INTRODUCTION A MATLAB

MATLAB, c'est un logiciel permettant d'effectuer des opérations complexes sur des matrices et vecteurs directement avec la syntaxe matricielle (presque la même écriture que dans les livres et publications). Matlab, c'est un interpréteur d'opérations matricielles. C'est aussi un langage de programmation de haut niveau. Matlab est écrit en C, mais il ne demande aucune connaissance du C de l'utilisateur.

Avec MATLAB, on peut effectuer tout ce que l'on peut faire avec un langage de programmation, mais beaucoup plus rapidement et efficacement. C'est un outil de recherche et d'application fantastique pour qui maîtrise bien l'écriture matricielle (et ses notions d'algèbre linéaire). Des petits exemples :

- l'inversion d'une matrice A s'écrit: inv(A).
- Les valeurs propres et vecteurs propres de A s'obtiennent de: [u,l]=eig(A).
- Les coefficients « b » d'une régression de Y par des variables X: b=inv(X'*X)*X'*Y
- etc...

Entrée des données

Deux façons: 1- Au terminal: une matrice commence par [et se termine par]. Les éléments d'une même ligne sont séparés par un blanc ou une virgule, les lignes sont séparées par un point-virgule.

ex. A=[4 3 1 ; 5 7 8] construit une matrice A ayant 2 lignes et 3 colonnes.

2- En lisant un fichier « texte » déjà existant avec la commande load «nom du fichier».

Corriger une erreur

Si la syntaxe est incorrecte, Matlab le détecte et fait une grimace sonore. On peut corriger n'importe quelle commande, celles-ci sont placées dans une liste circulaire. On accède aux commandes déjà entrées à l'aide des curseurs. On peut aussi taper le début de la commande et « flèche haut », Matlab retourne alors à la 1ère commande ayant le même début puis la seconde…

Types de fichiers

En Matlab, il y a 3 types de fichiers principaux :

1- fichiers « .m » : fichiers de commandes de Matlab

2- fichiers « .mat » : fichiers contenant des données (format spécifique à Matlab).

3- fichiers « .fig » : fichiers non-lisible contenant ce qu'il faut pour recréer une figure lors d'une séance ultérieure.

Fichier .m

Un fichier .m est un fichier contenant des énoncés Matlab comme si on les entrait soi-même au terminal. Ces énoncés peuvent être d'autres fichiers .m . On peut même rappeler le même fichier .m (récursivité). Il y a deux types de fichiers m:

1- Les fichiers .m ordinaires (script). Les variables sont toutes globales.

2- Les fichiers .m fonctions. Une ligne au début du fichier identifie qu'il s'agit d'un fichier fonction. Les variables de cette fonction sont alors locales et détruites après exécution. Les variables globales ne sont pas accessibles. On doit passer l'information d'entrée et récupérer la

sortie dans des variables passées en paramètres. Bref, cela s'apparente beaucoup aux sous-routines des langages de programmation usuels (Pascal, C, fortran)...

1- Fichiers scripts:

- On édite un fichier et on entre une série de commandes comme si on le faisait au terminal.
- On ferme le fichier sous le nom « nom_du_fichier »
- Dans Matlab, on tape >>nom_du_fichier.

2- Fichiers functions:

Analogue à une sous-routine en Pascal. La première ligne d'une fonction disons « acp » doit débuter par:

function [out1,out2,...outn]=acp(inp1,inp2...inpm)

suivent une série de commentaires qui seront affichés lors de help(acp)
suivent les commandes requises.
Cette fonction sera normalement sauvegardée par l'usager sous le nom acp.m. Dans Matlab, la fonction est connue par *le nom de sauvegarde du fichier* et non par le nom donné à la fonction sur la 1ère ligne.

Fichier .mat

Des fichiers .mat obtenus par «save» peuvent être relus avec «load». Ces fichiers contiennent les variables (en tout ou en partie) créées lors d'une session précédente. Ces fichiers ne sont pas lisibles directement en dehors de Matlab.

ex. >>save données % sauve toutes les variables et matrices (le « workspace ») dans le fichier données.mat

>>save données m1 m2 var1 var2 % sauve les variables et matrices m1, m2 , var1, var2 dans le fichier donnees.mat

On peut rendre un fichier de sauvegarde lisible par un éditeur de texte par exemple en faisant :
>>save fichier variable(s) –ascii. Les variables spécifiées seront dans fichier et pourront être lues dans Excel ou Word.

Ordre de préséance des noms : que fait Matlab quand il rencontre un nom dans un énoncé?

Dans l'ordre:
- Variable?
- Fonction interne?
- Fichier .m dans le répertoire actuel?
- Fichier .m dans un autre répertoire (matlabpath). (suit l'ordre des répertoires du « path » actuel)?

Quelques petits trucs et conseils

1. Matlab distingue les majuscules et minuscules. la matrice « M » et la matrice « m » ne sont donc pas la même entité pour Matlab. Règle générale évitez l'utilisation des majuscules.
2. Il ne faut pas donner des noms aux variables qui correspondent à des noms de fonction (soit celles de Matlab, soit celles de l'usager), sinon Matlab ne peut plus accéder à ces fonctions.
3. On utilise au maximum les fichiers « .m » (scripts et fonctions). Ceci sauve énormément de temps en bout de ligne. Rapidement, on doit apprendre à utiliser la page de base de Matlab comme une feuille de papier brouillon et réserver le travail sérieux aux fichiers « .m ».
4. Insérer les commentaires au fur et à mesure de l'écriture des fichiers « .m » et non seulement lorsque la programmation est terminée.
5. Il faut « vectoriser » le plus possible les opérations et éviter les boucles. Souvent, naturellement, on est porté à tout écrire sous forme de boucles. Ce n'est pas un problème pour de petites opérations mais un gain de temps considérable peut être obtenu en « vectorisant » ces opérations.

   ex. Trouver la différence maximale entre le vecteur x et le vecteur y

   ```
   >>for i=1 :n; dif(i)=x(i)-y(i); end; maxdif=max(dif);
   ```

   est beucoup plus lent que :

   ```
   maxdif=max(x-y);
   ```

   L'habileté à « vectoriser » les opérations s'acquiert avec l'expérience et augmente avec la compréhension du problème.
6. Il est préférable, surtout pour les grandes matrices remplies à l'intérieur de boucles, de les initialiser au début d'une procédure.

« Toolbox » (librairie de fonctions):

Matlab vient avec un nombre impressionnant de fonctions de base. Le fabricant distribue aussi des librairies de fonctions supplémentaires destinés à des usages spécifiques. Ces « toolbox » couvrent des domaines comme :

- analyse d'images
- statistique
- optimisation
- analyse de signal
- réseaux neuronaux
- analyse symbolique
- etc.

De plus, un grand nombre de fonctions et de toolbox sont disponibles gratuitement sur le WEB, y inclut sur le site de « mathworks » (www.mathworks.com). Ces fonctions sont parfois supérieures à celles du produit original. Il existe, entre autres, d'excellentes librairies pour produire des cartes contours de bonne qualité.

Affichage numérique :

On peut contrôler l'affichage des quantités numériques avec la commande « format ». >>format short g est un des plus pratiques. >>format compact permet d'éviter les lignes blanches séparant chaque entrée sortie.

Comment apprendre à utiliser Matlab?

Progressivement, en travaillant avec Matlab, et en faisant beaucoup d'erreurs (Matlab retombe vraiment bien sur ses pieds), en ayant recours à «help», à «help commande», à « lookfor », en faisant exécuter les démos avec la commande «demo», en n'hésitant jamais à poser une question.

Liste des éléments syntaxiques et des commandes parmi les plus importantes

Opérateurs

!	Exécute une commande de Dos. ex. !dir
[ ]	Définit un vecteur ou une matrice; peut être imbriqué plusieurs fois. ex. b=[2 13 4]
;	Sépare des lignes dans une matrice ex. b=[2 13 4;1 4 5] Aussi, empêche l'output d'une commande d'être affiché.
,	Sépare des éléments dans une matrice ou dans une commande ex. min(a,b)
=	Assignation. ex. a=b place une copie de b dans a.
==	Opérateur d'égalité. Retourne 1 si égal, 0 sinon.
~	Inverse les valeurs logiques (0 devient 1 et 1 devient 0)
~=	Non égal
<	Plus petit que
>	Plus grand que
<=	Plus petit ou égal.
>=	Plus grand ou égal.
|	Ou (union logique).
&	Et (intersection logique).
+	Addition (scalaire et matricielle).
-	Soustraction (scalaire et matricielle).
*	Multiplication (scalaire et matricielle).
.*	Multiplication élément par élément pour 2 matrices de même taille.
/	Division (scalaire et matricielle). a/b équivalent à a*inv(b) si b est carrée.
\	Division matricielle: a\b équivalent à inv(a)*b si a est carrée.
./ ou .\	Division élément par élément.
^	Exposant, ex. 2^3 a^3. Si a est une matrice carrée: matrice à la puissance 3 soit a*a*a.
.^	a.^5.2 : chaque élément de a est élevé à la puissance 5.2.
'	Transposée d'une matrice. Aussi entrée de texte dans une matrice ou un vecteur.
%	Commentaires dans un fichier .m.
>>	Prompt de Matlab.
NaN	Not a Number. Peut indiquer des valeurs manquantes. Il faut faire attention à la façon dont chaque fonction traite les valeurs manquantes.

Façons de référer à un ou plusieurs éléments d'une matrice:

a(2,3)=7	Place la valeur 7 à la position 2,3 de la matrice a.
a(:,3)=[1;1]	Place des 1 en 3^{e} colonne de a.
a(:,:)=ones(2,3)	Place des 1 partout dans a
b(1:3,5:8)=zeros(3,4)	Mets des zéros dans les positions (1,5),(2,5),(3,5),(1,6)...(3,8)
b([1 4],5:8)=zeros(2,4)	Mets des zéros aux positions (1,5),(4,5),(1,6),(4,6)...(4,8)
b(v,c)=...	Les lignes et les colonnes de b affectées sont données dans les vecteurs v et c. ex. v=[1,4,9] c=[2,3].
b(id,c)=ones(3,2)	Les lignes et les colonnes de b affectées sont données dans les vecteurs id et c. ex. id=[0 0 1 1 0 1] c=[2,3]. Note: la longueur de id doit correspondre ici au nombre de lignes de b. Le nombre de 1 détermine le nombre de lignes de la matrice à droite de l'affectation.

Commandes fréquentes

abs	Valeur absolue.
axis	Permet de fixer manuellement les limites des axes.
bar	Graphe, genre histogramme.
break	Arrêt.
clg	Vide l'écran graphique.
contour	Isocontours.
corrcoef	Matrice de corrélations.
cov	Matrice de covariances.
demo	Exécute un démo.
det	Déterminant.
diag	Créer une matrice diagonale ou extraire la diagonale d'une matrice.
diary	Permet de conserver les commandes tapées au terminal dans un fichier.
eig	Décomposition en valeurs propres et vecteurs propres.
end	Termine un if, for, while.
exp	Exponentiel.
eye	Matrice identité.
fix	Tronquer.
for	Boucle.
format	Change l'affichage des quantités numériques.
function	Définir une nouvelle fonction.
help	Donne la liste des commandes.
help(com)	Décrit la commande.
hist	Histogrammes.
hold	Retient le graphe, les autres graphiques sont superposés.
if..elseif..end	Test.
inv	Inverse d'une matrice carrée.
keyboard	retourne le contrôle au clavier à l'intérieur d'une fonction.
length	Longueur d'un vecteur.
load	Charger un fichier (.mat ou fichier ascii).
log log10	Logarithmes naturel et en base 10.
loglog	Graphe, échelle log-log.
max	Maximums des colonnes.
mean	Moyennes des colonnes d'une matrice.
mesh	Vue 3-D d'une surface.

min	Minimums des colonnes.
ones	Matrice de uns.
plot	Diagramme binaire
rand	Matrice de nombres aléatoires.
round	Arrondir.
save	Sauver dans un fichier (.mat ou ascii).
semilogx	Graphe, échelle log en x.
semilogy	Graphe, échelle log en y.
sin cos...	Fonctions trigonométriques.
size	Donne la taille d'une matrice ou d'un vecteur. ex. size(a)
sort	Tri des colonnes d'une matrice.
sqrt	Racine carrée.
std	Ecarts-types des colonnes d'une matrice.
subplot	Divise l'écran graphique en plusieurs sous-graphes.
sum	Somme des éléments d'une matrice.
svd	Décomposition en valeurs singulières (comme eig pour des matrices carrées).
text	Positionne des annotations à un graphe.
title	Titre sur un graphe
type	Comme pour dos.
while	Tant que.
whos	Donne la liste des variables et matrices définies par l'usager ainsi que leurs dimensions.
xlabel	Titre axe x.
ylabel	Titre axe y.
zeros	Matrice de zéros.

Éditeur Matlab :

L'éditeur Matlab est très puissant. Il permet de :

- Décaler les lignes automatiquement dans les boucles, énoncés « if », etc.
- Effectuer une vérification de la syntaxe (parenthèses qui s'équilibrent, etc.).
- Donner une rétroaction sous forme de couleur pour les commandes reconnues.
- Stopper le programme à l'intérieur d'une fonction soit obligatoirement, soit suite à une condition donnée (erreur, avertissement, etc.). une fois le programme stoppé, on peut consulter les variables locales à la fonction ou à toute autre fonction encore en voie d'évaluation, y inclut l'espace de travail.

Site internet:

www.mathworks.com

Vous y trouverez nombre de réponses aux bugs que vous pourriez rencontrer, une description des « toolbox » et des fonctions du domaine public.

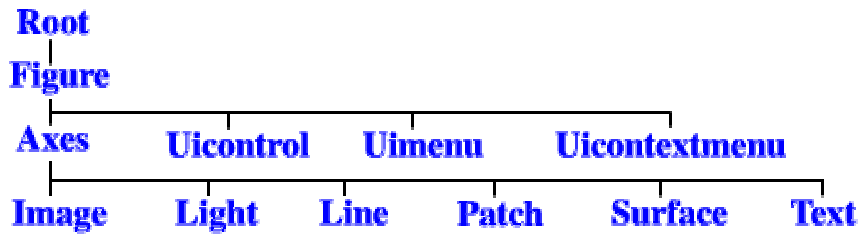
À propos des graphiques

Les fonctions graphiques de Matlab sont extrêmement puissantes. À priori, il faut se dire qu'il est possible de faire à peu près tout ce que l'on veut en fait de graphiques. La réalisation demande parfois un certain effort mais en vaut la chandelle. Matlab 6.0 et + est doté d'une interface pour modifier interactivement les

graphiques. On peut aussi sauver les graphiques (.fig) dans un fichier ou les exporter dans des formats courants.

Exemples de graphiques : on peut faire des diagrammes de points en 2D ou 3D, des surfaces en 2D ou 3D, des cartes isovaleurs. On peut superposer plusieurs graphiques sur les mêmes axes ou créer de nouveaux axes. On peut mettre des étiquettes où bon nous semble, on peut contrôler totalement la taille et le style des marqueurs, des lignes, la palette de couleur pour les surfaces. On peut contrôler la taille des figures à l'écran et à l'impression, etc. On peut mettre des boutons, construire des menus, etc.

Pour Matlab, une figure est un objet qui contient d'autres objets qui eux-mêmes contiennent d'autres objets. La structure hiérarchique des objets est la suivante :



Chaque objet vient avec une liste impressionnante de propriétés que l'on peut modifier à loisir. De plus à chaque objet est associé un « handle » (un identificateur) qui permet d'accéder à l'objet pour en modifier les propriétés.

Lorsque vous utilisez la commande « plot » par exemple, Matlab crée pour vous un objet « figure » un objet « axes » et un (ou plusieurs) objet « line ». Vous pouvez modifier les propriétés de chaque objet pour le rendre conforme à vos goûts.

Exemples : si vous faites : t=plot(x,y), t contiendra le « handle » de la ligne créée. Vous obtenez les propriétés actuelles de cette ligne en faisant : get(t)

Si vous voulez voir quelles sont les valeurs possibles d'une propriété donnée : get(t,'nom de la propriété')

Si vous voulez modifier cette propriété : set(t,'nom de la propriété', nouvelle valeur de celle-ci)

Pour obtenir le « handle » de l'objet « axes » parent de l'objet « line » : t2=get(t,'parent')

Pour obtenir les propriétés de l'objet « axes » get(t2)

et ainsi de suite.

Bien que l'on puisse modifier les graphiques interactivement, il est souvent très profitable de procéder par commandes avec get, set,... pour sauver du temps lors de la reconstruction de figures. Également on devrait toujours créer les figures à l'aide de fonctions .m et non directement du prompt Matlab.