantes stockées comme des réels en format double précision IEEE, même si nous avons défini ces tableaux à l'aide d'entiers.

L'opérateur «:»

Un outil puissant de MATLAB est l'opérateur «:»:

```
>> x=1:10      % création d'un vecteur ligne formé de 10 entiers
x =
     1     2     3     4     5     6     7     8     9    10
>> x=10:-1:1    % à rebours
x =
    10     9     8     7     6     5     4     3     2     1
>> x=0:0.1:0.5  % un incrément de 0.1 plutôt que 1
x =
         0    0.1000    0.2000    0.3000    0.4000    0.5000
```

On peut utiliser l'opérateur «:» pour extraire plusieurs composantes d'un vecteur ou d'une matrice:

```
>> d=x(2:4)     % on va chercher les composantes 2, 3 et 4 du vecteur x
d =
    0.1000    0.2000    0.3000
>> B=A(2:3,2:3) % soutirons une matrice 2x2 de la matrice A définie plus haut
B =
     4     5
     7     9
>> e=A(2,:)     % ou si on veut extraire la deuxième ligne de A
e =
     3     4     5
>> e=A(2:end,1) % utilisons le dernier indice de la ligne (end=3 ici)
e =
     3
     1
```

Un outil utile pour modifier les matrices est le tableau vide «[]»:

```
>> C(2,:)=[]    % on peut effacer la deuxième ligne de C
C =
     3     1     2
     5     1     2
```

qui est plus pratique que la commande équivalente `C=C([1 3],:)` pour de grands tableaux. L'opérateur «:» peut aussi être utilisé pour intervertir les lignes et les colonnes des tableaux:

```
>> C(:,[3 2 1])
ans =
     2     1     3
     2     1     5
```

Les opérations mathématiques

Les opérations de MATLAB entre vecteurs et matrices suivent les conventions utilisées en mathématiques:

```
>> x=(1:3);  
>> y=(3:-1:1);  
>> x-y  
ans =  
    -2     0     2  
>> x*y  
??? Error using ==> *  
Inner matrix dimensions must agree.
```

Du point de vue de l'algèbre linéaire, nous ne pouvons pas multiplier deux vecteurs lignes. Utilisons l'opérateur de transposition «'»:

```
>> x*y'  
ans =  
    10  
>> dot(x,y);           % on obtient le même résultat  
>> B*b                 % même problème entre la matrice B et b=[1 2]  
??? Error using ==> *  
Inner matrix dimensions must agree.  
>> B*e                 % pas de problème avec e=[3;1]  
ans =  
    17  
    30
```

Pour ce qui est des opérations matrice-matrice:

```
>> B*C                 % produit matrice-matrice  
ans =  
    37     9    18  
    66    16    32  
>> B^2                 % l'opération B*B  
ans =  
    51     65  
    91    116  
>> B\C                 % les divisions matricielles  
ans =  
     2     4     8  
    -1    -3    -6  
>> inv(B)*C            % qui est équivalent à B\C  
ans =  
    2.0000    4.0000    8.0000  
   -1.0000   -3.0000   -6.0000  
>> b/B                 % qui est équivalent à b*inv(B)  
ans =  
    -5     3
```


Ce qui nous mène naturellement vers la résolution d'un système d'équations linéaires $A\vec{x} = \vec{c}$:

```
>> x=A\c           % résolution par factorisation LU, et non par inv(A)*c
x =
    0.4231
    1.6923
   -0.8077
```

Une autre caractéristique de MATLAB qui en fait un outil puissant est sa série d'opérations sur les composantes des tableaux:

```
>> B/2             % chaque composante est divisée par 2
ans =
    2.0000    2.5000
    3.5000    4.5000
>> B+1             % on ajoute 1 à chaque composante de B (et non B+I)
ans =
     5     6
     8    10
```

qui donne le même résultat que

```
>> B+ones(2,2)     % ones(2,2) est une matrice de <<1>> de dimension 2x2
ans =
     5     6
     8    10
```

Les fonctions mathématiques élémentaires sont aussi définies sur les composantes des tableaux:

```
>> sqrt(B)
ans =
    2.0000    2.2361
    2.6458    3.0000
```

Qu'en est-il des opérations «*», «/» et «^» sur les composantes des tableaux? C'est ici que l'opérateur «.» vient simplifier beaucoup notre tâche:

```
>> B.*B             % B(i,j)*B(i,j) plutôt que B*B
ans =
    16    25
    49    81
>> C.^2             % C(i,j)^2 plutôt que C^2=C*C, qui n'a pas de sens ici
ans =
     9     1     4
    25     1     4
```

Par exemple, ces opérations sont utiles pour simplifier l'impression de tableaux de résultats numériques:

```
>> n=1:5;
>> [n ; n.^2 ; 2.^n]'
ans =
     1     1     2
     2     4     4
     3     9     8
     4    16    16
     5    25    32
```

Nous verrons plus loin que cette notation est souvent utilisée pour définir les fonctions.

Les programmes MATLAB (fichiers-M)

Afin d'éviter à 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 (Menu « File → New → M-File »), de créer un fichier en format texte qui contient une série de commandes MATLAB. 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. 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 ↑). Cette procédure vous évite de retaper une série de commandes à répétition. C'est la procédure recommandée pour vos travaux pratiques.

Il est possible de programmer des boucles et des branchements dans les fichiers-M:

```
for i=1:10      % boucle for
    if i<5      % branchement
        x(i)=i; % n'oubliez pas votre point-virgule, sinon...
    else        % on peut utiliser le else
        x(i)=0;
    end         % on n'oublie pas de terminer le branchement
end            % et la boucle
```

Le point-virgule à la fin d'une ligne signale à MATLAB de ne pas retourner le résultat de l'opération 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 un programme, afin de voir ce qui se passe.

En général, on essaye d'éviter un tel programme. Puisque l'on utilise un langage interactif, ce bout de code fait un appel au logiciel à chaque itération de la boucle et à chaque évaluation de la condition. De plus, le vecteur change de taille à chaque itération. MATLAB doit donc faire une demande au système d'exploitation pour avoir plus de mémoire à chaque itération. Ce n'est pas très important pour un programme de petite taille, mais si on a un programme qui fait une quantité importante de calculs, un programme optimisé peut être accéléré par un facteur 1000!

Il est possible d'éviter ces goulots d'étranglement en demandant initialement l'espace mémoire nécessaire à vos calculs, puis en *vectorisant* la structure du programme:

```
x=zeros(1,10); % un vecteur de dimension 1x10 est initialisé à zéro
x(1:4)=1:4;    % une boucle à l'aide de l'opérateur :
```

Dans cet exemple, l'espace mémoire n'est alloué qu'une seule fois, et il n'y a que deux appels faits à MATLAB. Notons que le concept de vectorisation n'est pas particulier à MATLAB. Il est aussi présent dans les langages de programmation d'ordinateurs vectoriels et parallèles. Ceci étant dit, précisons que la vectorisation d'algorithmes est un « art » difficile qui demande du travail.

Les programmeurs expérimentés prennent l'habitude de débiter leurs programmes MATLAB avec une série de leurs commandes préférées, qui ont pour but d'éviter les mauvaises surprises. Par exemple:

```
>> clear all;      % on efface toutes les variables, fonctions, ...
>> format compact; % suppression des retours de chariots superflus
>> format short e;  % éviter qu'une petite quantité s'affiche comme 0.0000
>> dbstop if error  % faciliter l'impression d'une variable en cas d'erreur
```

Les fonctions MATLAB

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 cond` pour avoir de l'information sur la fonction `cond`. 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 cond`, par exemple.

Il est aussi possible de créer ses propres fonctions MATLAB. Le concept de fonction en MATLAB est similaire aux fonctions avec d'autres langages de programmation, i.e. une fonction prend un/des argument(s) en entrée et produit un/des argument(s) en sortie. La procédure est simple. Il s'agit de créer un fichier-M, nommons-le `mafonction.m`. Ce qui différencie un fichier-M d'une fonction est que la première ligne de la fonction contient le mot clef `function`, suivi de la définition des arguments en entrée et en sortie. Par exemple, voici une fonction qui interverti l'ordre de deux nombres:

```
function [y1,y2]=mafonction(x1,x2)
%
% Définition de la fonction "mafonction":
% arguments en entrée: x1 et x2
% arguments en sortie: y1 et y2
%
y1 = x2;
y2 = x1;
```

Il s'agit ensuite d'appeler votre fonction à l'invite MATLAB:

```
>> [a,b]=mafonction(1,2)
a =
    2
b =
    1
```

Si on veut définir une fonction pour calculer x^2 , on écrira

```
function y=carre(x)
%
% Définition de ma fonction x^2
%
y = x*x;
```

En entrant `help carre`, vous verrez les lignes de commentaires qui suivent immédiatement le mot clef `function`:

```
>> help carre

Definition de ma fonction x^2
```

Utilisez cette fonctionnalité pour vos fonctions.

Les communications entre les fonctions et les programmes se fait donc à l'aide des arguments en entrée et en sortie. La portée des variables définies à l'intérieur d'une fonction est donc limitée à cette fonction.

Cependant, dans certaines situations il peut être pratique d'avoir des variables globales, qui sont définies à l'aide de la commande `global`:

```
function y=Ccarre(x)
%
% Définition de la fonction (C^3) x^2
%
    global C          % variable globale, initialisée à l'extérieur de Ccarre
    e = 3             % variable locale à Ccarre, donc invisible ailleurs
    y = (C^e)*(x*x);
    C = 1             % C est modifiée partout où elle est définie comme globale
```

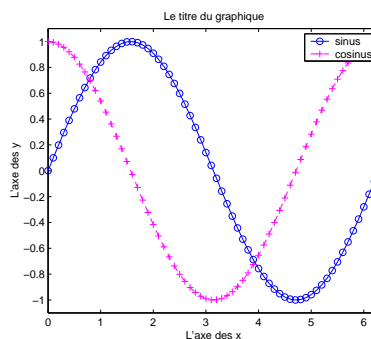
où la variable `C` serait aussi définie comme globale dans le fichier-M où on veut y avoir accès. Par convention, les variables globales sont écrites en majuscules.

Les graphiques

La fonction de base pour tracer un graphique avec MATLAB est la commande `plot`, qui prend comme arguments une série de points donnés sous la forme de 2 vecteurs, qu'elle relie de segments de droites. C'est la responsabilité de l'utilisateur de générer assez de points pour qu'une courbe régulière paraisse régulière à l'écran. De plus, il est possible de donner des arguments additionnels, ou d'utiliser d'autres fonctions, pour contrôler l'apparence du graphique:

```
>> x=0:0.1:2*pi;                % L'ordonnée
>> plot(x,sin(x),'b-o',x,cos(x),'m--+'); % Le graphe de sinus et de cosinus
>> axis([0 2*pi -1.1 1.1]);      % Définition des axes
>> title('Le titre du graphique');
>> xlabel('L''axe des x');
>> ylabel('L''axe des y');
>> legend('sinus','cosinus');
```

Ces commandes donnent comme résultat:



On aurait aussi pu tracer ce graphique avec les commandes:

```
>> clf reset                    % on réinitialise l'environnement graphique
>> hold on
>> plot(x,sin(x),'b-o');        % un premier graphique
>> plot(x,cos(x),'m--+');       % on superpose un deuxième graphique
>> hold off
```

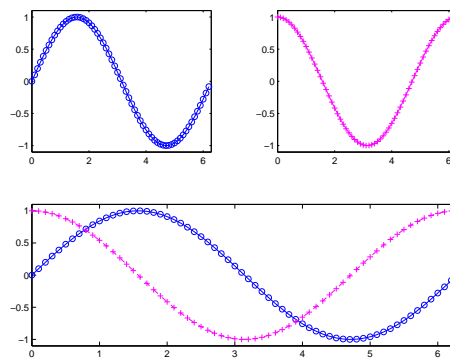
La commande `hold` évite que le premier graphique soit écrasé par le deuxième.

L'opérateur «`:`» n'est pas idéal pour créer le vecteur `x`. Comme vous pouvez le voir sur la figure précédente, `x(end)` n'est pas égal à 2π . Dans ce cas, on préfère utiliser la commande `linspace` pour générer le nombre de points voulu dans un intervalle donné, alors que l'opérateur «`:`» nous donne plutôt le contrôle sur la distance entre les points.

```
>> x1=0:1:2*pi % combien a-t-on de points entre 0 et 2*pi?
x1 =
    0    1    2    3    4    5    6
>> x2=linspace(0,2*pi,7) % ici on sait qu'on en a 7, allant de 0 à 2*pi
x2 =
    0    1.0472    2.0944    3.1416    4.1888    5.2360    6.2832
```

On peut aussi comparer des résultats sous forme graphique à l'aide de la commande `subplot`:

```
>> subplot(2,2,1); plot(x,sin(x),'b-o');axis([0 2*pi -1.1 1.1]);
>> subplot(2,2,2); plot(x,cos(x),'m-+');axis([0 2*pi -1.1 1.1]);
>> subplot(2,2,3:4);plot(x,sin(x),'b-o',x,cos(x),'m-+');
>> axis([0 2*pi -1.1 1.1]);
```



La fonction `fplot` facilite le tracé de graphes de fonctions, en automatisant le choix des points où les fonctions sont évaluées:

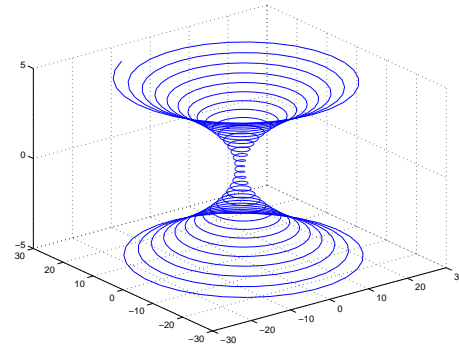
```
>> fplot('sin(x),cos(x)',[0 2*pi],'b-+')
```

On perd cependant de la latitude dans le choix de certaines composantes du graphique.

Finalement, les graphiques tridimensionnels, de type paramétrique, sont tracés à l'aide d'une généralisation de la commande `plot`:

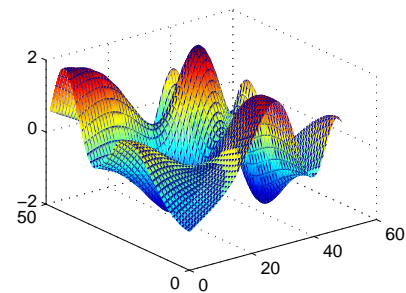
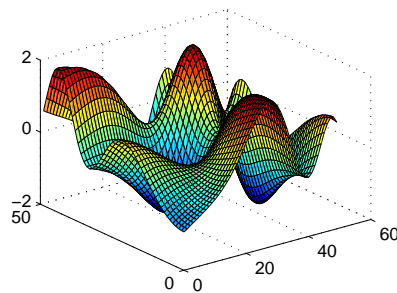
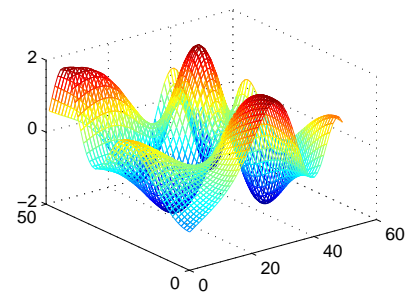
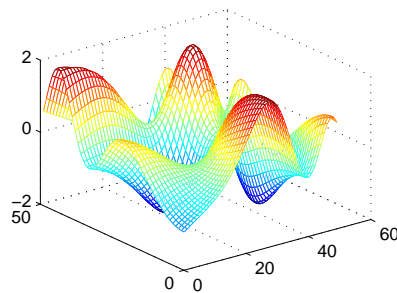
```
>> t=linspace(-5,5,1000);
>> x=(1+t.^2).*sin(20*t); % utilisation de l'opérateur .^
>> y=(1+t.^2).*cos(20*t);
>> z=t;
>> plot3(x,y,z);
>> grid on;
```

qui donne comme résultat:



Pour ce qui est des graphes tridimensionnels, une étape intermédiaire est nécessaire avant d'utiliser les diverses commandes à notre disposition:

```
>> x=linspace(0,pi,50);
>> y=linspace(0,pi,50);
>> [X,Y]=meshgrid(x,y);           % on génère une grille sur [0,pi]x[0,pi]
>> Z=sin(Y.^2+X)-cos(Y-X.^2);
>> subplot(2,2,1); mesh(Z);
>> subplot(2,2,2); mesh(Z); hidden off;
>> subplot(2,2,3); surf(Z);
>> subplot(2,2,4); surf(Z); shading interp
```



Les sorties formatées

Nous aurons souvent à présenter des résultats numériques sous la forme de tableaux de nombres. Les programmeurs C ne seront pas dépayés par les commandes MATLAB qui nous permettent de formater l'écriture dans des fichiers :

```
>> uns=(1:5)' ones(5,2) % création d'un tableau
uns =
     1     1     1
     2     1     1
     3     1     1
     4     1     1
     5     1     1
>> sortie=fopen('sortie.txt','w'); % ouverture du fichier
>> fprintf(sortie,'\t%s\n','Une serie de uns:'); % écriture de caractères,
>> fprintf(sortie,'\t%s\n','-----'); % d'entiers et de réels
>> fprintf(sortie,'\t%2d %4.2f %5.2e\n',uns'); % attention à la transposée
>> fclose(sortie); % fermeture du fichier
```

où la sortie dans le fichier `sortie.txt` sera:

```
>> type sortie.txt

    Une serie de uns:
    -----
    1 1.00 1.00e+00
    2 1.00 1.00e+00
    3 1.00 1.00e+00
    4 1.00 1.00e+00
    5 1.00 1.00e+00
```

Les caractères de contrôle pour le format de l'impression sont identiques à ceux que l'on retrouve dans le langage C. Dans l'exemple ci-dessus, «`\t`» représente une tabulation, «`\n`» un saut de ligne, «`%s`» une chaîne de caractères, «`%nd`» imprime un entier sur n caractères, «`%m.nf`» un réel sur m caractères avec n caractères après le point et «`%m.ne`» fait la même chose en notation scientifique.

Comme la plupart des commandes MATLAB, `fprintf` peut être utilisée sous forme vectorielle. Vous êtes donc encouragés à utiliser cette fonctionnalité dans vos programmes pour qu'ils s'exécutent plus rapidement.

Dernières remarques

Nous avons vu plus haut que la commande `type` nous permet d'afficher le contenu d'un fichier. Les commandes `dir` et `ls` affichent le contenu d'un répertoire et `cd` nous permet de changer de répertoire. Nous pouvons aussi passer des commandes directement au système d'exploitation en les précédant d'un «`!`» à l'invite MATLAB. Finalement, la commande `path` nous permet d'ajouter un chemin vers les fichiers-m que nous voulons utiliser. Par exemple, si nous avons des fichiers MATLAB sur une disquette:

```
>> path(path,'A:'); % sur DOS
>> path(path,'/mnt/floppy'); % sur RedHat Linux, par exemple
```

Ces chemins seront alors ajoutés à la variable `path`, qui contient tous les chemins.

Catalogue de commandes

Disponible bientôt.

Bibliographie

Les documents officiels

- [Getting Started with MATLAB](#) (*Un « court » guide d'introduction*)
- [Mathematics](#) (*Les commandes mathématiques*)
- [Using MATLAB Graphics](#) (*Le manuel de référence pour tout ce qui est graphique*)
- [Programming](#) (*Le manuel de référence pour la programmation*)
- [Programming Tips](#) (*Trucs de programmation MATLAB*)
- [Function Reference Volume 1: A-E, Volume 2: F-O, Volume 3: P-Z](#)
(*La liste de toutes les fonctions, en 3 volumes!*)

Les notes techniques

- [What Is the EVAL Function?](#) (*Note technique 1103*)
- [Memory Management Guide](#) (*Note technique 1106*)
- [Code Vectorization Guide](#) (*Note technique 1109*)
- [Movie and Animation Guide](#) (*Note technique 1204*)

Les livres

- [MATLAB Guide](#), par Higham et Higham
- [Mastering MATLAB 6](#), par Hanselman et Littlefield

Les guides

- [MATLAB: Guide d'apprentissage](#), par Cloutier, Gourdeau et Leblanc (disponible à COOPOLY)
- [An Introduction to MATLAB](#), par David F. Griffiths
- [Introduction to MATLAB](#), par Graeme Chandler
- [A Practical Introduction to MATLAB](#), par Mark S. Gockenbach
- [Introduction to MATLAB](#), par Jeffery Cooper
- [MATLAB Array Manipulation Tips and Tricks](#), par Peter J. Acklam

Autres documents

- [Floating points](#), par Cleve Moler (*La norme IEEE et MATLAB*)
- [Exporting Figures for Publication](#), par Ben Hinkle
- [Exporting Figures for Publication: Part 2](#), par Ben Hinkle
- [MATLAB FAQ](#)

Annexe: Licence de Documentation Libre GNU

GNU Free Documentation License

Version 1.1, March 2000

Copyright © 2000 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0 Preamble

The purpose of this License is to make a manual, textbook, or other written document “free” in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of “copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1 Applicability and Definitions

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The “Document”, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as “you”.

A “Modified Version” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A “Secondary Section” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “Invariant Sections” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not “Transparent” is called “Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Textinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

2 Verbatim Copying

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may

accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3 Copying in Quantity

If you publish printed copies of the Document numbering more than 100, and the Document’s license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4 Modifications

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).
- State on the Title page the name of the publisher of the Modified Version, as the publisher.
- Preserve all the copyright notices of the Document.
- Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document’s license notice.
- Include an unaltered copy of this License.
- Preserve the section entitled “History”, and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled “History” in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

- Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the “History” section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- In any section entitled “Acknowledgements” or “Dedications”, preserve the section’s title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- Delete any section entitled “Endorsements”. Such a section may not be included in the Modified Version.
- Do not retile any existing section as “Endorsements” or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version’s license notice. These titles must be distinct from any other section titles.

You may add a section entitled “Endorsements”, provided it contains nothing but endorsements of your Modified Version by various parties – for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5 Combining Documents

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled “History” in the various original documents, forming one section entitled “History”; likewise combine any sections entitled “Acknowledgements”, and any sections entitled “Dedications”. You must delete all sections entitled “Endorsements.”

6 Collections of Documents

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7 Aggregation With Independent Works

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a

Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an “aggregate”, and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document’s Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

8 Translation

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9 Termination

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10 Future Revisions of This License

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License “or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright © YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST. A copy of the license is included in the section entitled “GNU Free Documentation License”.

If you have no Invariant Sections, write “with no Invariant Sections” instead of saying which ones are invariant. If you have no Front-Cover Texts, write “no Front-Cover Texts” instead of “Front-Cover Texts being LIST”; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.