VI. Représentations graphiques

1. Introduction

Matlab possède un grand choix de commandes pour créer toutes sortes de représentations graphiques : graphiques standards en 2D ou 3D avec axes linéaires, semi-log ou logarithmiques, histogrammes, en escalier, en fromage, rosette, représentation de coordonnées polaires, de courbes de niveau (« contour surface »), de mailles régulières (« mesh », « grid »), parmi d'autres. Tous les objets d'un graphique peuvent être formatés pour obtenir l'apparence désirée. Texte, légendes et commentaires ainsi que des aides à la visualisation (commentaires, grilles) peuvent être facilement ajoutés. Un même graphique peut contenir plusieurs jeux de données, plusieurs graphiques peuvent être placés sur une même page, etc. Ce chapitre présente les commandes les plus importantes pour élaborer les représentations graphiques les plus communes. Il existe cependant un grand nombre de commandes et scripts pour l'élaboration de graphiques plus sophistiqués.

2. La commande « plot » et « figure »

La commande `plot` est utilisée pour créer un graphique en 2D avec axes linéaires. Il s'agit de la façon la plus simple de créer une représentation graphique dans Matlab :

```
plot (x,y)          2-D graphique avec axes linéaires
```

Les arguments x (valeurs en axis) et y (valeurs en ordonnées) sont des vecteurs (1D matrices), lesquels doivent être de même longueur. Quand la commande « plot » est exécutée, une fenêtre, appelé *figure*, apparaît automatiquement. Il est possible en outre de créer une figure pour chaque graphique élaboré à l'aide de la commande `figure`. Tous les paramètres de formatage du graphique (épaisseur des lignes, échelles, limite d'axes) sont par défaut. Ils peuvent être cependant modifiés en ajoutant des arguments (optionnels):

```
plot (x,y, 'LineSpec', 'PropertyName', 'PropertyValue')
```

LineSpec: « line specifiers » qui définissent le style et la couleur des lignes ainsi que le type de marqueurs. Ils peuvent se combiner entre eux et l'ordre n'importe pas.

Style	LineSpec	Couleur	LineSpec	Marqueur	LineSpec
solide	`-`	rouge	`r`	plus	`+`
traitillé	`--`	vert	`g`	cercle	`o`
pointillé	`:`	bleu	`b`	astérisque	`*`
point-trait	`-.`	cyan	`c`	point	`.`
		magenta	`m`	croix	`x`
		jaune	`y`	triangle	`^, >, <, v`
		noir	`k`	carré	`s`
		blanc	`w`	diamant	`d`
				étoile	`p , h`

Tab. 2 : LineSpec

PropertyName et *PropertyValue* permettent de spécifier l'épaisseur des lignes, la taille des marqueurs ainsi que leur apparence (p.ex. couleur du contour).

PropertyName	Description	PropertyValue
`LineWidth`	épaisseur de la ligne	scalair (defaut = 0.5)
`Markersize`	taille du marqueur	scalair (defaut = 10)
`MarkerEdgeColor`	couleur de la bordure du marqueur	r, g, b, … (cf. Tab. 1)
`MarkerFaceColor`	couleur du marqueur	r, g, b, … (cf. Tab. 1)

Exemple :

```
x = [-pi:0.5:pi]
y = sin(x)

figure
plot(x,y,'--b*','LineWidth',2,'MarkerSize',12)
```

2.1. Graphiques multiples

La commande `plot` permet aussi de représenter plusieurs jeux de données dans un même graphique. Les paramètres de formatages se comportent de la même façon.

```
plot (x1,y1, x2,y2,x3,y3, …, xn,yn)   → 2-D graphique avec axes linéaires
```

Exemple:

```
x1 = [-pi:0.5:pi]
y1 = sin(x);
y2 = cos(x);

figure
plot(x1,y1,'--b*',x1,y2,'--g*','LineWidth',2)
```

La commande `hold on` permet de garder une fenêtre (figure) ouverte et ainsi ajouter d'autres graphiques dans cette fenêtre. Les propriétés des axes ainsi que son format est pareil au premier graphique introduit, si rien n'est spécifié. La commande `hold off` arrête ce procédé.

```
hold on / hold off          → garder la fenêtre graphique ouverte
```

Exemple:

```
x1 = [-pi:0.5:pi]
y1 = sin(x); y2 = cos(x);

figure
plot(x1,y1,'--b*','LineWidth',2)
hold on
plot(x1,y2,'g*','LineWidth',5)
hold off
```

La commande `line` permet d'ajouter un graphique (type « ligne ») à un graphique déjà existant.

```
line (x,y,'PropertyName','PropertyValue')
                    ajouter un graphique « ligne » à une fenêtre existante
```

Elle n'a cependant pas le même *Line Specifier*; le style, la couleur ainsi que le format des marqueurs doit être spécifié :

Line Specifier	Description	PropertyValue
`LineStyle`	style de la ligne	cf. Tab 1
`Color`	couleur	cf. Tab 1
`Marker`	style du marqueur	cf. Tab 1

Exemple :

```
x1 = [-pi:0.5:pi]
y1 = sin(x);
y2 = cos(x);

figure
plot(x1,y1,'--b*','LineWidth',2)
line(x1,y2,'LineStyle','.-','color','g')
```

3. Formater un graphique

C.-à-d. spécifier le « look » et ajouter des informations dans le graphique.

3.1. la commande « legend »

Place une légende pour chaque graphique. Le nom et l'emplacement doit être spécifié :

```
legend('string1','string2',…,'stringn',position);
        → n = nombre de graphiques
```

```
legend('string1',…,'stringn','Location','location');
        → n= nombre de graphiques
```

Il existe des arguments optionnels pour choisir l'emplacement, l'orientation, etc. de la légende :

position	location specifier	Description
`-1`	`NorthEastOutside`	en dehors des axes sur la droite
`0`	`Best`	à l'intérieur des axes dans une position qui ne dérange pas le graphique
`1`	`NorthEast`	en haut à droite
`2`	`NorthWest`	en haut à gauche
`3`	`SouthWest`	en bas à droite
`4`	`SouthEast`	en bas à gauche

	orientation	Description
	`vertical`	une légende sur l'autre
	`horizontal`	à la suite

	format	Description
	`'off'`	supprime la légende
	`boxoff / boxon`	supprime/ajoute un cadre autour de la légende

Exemples

```
legend('sin(x)','cos(x)','Location','NorthWest','orientation','horizontal')
```

```
legend('sin(x)','cos(x)',1)
```

```
legend boxoff
```

```
legend 'off'
```

3.2. La commande « title »

Ajoute un titre à la fenêtre du graphique. Le format du texte peut être spécifié à l'aide d'arguments optionnels ainsi que de modifiants:

```
title('string','PropertyName',PropertyValue,...)
```

Modifiant	Description	Modifiant	Description
`\bf`	en gras	`\fontname{fontname}`	nom de la police
`\it`	en italic	`\fontsize{fontsize}`	grandeur des polices (scalair)
`\rm`	normal	`^ / _`	exposant / indice

Lettres Grecques			
`\alpha`	α	`\Phi`	Φ
`\beta`	β	`\Delta`	Δ
`\gamma`	γ	`\Gamma`	Γ
…		…	

PropertyName	Description	PropertyValue
`Rotation`	orientation du texte	scalar en °, defaut = 0°
`FontAngle`	italique ou normal	defaut = normal
`FontName`	nom des polices de caractères	ceux disponible dans Matlab
`FontSize`	grandeur des polices	defaut = 10 (scalaire)
`FontWeight`	poids des polices	light, normal (defaut), bolt
`Color`	Couleur du texte	cf. Tab 1
`BackgroundColor`	Couleur du box entourant le texte	cf. Tab 1
`EdgeColor`	Couleur du pourtour du textbox	cf. Tab 1
`LineWidth`	Largeur du pourtour du textbox	Scalaire, defaut = 0.5

Tab. 2 : arguments optionnels et modifiant

Le modifiant ne peut être appliqué qu'à une partie du titre en tapant les caractères que l'on veut modifier à l'intérieur de `{ }`.

Il est possible d'écrire le titre sur plusieurs lignes en séparant les lignes par de la façon suivante : `{'ligne 1';'ligne 2'}`.

Des valeurs numériques peuvent être introduites en les transformant au format « string » en utilisant la commande `num2str(x)`.

Exemples :

```
title('\ite^{\omega\tau} = cos(\omega\tau) + isin(\omega\tau)')
```

```
title('\fontname{Arial}exemple de titre', 'FontSize', 12, ...
       'EdgeColor', 'r')
```

```
title(['Temperature is ',num2str(c),'C'])
```

3.3. La commande « xlabel » et « ylabel »

C.-à-d. le nom des axes : `xlabel` pour l'abscisse et `ylabel` pour l'ordonnée. Le nom est donné en format texte et les formats peuvent être modifiés comme vu précédemment (cf. Tab. 2).

```
xlabel('string','PropertyName',PropertyValue,...)
```

Il est en outre possible de créer le nom des axes (pareil pour titre, légende, etc.) sur plusieurs lignes en utilisant le format cellule de caractère multilignes (cell-array) :

```
ylabel({'1er ligne' ; '2ème ligne'})
```

Exemple :

```
ylabel({'exemple de';'label'}, 'FontName', 'arial', ...
       'fontweight', 'light')
```

3.4. La commande « text »

Permet d'introduire du texte dans la fenêtre du graphique. Le côté bas gauche de la première lettre détermine la position, laquelle est donnée dans l'échelle des coordonnées (x,y) des axes. Les options de formatage restent les mêmes que précédemment.

```
text(x,y,z,'string','PropertyName',PropertyValue....)
```

Exemple

```
text(X/2, max(Z),['vitesse initiale = ' num2str(Vo,'%5.2f'), ...
       'km/h'])
```

3.5. La commande « axis », « xlim » et « ylim »

Ce sont diverses commandes permettant de définir les limites des axes du graphique.

```
axis([xmin xmax ymin ymax],'option'); → option: square, fill, tight, image...

xlim([xmin xmax]);
xlim('option');                          → option = mode, auto, manual
```

3.6. La commande « box », « grid » et « colorbar »

La commande `grid on` permet d'ajouter une grille (réseau de lignes) en arrière plan. Quant à la commande `box off`, elle permet de supprimer le cadre du graphique.

La commande `colorbar` permet de créer une légende sous forme de barre de couleur. Plusieurs arguments optionnels existent pour contrôler les couleurs (cf. `caxis` dans Matlab help)

3.7. Créer une annotation dans la fenêtre graphique

La commande `annotation` permet de créer différents objets d'annotation, comportant aussi bien du texte que des objets graphiques. L'objet créé est placé dans la fenêtre graphique à l'aide de coordonnées (x,y), correspondant aux extrémités de l'objet (sous forme vectorielle *x = [x1,x2]* et *y = [y2 y2]*. Quand la largeur et hauteur de l'objet est donné, les coordonnées sont donnés dans un simple vecteur (x,y) correspondant au coin du bas gauche de l'objet.

annotation	Description	Commande
`line`	crée un objet ligne définit entre le point (x_1,y_1) et (x_2,y_2)	`annotation('line',x,y)`
`arrow`	crée une flèche	`annotation('arrow',x,y)`
`doublearrow`	crée une double flèche	`annotation('doublearrow',x,y)`
`textarrow`	crée une flèche attachée à un « text box »	`annotation('textarrow',x,y)`
`ellipse`	crée une ellipse de largeur *w* et hauteur *h*	`annotation('ellipse',[x y w h])`
`rectangle`	crée un rectangle de largeur *w* et hauteur *h*	`annotation('rectangle',[x y w h])`

4. Types de représentations graphiques

(Se référer à Matlab help pour les différents arguments optionnels de chaque type)

Type	Description	commande
`semilogy`	axe des y en log de base 10 et axe des x linéaire	`semilogy(x,y)`
`semilogx`	axe des x en log et axes des y linéaire	`semilogx(x,f(x))`
`loglog`	les deux axes sont en log de base 10	`loglog(x,y)`
`errorbar`	graphique avec bar d'erreur en y sur chaque valeur	`errorbar(x,y,e);` e vecteur erreur en chaque point de x. `errorbar(x,y,e^up,e^down);` e^{up} étant la limite supérieure de l'erreur et e^{down} la limite inférieure.
`bar` `barh`	graphique à bars verticales ou horizontales	`bar(x,y)` `barh(x,y)`
`hist`	histogramme	`hist(y,nbins);` nbins = nbre de barreaux `hist(y,x) ;` x = location du centre du barreau
`stairs`	graphique en escalier	`stairs(x,y,LineSpec)`
`stem`	graphique de données discrètes	`stem(x,y,LineSpec)`
`pie`	en fromage	`pie(x,labels)`
`polar`	graphique à coord. polaires	`polar(theta, radius,'lineSpec')`

Voir aussi : `patch`, `light`, `surface`.

4.1. La commande « axes »

Permet de créer les axes des objets graphiques. Sa position est donnée par les coordonnées (en relatif, c.-à-d. entre [0 1]) des quatre coins de la boîte contenant le graphique dans la fenêtre.

```
axes('position',[.1   .1   .8   .6])
```

5. Représenter plusieurs graphiques dans la même fenêtre

Plusieurs graphiques peuvent être mis l'un à côté de l'autre à l'aide de la commande `subplot`. Cette commande divise la fenêtre (figure) en *m x n* sous-fenêtres rectangulaires contenant chacun un graphique. Les sous-fenêtres p sont numérotées de 1 à m x n, celui en haut à gauche étant le n° 1 et celui en bas à droite étant le n° m x n. Les attributs de formatage et options s'applique à chaque graphique individuellement.

```
subplot(m,n,p) ;     m = nbre de ligne, n=nbre de colonne, p = no du graphique
```

Exemple:

```
figure
subplot(2,2,1)
plot(x,y1,'--b*','LineWidth',2)
hold on
subplot(2,2,4)
plot(x,y2,'--g*','lineWidth',5)
```

5.1. Graphique 2D comportant deux axes linéaires et indépendant

La commande `plotyy` permet de représenter un graphique avec deux ordonnés indépendantes. L'axe de gauche y_1 se réfère à l'axis x_1 et l'axe de droite y_2 se réfère à l'axis x_2, x_1 et x_2 étant représentés sur le même axis.

Exemple :

```
figure;
hold on
[AX,H1,H2]=plotyy(x1,y1,x2,y2);
```

Les propriétés `XAxisLocation` and `YAxisLocation` permettent de spécifier de quel côté du graphique sont placés les axes x et y. Ces propriétés sont très utiles quand il s'agit de représenter plusieurs jeux de données avec des axes d'échelles différentes dans une même fenêtre (cf. exemple plus bas)

6. Manipulation des objets dans un graphique

Matlab possède des commandes servant à sélectionner les objets contenus dans la fenêtre graphique (les figures, les axes, etc…) afin de pouvoir les manipuler séparément. Il est important de bien comprendre la distinction entre figure (la fenêtre des graphiques), les graphiques et leurs objets.

6.1. les commandes « set », « get » et « findobj »

La commande `set` permet de fixer les attributs de formatages et autres options de chaque objet dans un graphique:

```
set(fig,'PropertyName', PropertyValue,…)
```

D'autres attributs peuvent ainsi être ajoutés, par ex. grâce à la commande `XTick` et `XtickLabel` :

```
set(gca,'XTick',-pi:pi/2:pi)
set(gca,'XTickLabel',{'-pi','-pi/2','0','pi/2','pi'})
```

La commande `get` permet de questionner les propriétés et attributs des objets du graphique et de la fenêtre, lesquelles peuvent être attribués dans une variable de type *structure*.

```
a = get(h,'PropertyName')
```

La commande `findobj` permet de localiser les propriétés spécifiques des objets graphiques. Il est possible de limiter la recherche en introduisant des propriétés ou en spécifiant une branche de la hiérarchie.

```
h = findobj('PropertyName',PropertyValue,...)
```

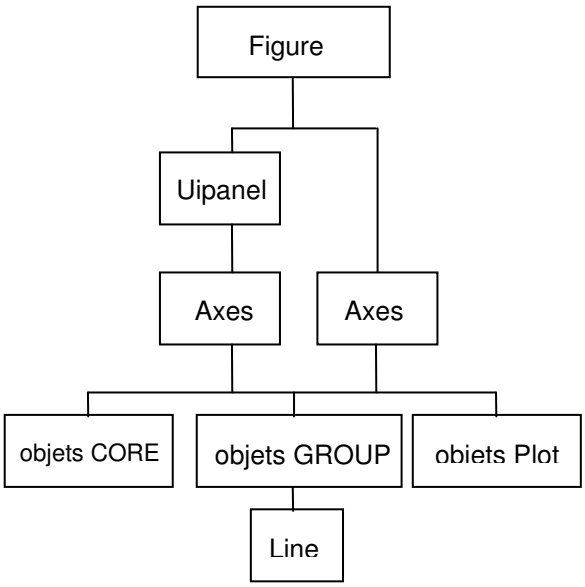
Il est communément utilisé pour effectuer des changement dans une représentation graphique :

```
set(findobj('Type','line'),'Color','r');  change toutes les lignes en rouge
```

En effet, les objets sont attribués selon une hiérarchie d'objet « parent » et « enfant » :

- parent: l'élément parent des objets graphiques est la figure (fenêtre). La commande gcf retourne la manipulation des objets graphiques parents en cours.
- children: les éléments enfants d'un objet graphique, c-à-d un vecteur contenant la manipulation des axes ainsi que les objets de l'interface graphique.

Exemple : hiérarchie des objets (après [1])




```
set(get(gcf,'children'),'ycolor',[0 0 1]);
                → fixe en bleu la ligne des axes, les ticks, les labels, etc...
```

6.2. Les commande « gcf » et « gca »

Les commandes `gcf` (get current figure) et `gca` (get current axes) s'apparente à la commande `get` et permettent de directement manipuler soit la figure en cours, soit tous les objets graphiques dan la fenêtre en cours.

```
h = gcf
set(gca,'Color',[1 1 1]);
```

6.3. La commande « clf » et « cla » ; « close » « refresh »

Commande	Description	Exemple
`clf`	efface tous les graphiques dans la fenêtre en cours	`clf('reset')`
	efface le graphique (fig) mentionné	`clf(fig)`
`close`	efface la/ toutes les figure(s)	`close fig` `close all`
`refresh`	redessine la figure courante	`refresh` `refresh(fig)`
`cla`	efface des axes en cours tous ses objets graphiques	`cla reset` `cla(ax)`

Exemple général tiré de [1] pour générer un graphique à axes doubles !

```
%ouvrir une fenêtre
figure;

%maintenir la fenêtre ouverte
hold on

%afficher le graphique en l'attribuant dans une variable
hl1 = plot(x1,y1,'-r');

%attribuer les axes du graphique dans une variable
ax1 = gca;

%fixer les couleurs aux axes
set(ax1,'XColor','r','YColor','r');

%créer deux nouveaux axes en haut et à droite de la fenêtre
ax2 = axes('Position',get(ax1,'Position'),...
    'XAxisLocation','top',...
    'YAxisLocation', 'right',...
    'Color','none',...
    'XColor','b','Ycolor','k');

%ajouter un second graphique avec les axes définis précédemment
hl2 = line(x2,y2,'Color','c','Parent',ax2);
```

7. Représentation graphique en 3D

(Se référer à Matlab help pour les différents arguments optionnels de chaque type)

Type	Description	commande
`plot3`	Graphique type ligne en 3D	`plot3(x,y,z,'style')`
`mesh`	Graphique type mailles en 3D	`mesh(x,y,z)`
`surf`	Graphique type surface en 3D	`surf(x,y,z)`
`fill3`	Graphique type polygone 3D	`fill3(x,y,z,ColorSpec)`

Voir aussi : `zlabel`.

8. Références

[1] MATLAB Help

[2] Amos, Gilat, 2007. *Matlab, an introduction with application,* John Willey and Sohn, Inc.