



INF1005A: Programmation procedurale

Chapitre 9: Graphiques MATLAB



Agenda

- Traçage de graphiques
- Graphiques – Ajouter des propriétés
- Graphiques – fonctions utiles
- Graphiques – exporter un graphique
- Différents graphiques
- Quelques graphiques intéressants
- Rappel de fonctions

Traçage de graphiques exemple

```

%% * MISE A JOUR: 2014/03/30
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% preparation des donnees
x1=0:0.2:12; % x1 prends valeur
x2=0:0.3:15; % x2 prends valeur
x3=0:0.4:20; % x3 prends valeur

y1=besselj(1,x1);
y2=bessely(2,x2);
y3=besselh(3,x3);

figure(1);
subplot(2,2,1) %positionnement

h=plot(x1,y1,x2,y2,x3,y3);
set(h, 'LineWidth', 2, {'LineStyle','Color'},{'r';'m';'b'});

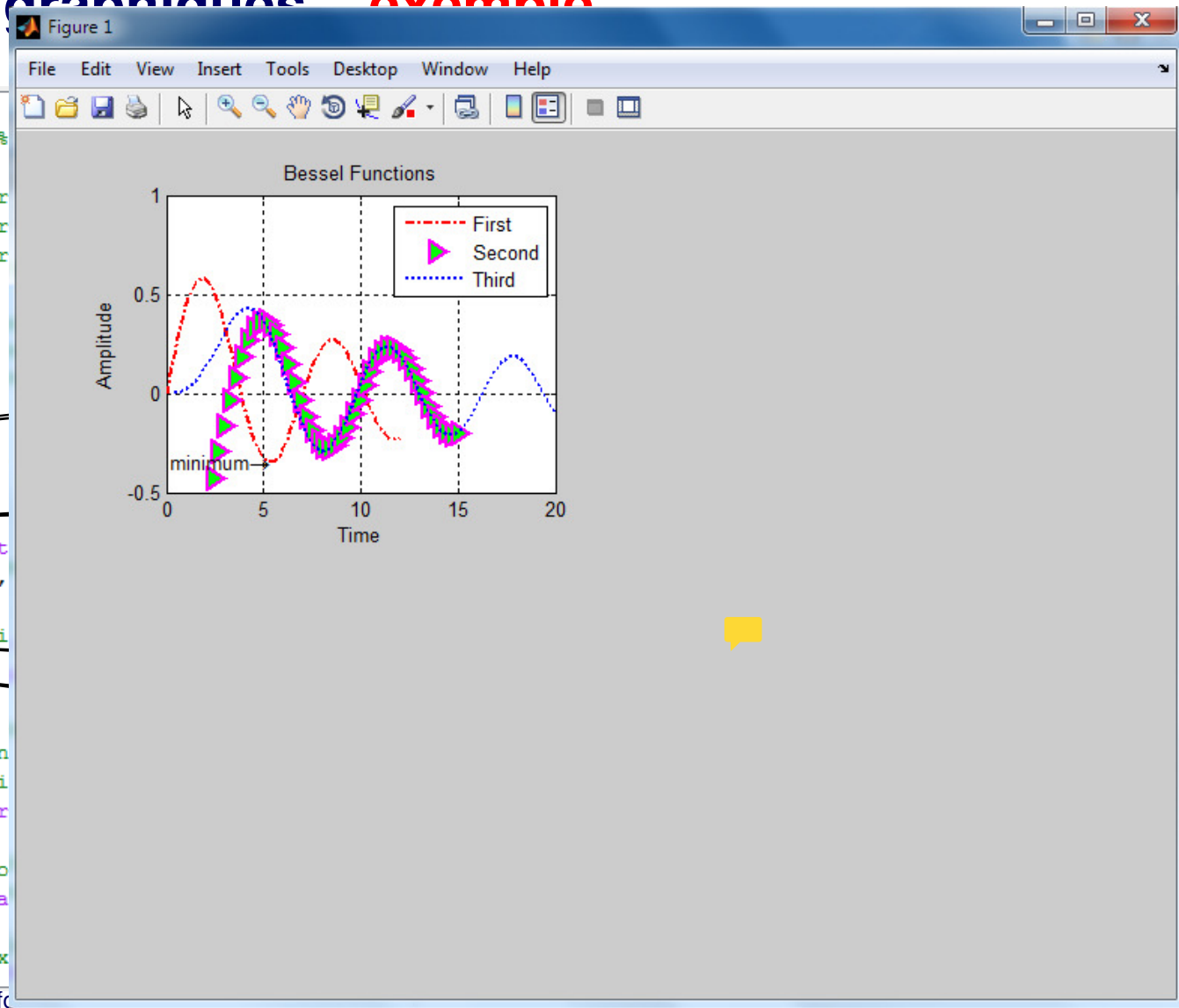
axis ([0 inf -0.5 1]); % defini
grid on; % grille visible

xlabel('Time'); % notation axe
ylabel('Amplitude'); % notation
title('Bessel Functions'); % ti
legend(h, 'First', 'Second', 'Thir

[y,ix]=min(y1); % un point impo
text(x1(ix), y, 'minimum\righta

print -djpeg -r500 dessin; % ex
  
```

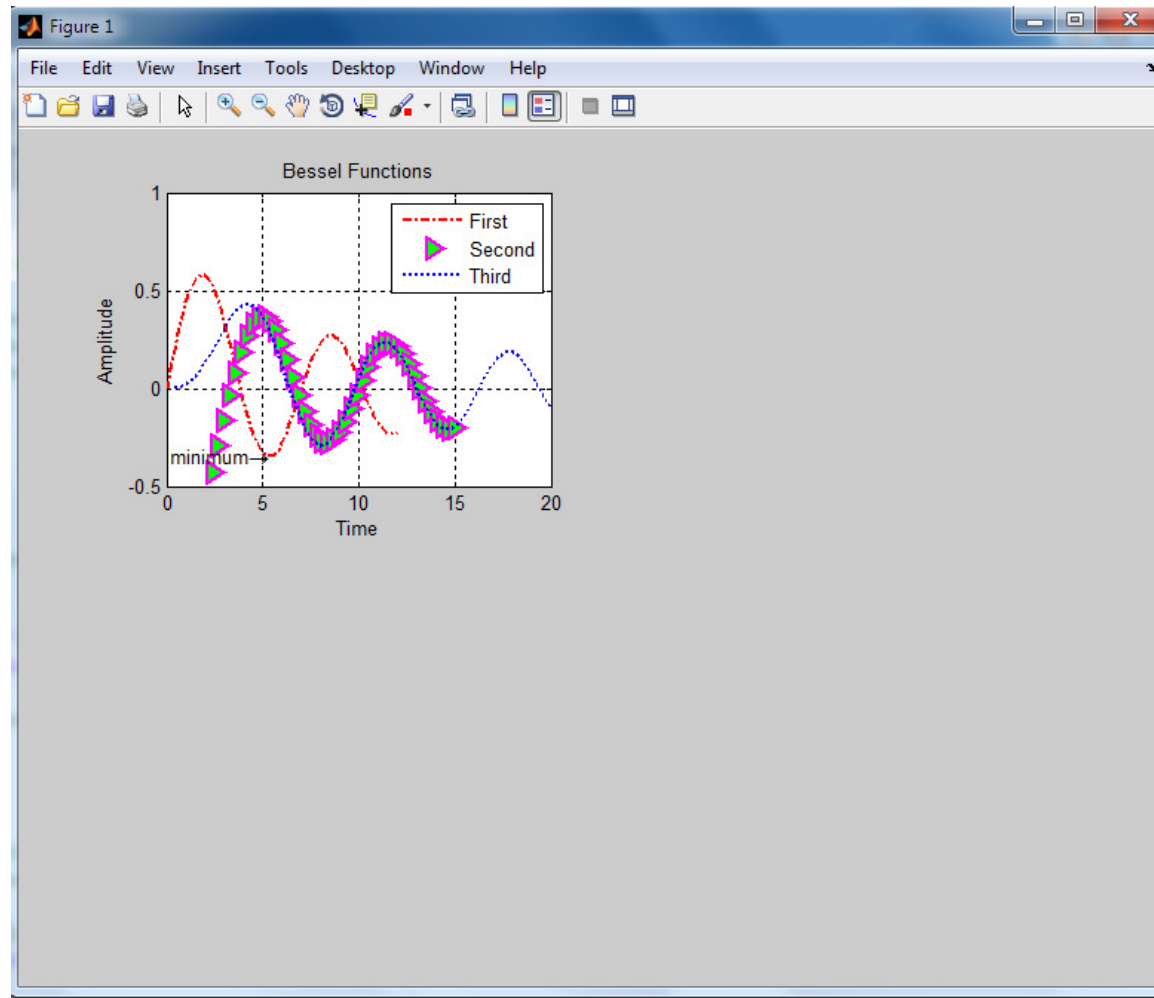
* Génie info



Exporter graphique



Traçage de graphiques – exemple

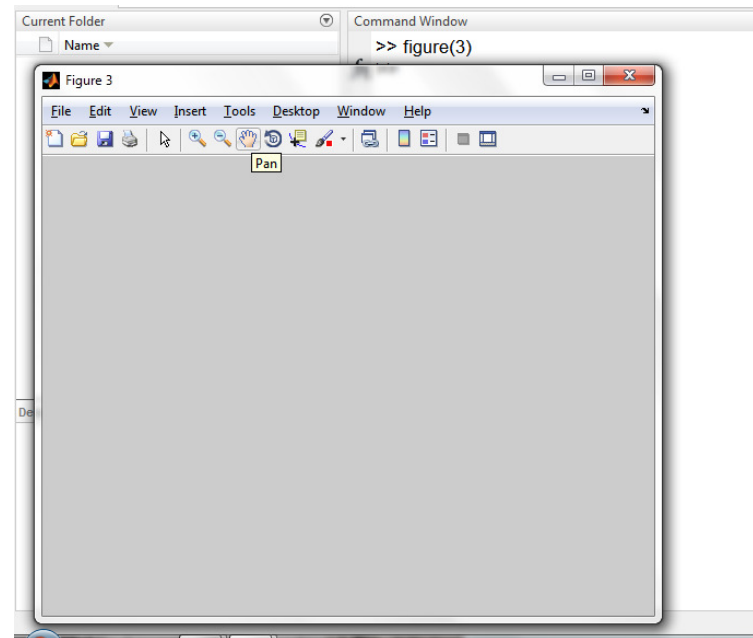




Traçage de graphiques— **choix de la fenêtre**

`figure (no_de_figure)`

- la fonction permet de créer une nouvelle fenêtre.
- possible de donner un numéro à la fenêtre.
- utile lors du traçage de plusieurs graphiques.



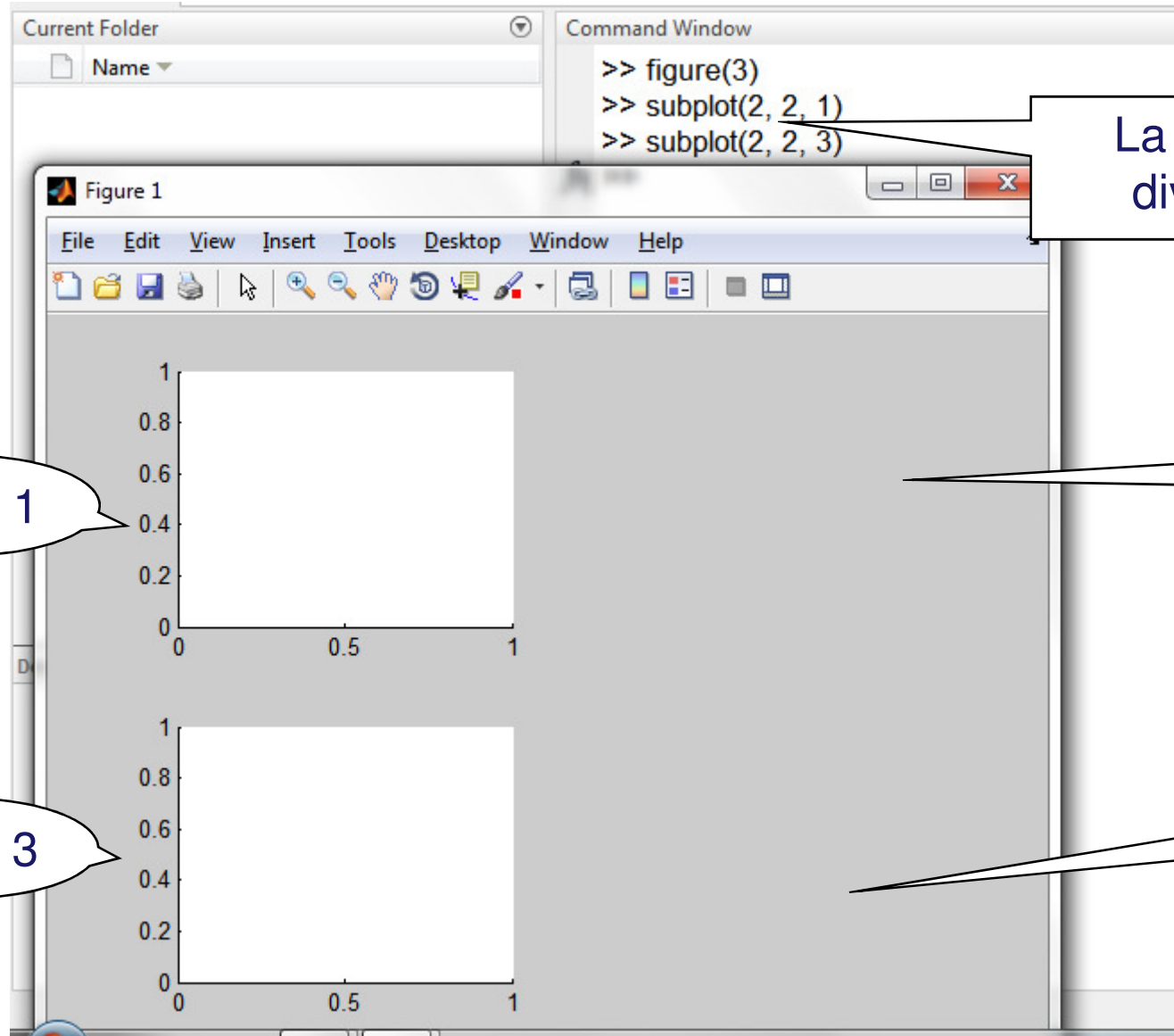


Traçage de graphiques – positionnement de la fenêtre

`subplot(ligne, colonne, position)`

- `ligne` : nombre de lignes que contiendra la matrice qui découpe la fenêtre.
- `colonne` : nombre de colonnes que contiendra la matrice qui découpe la fenêtre.
- `position` : entier qui indique dans quelle case de la matrice le graphique sera inséré. Les cases se comptent de gauche à droite et de haut en bas.
- cette fonction permet d'insérer plusieurs graphiques à l'intérieur d'une même fenêtre.
- la fenêtre est décomposée en une matrice de dimension voulue et le graphique est tracé dans la case demandée.

Traçage de graphiques – positionnement de la fenêtre



La fenêtre est
divisée en 4

Case 1

Case 2

Case 3

Case 4

Traçage de graphiques – positionnement de la fenêtre

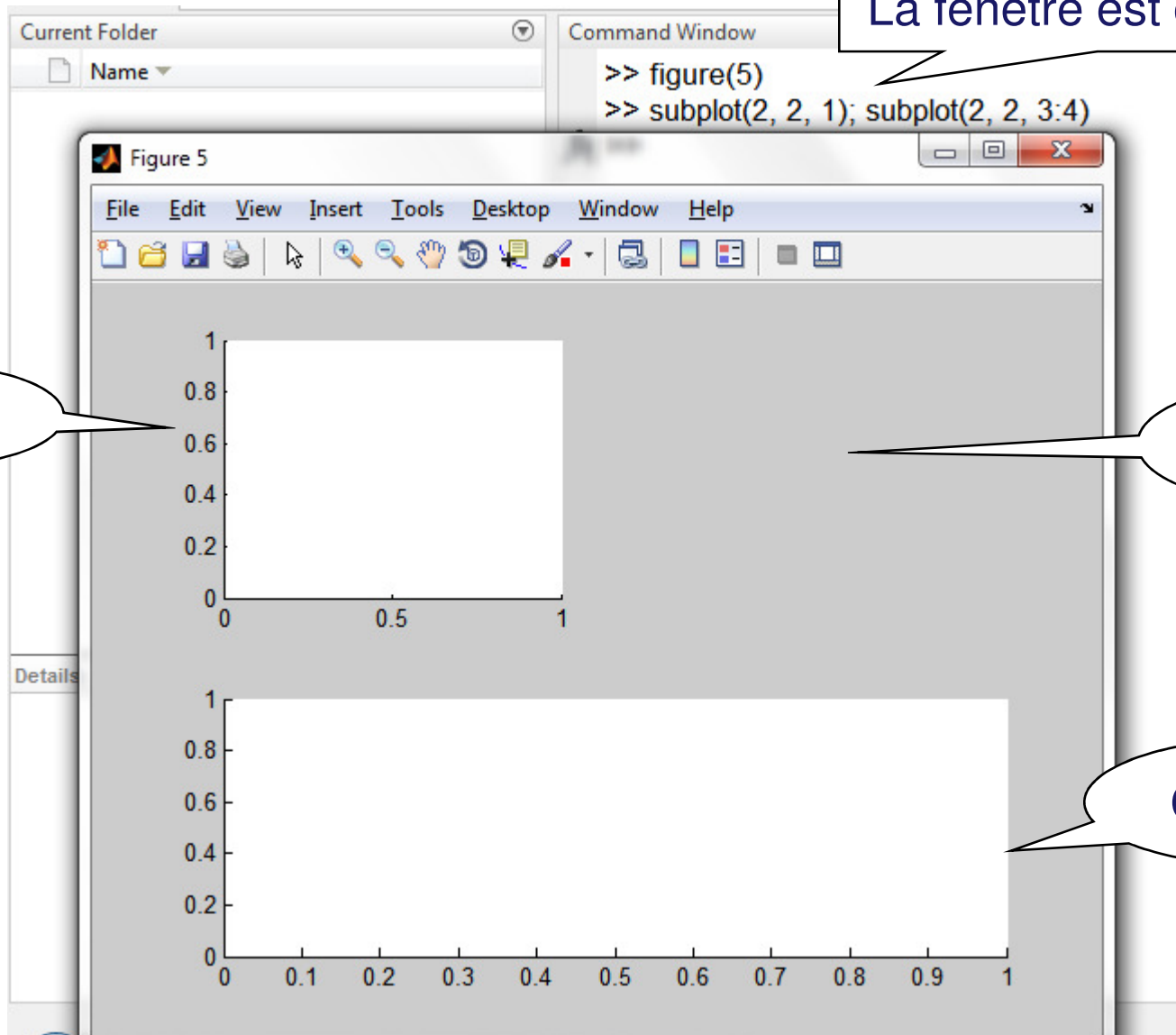
La fenêtre est divisée en 4

```
>> figure(5)  
>> subplot(2, 2, 1); subplot(2, 2, 3:4)
```

Case 1

Case 2

Cases 3:4



Traçage de graphiques – **fonction de traçage**

`plot(X1,Y1,'format_ligne1',X2,Y2,'format_ligne2'...)`

X_n : Vecteur contenant les coordonnées en X de la ne série de données.

Y_n : Vecteur contenant les coordonnées en Y de la ne série de données.

`format_lignen` : Différentes options qui définissent le format de la ligne (son épaisseur, sa couleur, etc..).

- Fonction principale de traçage à laquelle il faut transmettre les données à tracer et le format voulu pour la trace.

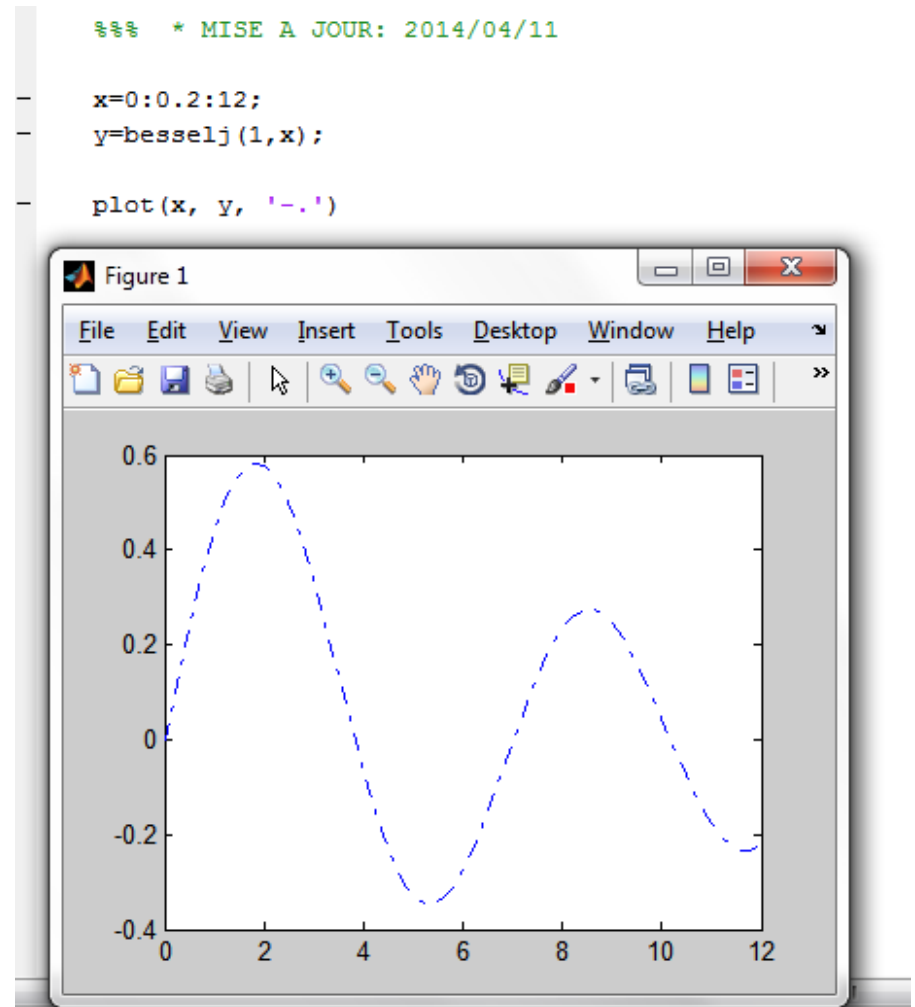
Traçage de graphiques – **format des lignes**

Spécifier les attributs des traces :

- le **style de la ligne** :

i.e. `plot(x, y, '-.')`

Caractère	Style
-	Ligne pleine (par défaut)
--	Ligne en trait tireté
:	Ligne pointillée
-.	Ligne en trait tireté et pointillée



Traçage de graphiques – **format des lignes**

Spécifier les attributs des traces :

- l'épaisseur de la ligne ('linewidth'):

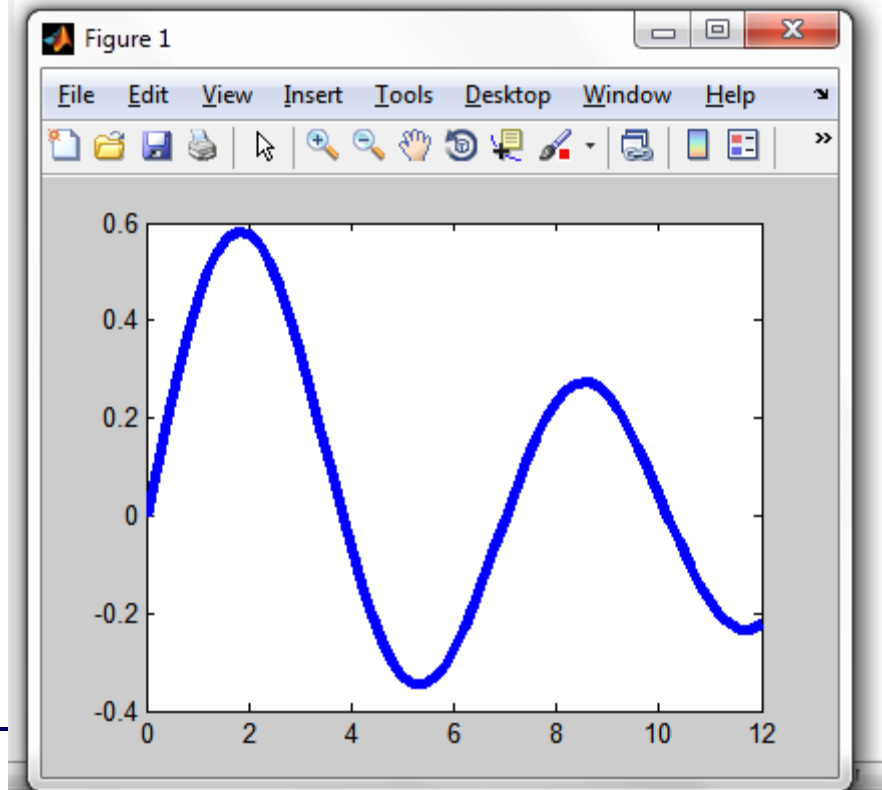
i.e. `plot(x,y, 'linewidth', 4)`

```
clear all; clc

%% * MISE A JOUR: 2014/04/11

x=0:0.2:12;
y=besselj(1,x);

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



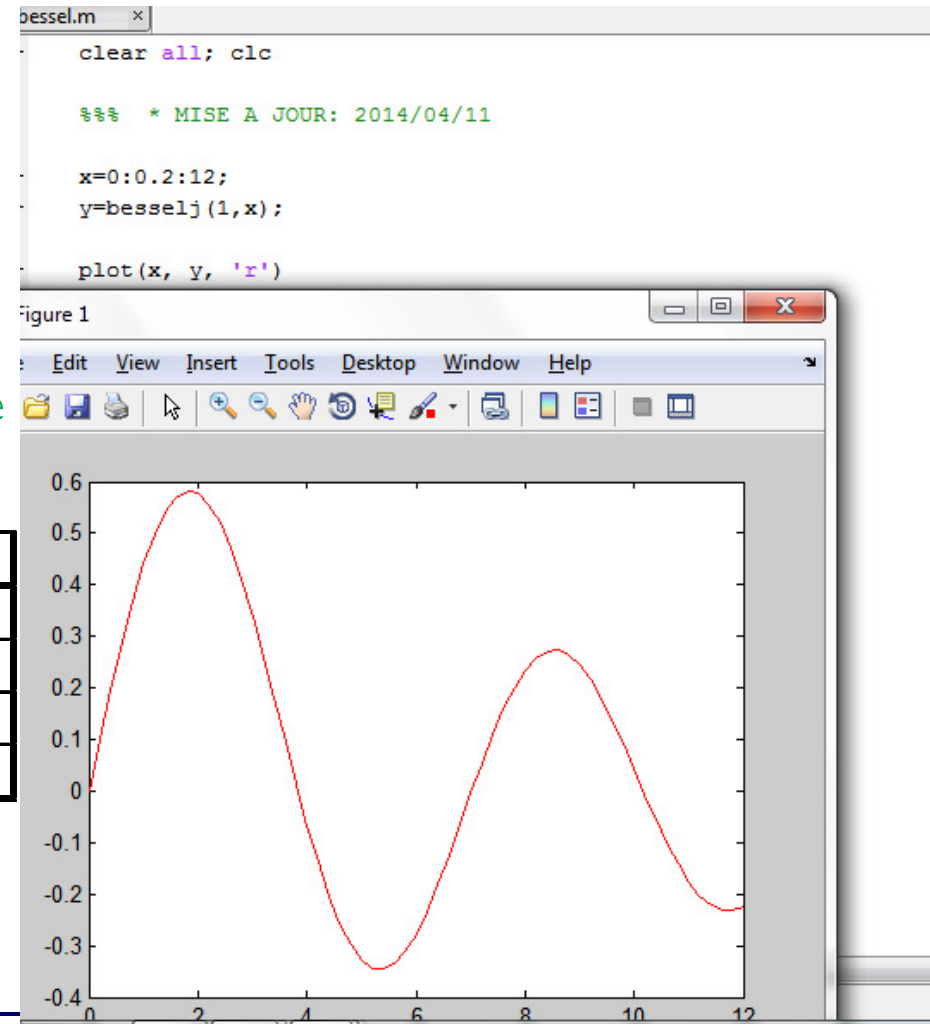
Traçage de graphiques – **format des lignes**

Spécifier les attributs des traces :

- la **couleur de la ligne** :

i.e. `plot(x,y, 'r')` % rouge

Caractère	Couleur	Caractère	Couleur
r	rouge	m	magenta
g	vert	y	jaune
b	bleu	k	noir
c	cyan	w	blanc

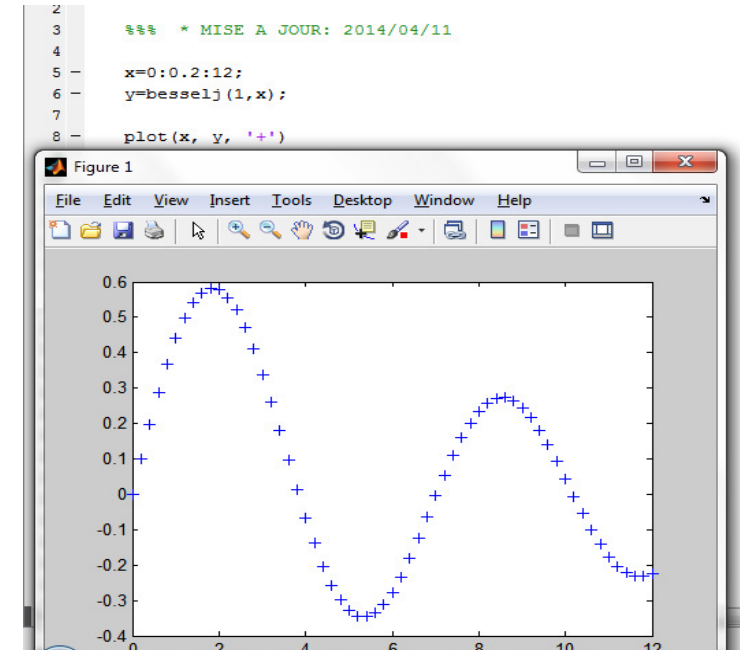


Traçage de graphiques – **format des lignes**

Spécifier les attributs des traces :

- le **type de marqueur** :

i.e. `plot(x,y, '+')`



Caractère	Type de marqueur	Caractère	Type de marqueur
+	Symbole plus	^	Triangle pointant par en haut
o	Cercle	v	Triangle pointant par en bas
*	Astérisque	>	Triangle pointant vers la droite
.	Point	<	Triangle pointant vers la gauche
x	Croix	p	Pentagone
s	Carré	h	Hexagone
d	Losange		

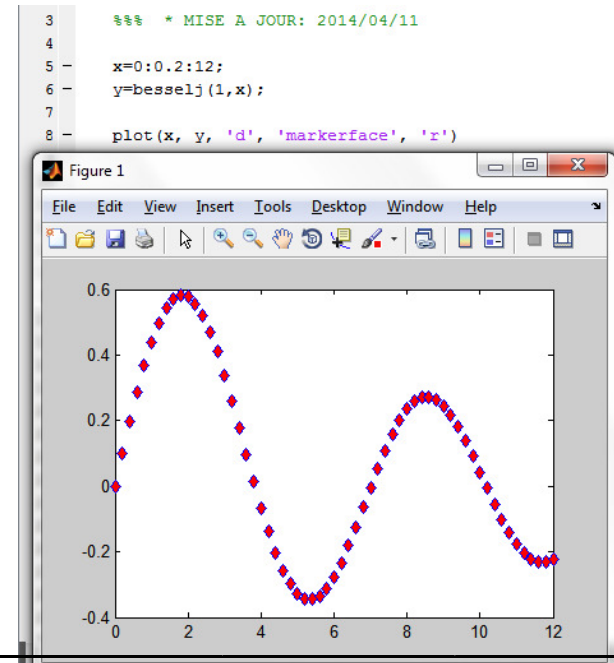
 Marqueurs avec contours

Traçage de graphiques – **format des lignes**

Spécifier les attributs des traces :

- le **couleur de la face de marqueur**
(ceux qui ont des contours) 'markerface':

i.e. `plot(x,y, 'd', 'markerface', 'r')`



Caractère	Type de marqueur	Caractère	Type de marqueur
+	Symbole plus	^	Triangle pointant par en haut
o	Cercle	v	Triangle pointant par en bas
*	Astérisque	>	Triangle pointant vers la droite
.	Point	<	Triangle pointant vers la gauche
x	Croix	p	Pentagone
s	Carré	h	Hexagone
d	Losange		

 Marqueurs avec contours

Traçage de graphiques – **format des lignes**

Spécifier les attributs des traces :

- le couleur du contour des marqueurs (pas tous les marqueurs) :

i.e.

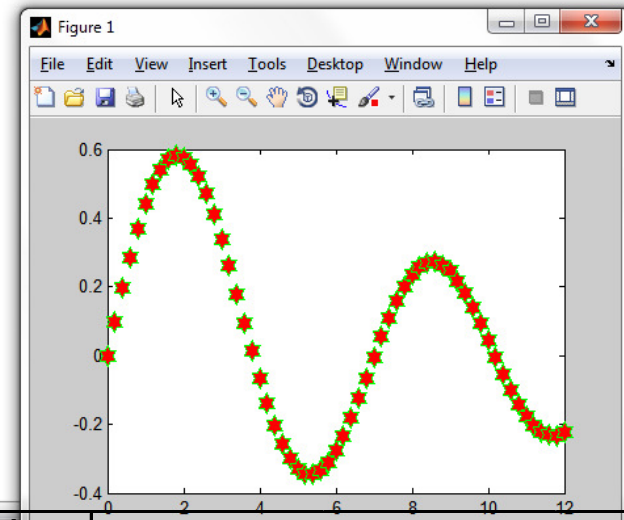
~~`plot(x,y, 'd', 'markeredge', 'g')`~~

`plot(x,y, 'd', 'markersize', 12,
'markerfacecolor', 'r',
'markeredge', 'g')`

```
%% * MISE A JOUR: 2014/04/11
```

```
x=0:0.2:12;  
y=besselj(1,x);
```

```
plot(x, y, 'h', 'markersize', 12, 'markerfacecolor', 'r', 'markeredge', 'g')
```



Caractère	Type de marqueur	Caractère	Type de marqueur
+	Symbole plus	^	Triangle pointant par en haut
o	Cercle	v	Triangle pointant par en bas
*	Astérisque	>	Triangle pointant vers la droite
.	Point	<	Triangle pointant vers la gauche
x	Croix	p	Pentagone
s	Carré	h	Hexagone
d	Losange		



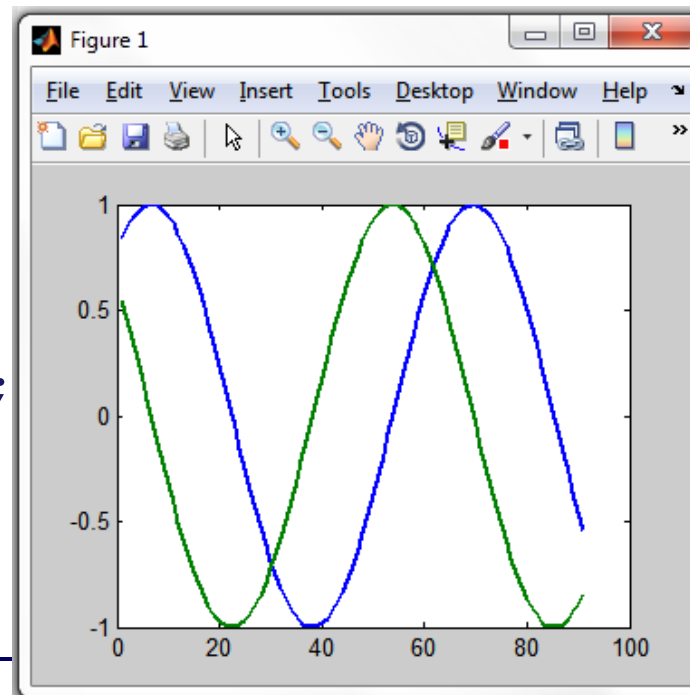
Marqueurs avec contours

Traçage de graphiques – **matrices**

- Soit Y est une matrice composée de plusieurs colonnes et de plusieurs lignes, MATLAB tracera une courbe pour chaque colonne de la matrice
- Si aucun vecteur de données en X n'est spécifiée, MATLAB générera un vecteur X de 1 à m où m est le nombre de rangée de Y .

i.e.

```
n = 1:0.1:10;  
A(:,1)=sin(n);  
A(:,2)=cos(n);  
plot(A, 'linewidth', 2);
```





Agenda



Traçage de graphiques

- Graphiques – Ajouter des propriétés
- Graphiques – fonctions utiles
- Graphiques – exporter un graphique
- Différents graphiques
- Quelques graphiques intéressants
- Rappel de fonctions

Ajouter des propriétés – `set()`

– `set()` – fonction utile si certaines propriétés ont été omises lors de la définition du graphique:

i.e.

```
nograph = plot(X1,Y1,'r')
```

```
set(nograph, 'nom_propriété1', valeur_propriété1, ...)
```

`nograph` –  numéro du graphique - généré lors de sa création.

- la définition des propriétés est semblable à la fonction `plot()`.
- les éléments des différentes traces doivent être séparés par un point virgule (;) et mises entre accolades { }.

i.e. `set(h, {'Color'}, {'r'; 'g'; 'b'});`

Ajouter des propriétés – `axis()`

`axis([xmin xmax ymin ymax])`

– fonction qui permet de spécifier les limites des axes.

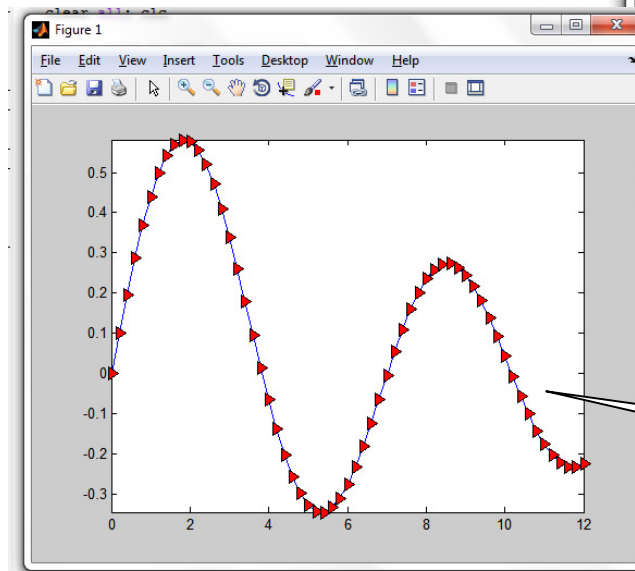
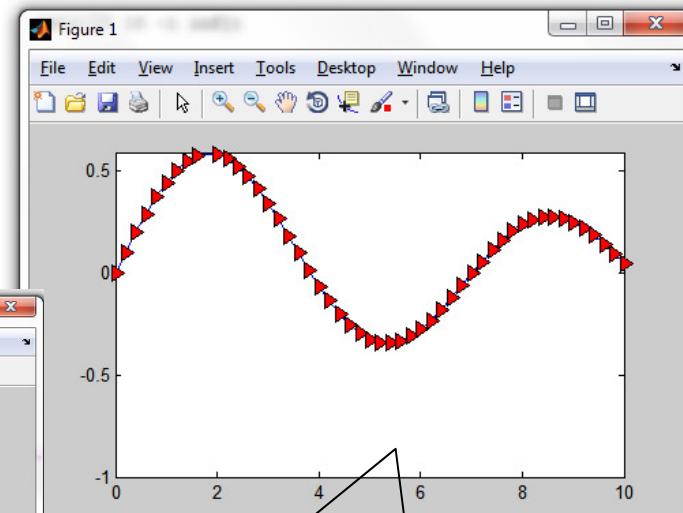
- lorsque le maximum ou le minimum d'une fonction n'est pas connu, l'utilisation de `inf` et `-inf` permet à MATLAB de déterminer lui-même la limite voulue.

```

x1=0:0.2:12; % x1
y1=besselj(1,x1); % y1

figure;
h=plot(x1,y1,'->','markersize',8,'markerfacecolor','r','markeredgecolor','k')

```



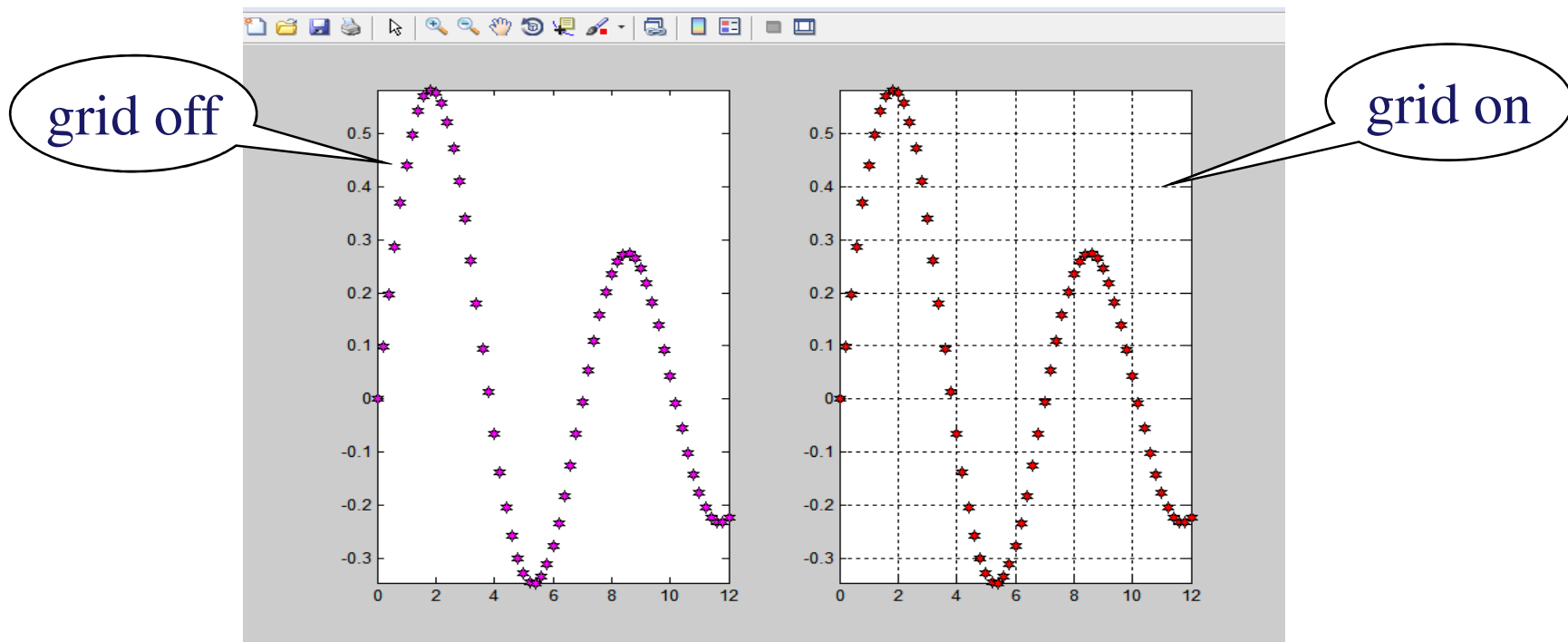
`axis([0 10 -2 inf]);`

`axis('tight')`

Ajouter des propriétés – **grid()**

`grid on` / `grid off` ou `grid ('on')` / `grid ('off')`

– fonction qui permet d'activer ou de désactiver l'affichage du quadrillage





Ajouter des propriétés – **annotation du graphique – xlabel(), ylabel(), title()**

```
xlabel('titre_axe_des_x')  
ylabel('titre_axe_des_y')
```

– fonctions qui permettent de donner un **titre aux axes** du graphique (la chaîne de caractères entre parenthèses)

```
title('titre_du_graphique_y')
```

– fonction qui permet de donner un **titre au graphique**
- semblable aux `xlabel()` et `ylabel()`



Ajouter des propriétés – **annotation du graphique – text ()**

```
text(x, y, 'commentaire', 'format_commentaire')
```

– fonction qui permet d'ajouter des textes explicatifs sur le graphique

x : coordonnée en x du début des commentaires.

y : coordonnée en y du début des commentaires.

commentaire : chaîne de caractères.

format_commentaire : définition de certains paramètres textuels comme l'alignement (gauche, droite, centré).

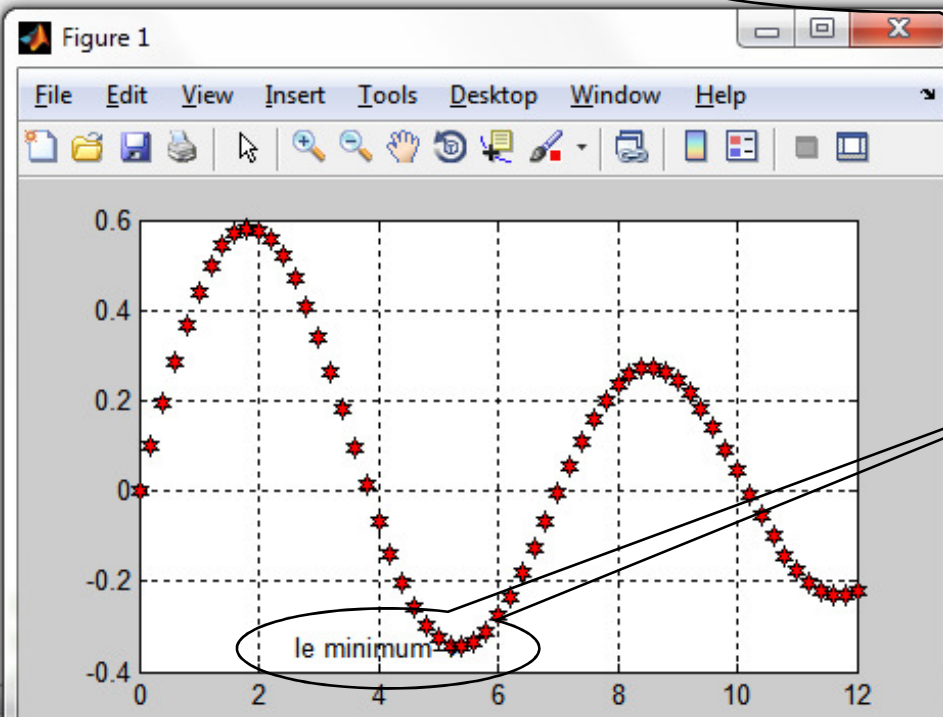
Ajouter des propriétés – annotation du graphique – text ()

i.e.

```
x1=0:0.2:12; % x1
y1=besselj(1,x1); % y1

figure(1);
h=plot(x1,y1,'h','markersize', 8, 'markerfacecolor', 'r', 'markeredgecolor','k')
grid on
axis([0 12 -0.4 0.6])

[y,ix]=min(y1); % un point important sur le graphique, le min de y1
text(x1(ix), y, 'le minimum\rightarrow', 'HorizontalAlignment','right')
```



aligné à droite

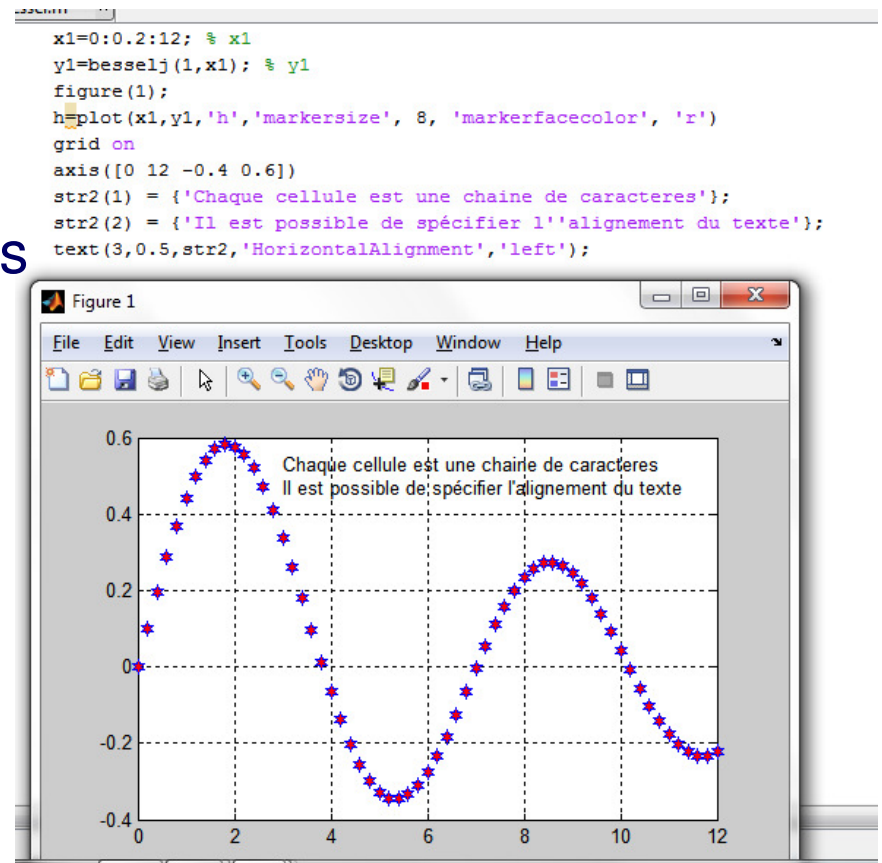
Ajouter des propriétés – annotation du graphique – `text()`

- il est possible d'insérer plusieurs lignes de commentaires avec la fonction `text()`
- il suffit de créer un ensemble de cellules contenant les différentes lignes de texte.

i.e.


```
str2(1) = {'Chaque cellule est une  
chaîne de caractères'};  
str2(2) = {'Il est possible de  
spécifier l'alignement du texte'};
```

```
text(3, 0.5, str2, 'HorizontalAlignment', 'left');
```



Ajouter des propriétés – **annotation du graphique – legend()**

```
legend('chaîne1', 'chaîne2', . . . position)
```

- fonction qui permet d'ajouter une légende à un graphique
- la légende est composée des chaînes de caractères fournies dans l'appel de la fonction ainsi que d'un échantillon du type de ligne associée à chaque description. Le tout est encadré.
- ~~position~~: permet de spécifier l'endroit où la légende sera affichée: 

Position	Description
-1	À l'extérieur du graphique, à droite.
0	À l'intérieur du graphique, cachant le moins de point possible.
1	À l'intérieur du graphique, en haut à droite (par défaut).
2	À l'intérieur du graphique, en haut à gauche.
3	À l'intérieur du graphique, en bas à gauche.
4	À l'intérieur du graphique, en bas à droite.

Ajouter des propriétés – **annotation du graphique – legend()**

```
legend('chaîne1', 'chaîne2', . . . position)
```

- il est possible de donner des descriptions composées de plusieurs lignes - le caractère spécial `\newline`

i.e.

```
legend('Ceci est la \newline courbe pour sin(x)', 'cos', 1)
```

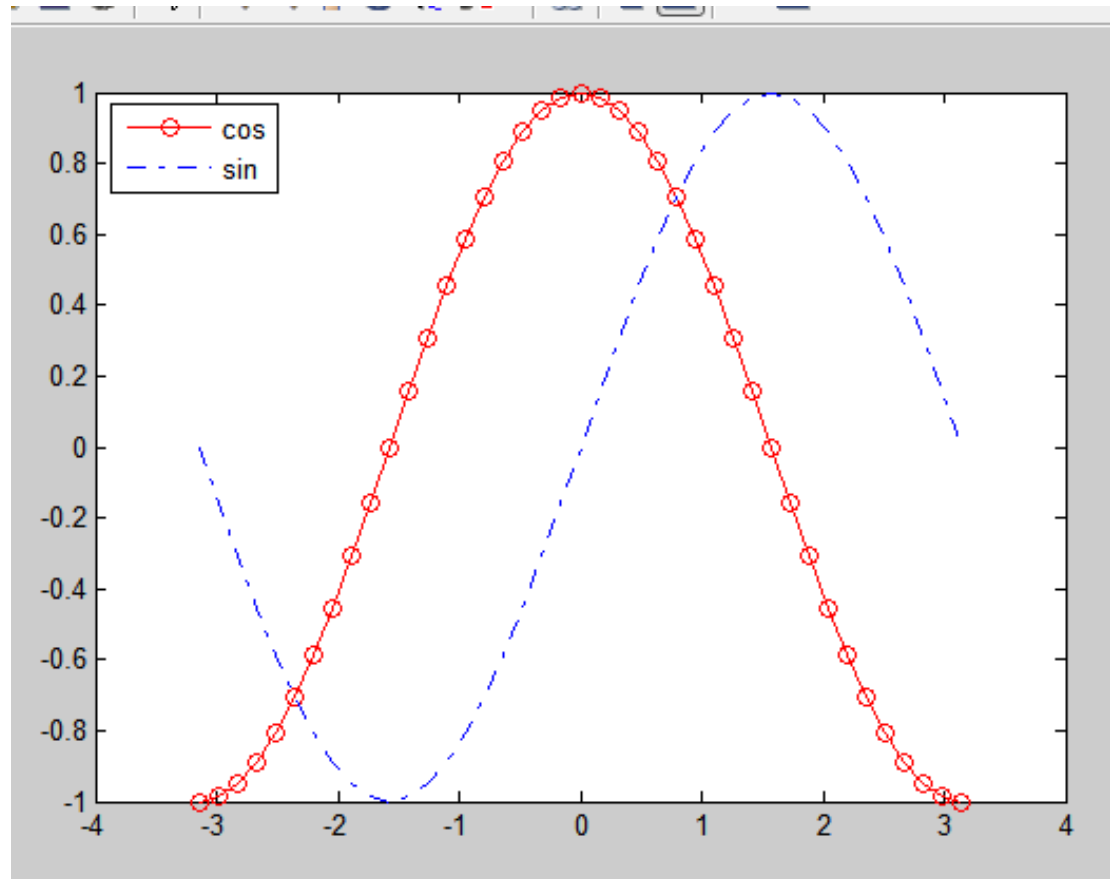
et pour ne pas avoir un espace avant 'courbe' :

```
legend('Ceci est la \newlinecourbe pour sin(x)', 'cos', 1)
```

- il est possible d'enlever ou d'ajouter différentes options à la légende. Pour connaître une liste de ces options, se référer à l'aide de MATLAB.

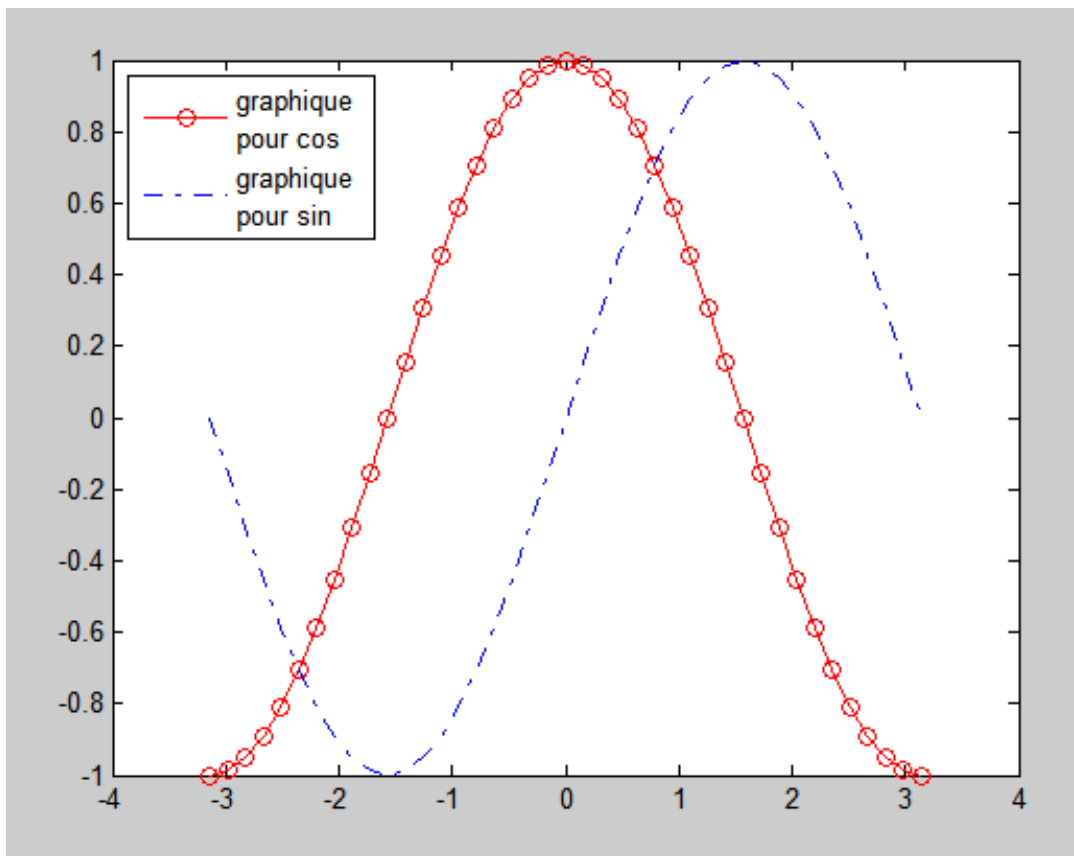
Ajouter des propriétés – **annotation du graphique – legend()**

```
x = -pi:pi/20:pi;  
plot(x,cos(x),'-ro',x,sin(x),'-.b');  
legend('cos','sin',2);
```



Ajouter des propriétés – annotation du graphique – `legend()`

```
x = -pi:pi/20:pi;  
plot(x,cos(x),'-ro',x,sin(x),'-.b');  
legend('graphique \newlinepour cos', 'graphique \newlinepour sin', 2)
```





Agenda



Traçage de graphiques



Graphiques – Ajouter des propriétés

- Graphiques – fonctions utiles
- Graphiques – exporter un graphique
- Différents graphiques
- Quelques graphiques intéressants
- Rappel de fonctions

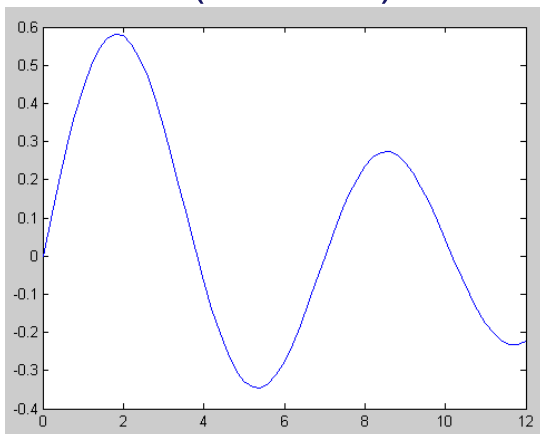


Fonctions utiles – **hold()**

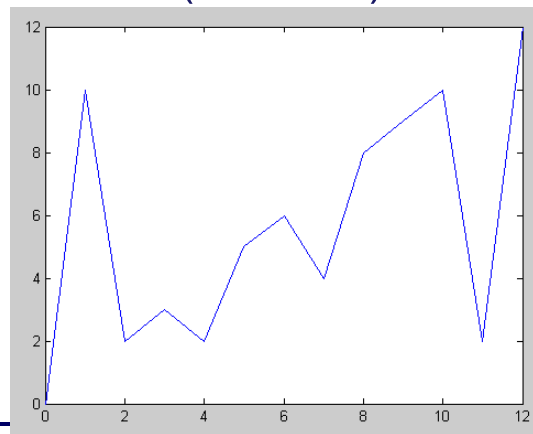
`hold on` / `hold off` ou `hold ('on')` / `hold ('off')`

- fonction qui **ajoute** le (les) prochain(s) graphique(s) **au graphique déjà existant** – `hold on`. Lorsque la fonction est désactivée (`off`) le graphique existant est remplacé par le nouveau.-
- il est important de désactiver la fonction (`hold off`) après le traçage de graphiques.

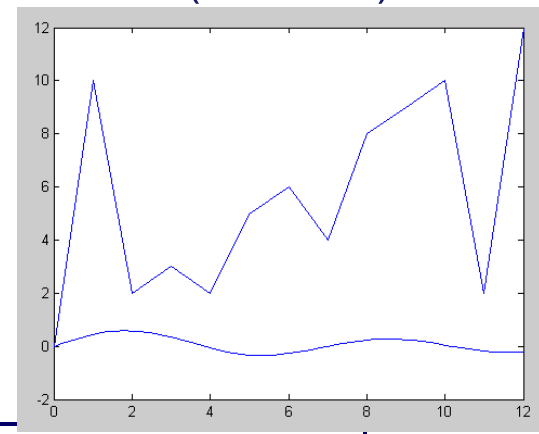
1ère fonction
(hold off)



2e fonction
(hold off)



2 fonctions
(hold on)



Fonctions utiles – **colormap()**

`colormap(type)`

- fonction qui permet de personnaliser la palette de couleurs disponible pour certains graphiques.

`type`: nom d'une collection de couleurs prédéfinies dans MATLAB

Terme	Description
autumn	Différents tons qui vont du rouge en passant par orange jusqu'au jaune.
bone	Tons de gris avec de plus grandes valeurs pour la composante bleue.
colorcube	Inclut le plus de couleurs opposées possibles.
cool	Varie doucement du cyan jusqu'au magenta.
copper	Varie doucement du noir jusqu'au cuivre brillant.
flag	Composée des couleurs rouge, blanc, bleu et noir.
gray	Tous les tons de gris possibles.
hot	Varie doucement du noir en passant par le rouge, l'orange, le jaune jusqu'au blanc.
hsv	Cette couleur commence avec rouge, passe par le jaune, le vert, le cyan, le bleue, le magenta et retourne au rouge.
jet	Varie entre le bleu et le rouge en passant par le cyan, le jaune et l'orange.
lines	Produit des couleurs spécifiées par l'ordre des couleurs des axes de MATLAB et certains tons de gris.
pink	Plusieurs tons pastels de rose.
prism	Répète six couleurs : rouge, orange, jaune, vert, bleue et violet.
spring	Varie du magenta au jaune.
summer	Varie du vert au jaune.
white	Uniquement composé de blanc.
winter	Varie du bleu au vert.

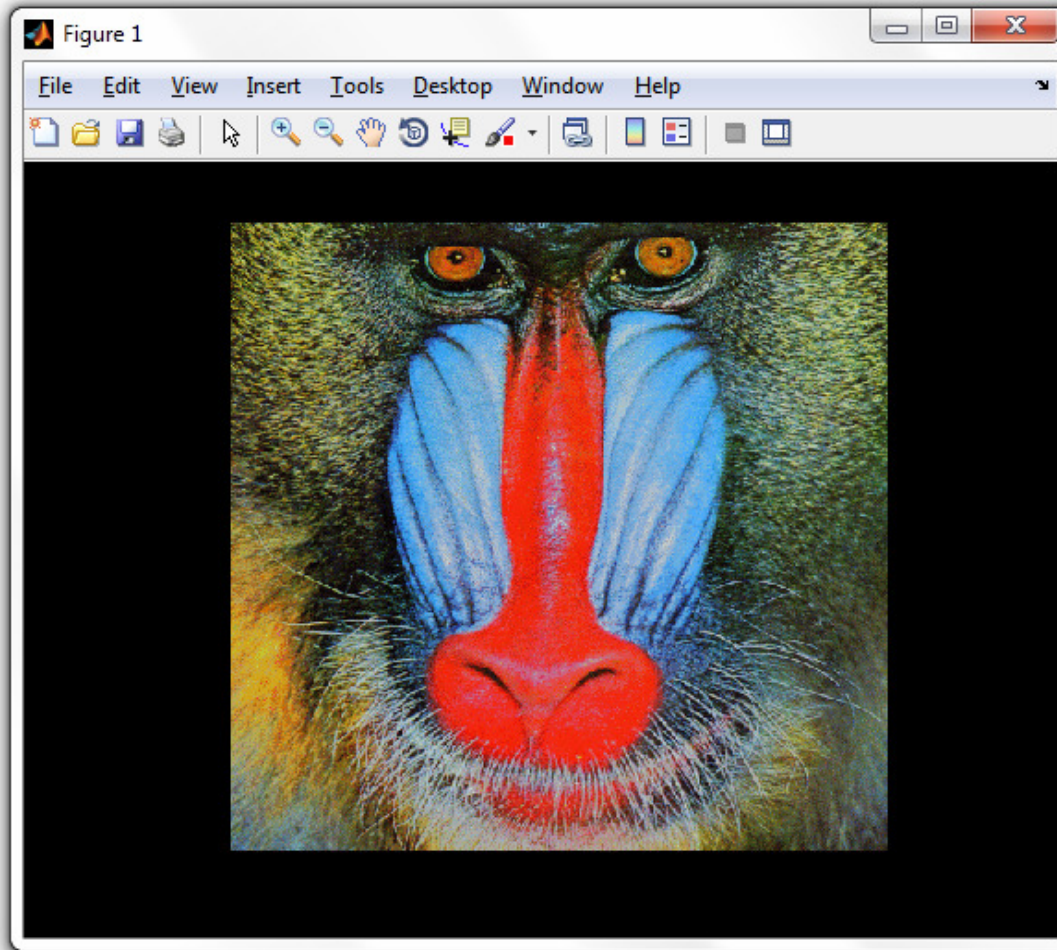


Fonctions utiles – **colormap()**

Command Window

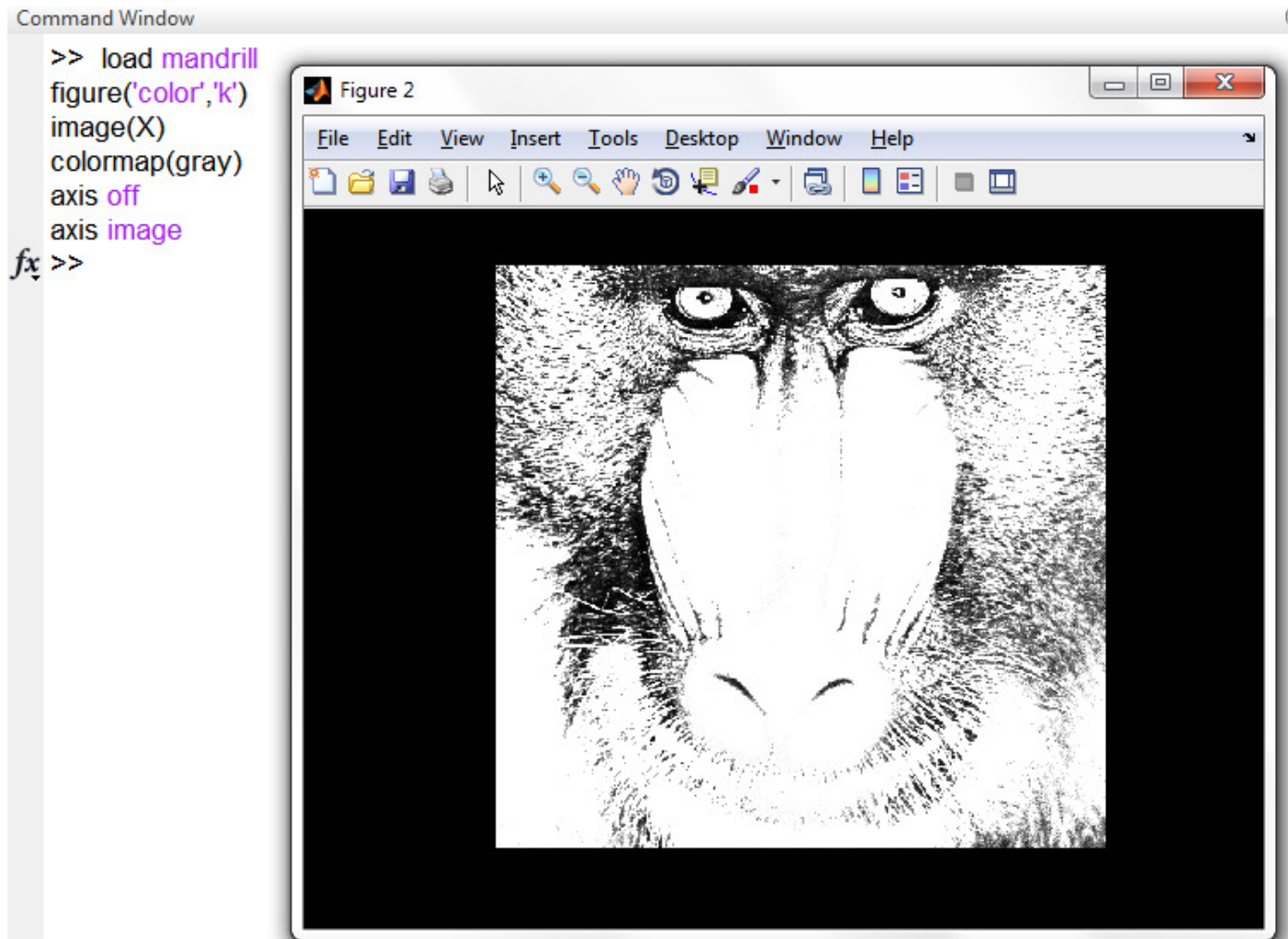
```
>> load mandrill  
figure('color','k')  
image(X)  
colormap(map)  
axis off  
axis image  
>>
```

Appel de
la fonction





Fonctions utiles – `colormap()`





Agenda



Traçage de graphiques



Graphiques – Ajouter des propriétés



Graphiques – fonctions utiles

- Graphiques – exporter un graphique
- Différents graphiques
- Quelques graphiques intéressants
- Rappel de fonctions

Exporter un graphique – `print()`

`print - format_fichier -options nom_fichier`

– la fonction permet d'exporter un graphique dans un fichier d'un format donné.

- deux formats très utilisés à connaître sont les formats EPS et JPEG

`print -depsc -tiff -r300 fic_graph1`

`depsc` : indique que le format voulu est EPS.

`tiff` : spécifie à MATLAB d'inclure l'information pour que le contenu du fichier puisse être visualisé. (fonction spécifique aux fichiers EPS)

`r300` : indique la résolution voulue pour le graphique (ici 300 dpi)

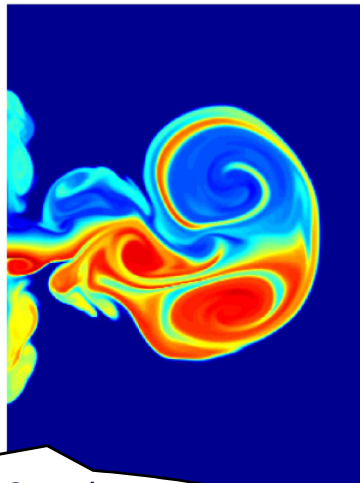
`print -djpeg -r300 fic_graph2`

`djpeg` : indique que le format voulu est JPEG

Exporter un graphique – `print()`

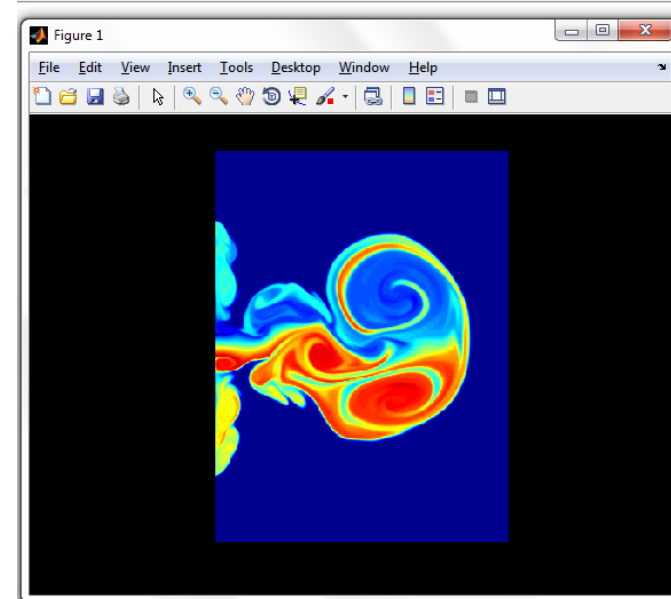
```
load flujet  
image(X)  
colormap(map)  
axis off  
axis image  
print -djpeg -r300 fic_graph2
```

graphique



fic_graph2.jpe

g





Agenda



Traçage de graphiques



Graphiques – Ajouter des propriétés



Graphiques – fonctions utiles



Graphiques – exporter un graphique

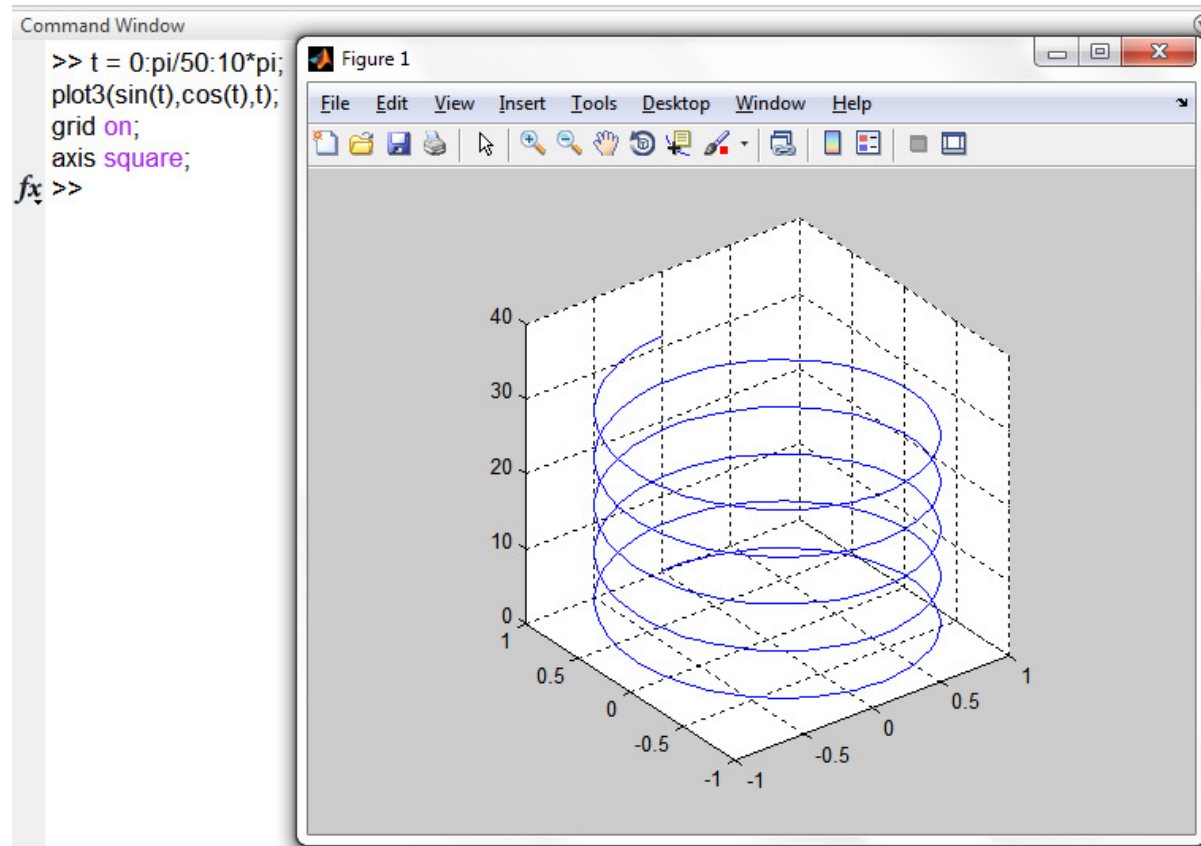
- Différents graphiques
- Quelques graphiques intéressants
- Rappel de fonctions

Différents graphiques – graphique en 3 dimensions

`plot3(X1,Y1,Z1,'format_ligne1',...)`

- la fonction est similaire à `plot()`; il faut ajouter une série de données correspondant aux coordonnées en Z de différents points.

i.e.

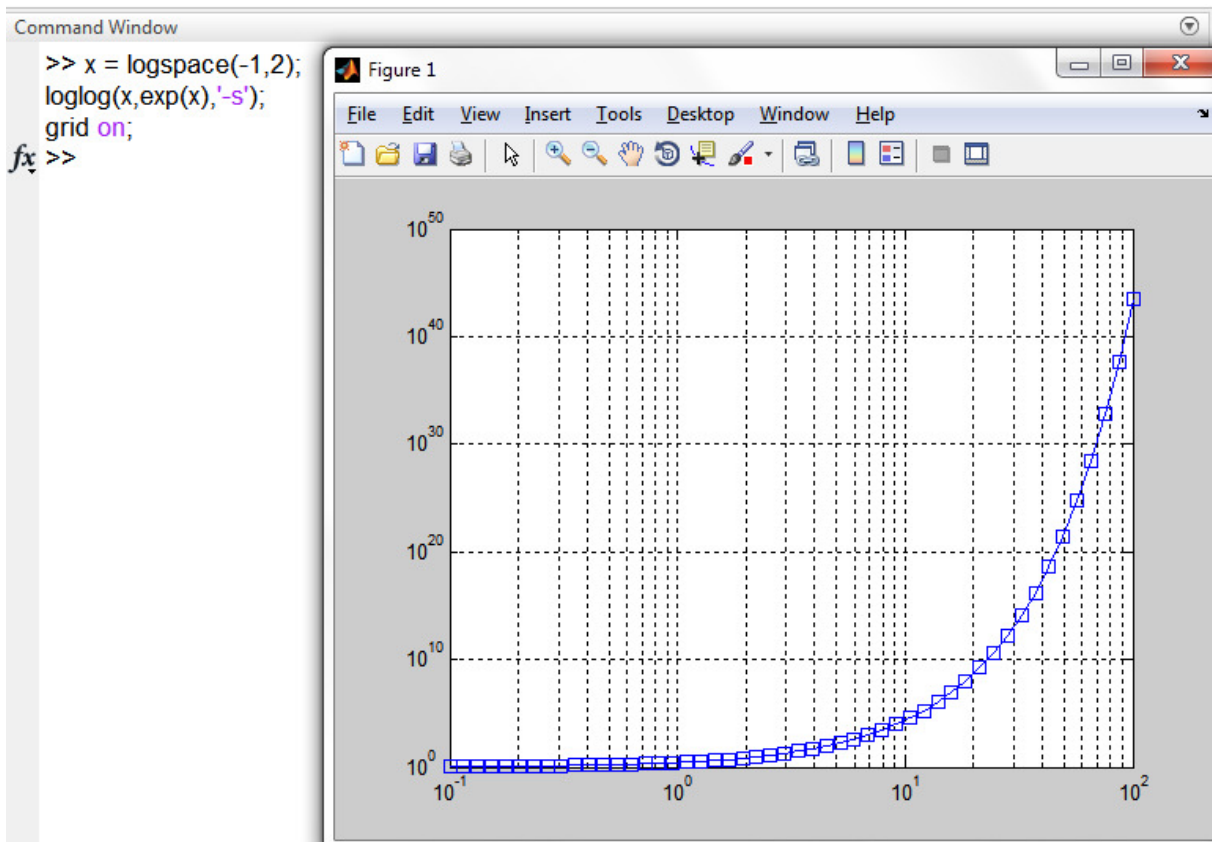


Différents graphiques – graphique logarithmique

`loglog(X1,Y1,'format_ligne1',...)`

– la fonction trace un graphique à échelle logarithmique; elle est similaire à `plot()`;

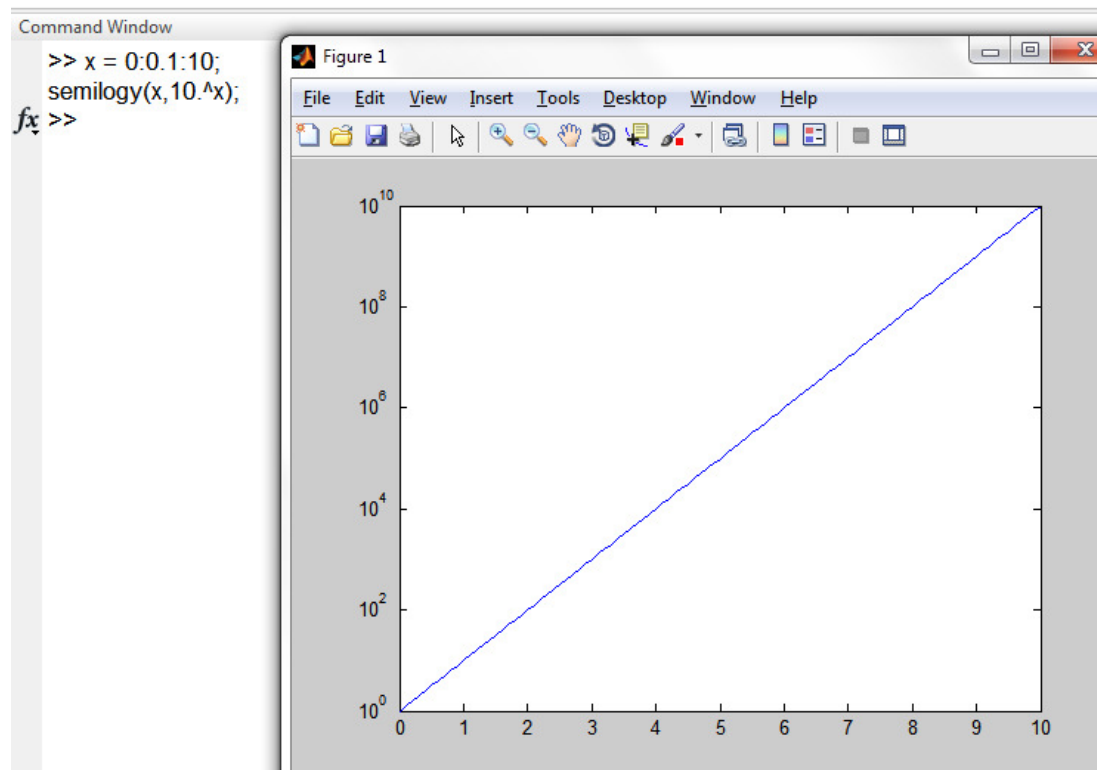
i.e.



Différents graphiques – graphique semi-logarithmique

```
semilogx(X1,Y1,'format_ligne1',...)  
semilogy(X1,Y1,'format_ligne1',...)
```

– les fonctions tracent de graphiques à échelle semi-logarithmique;
sont similaires à `plot()`;
i.e.



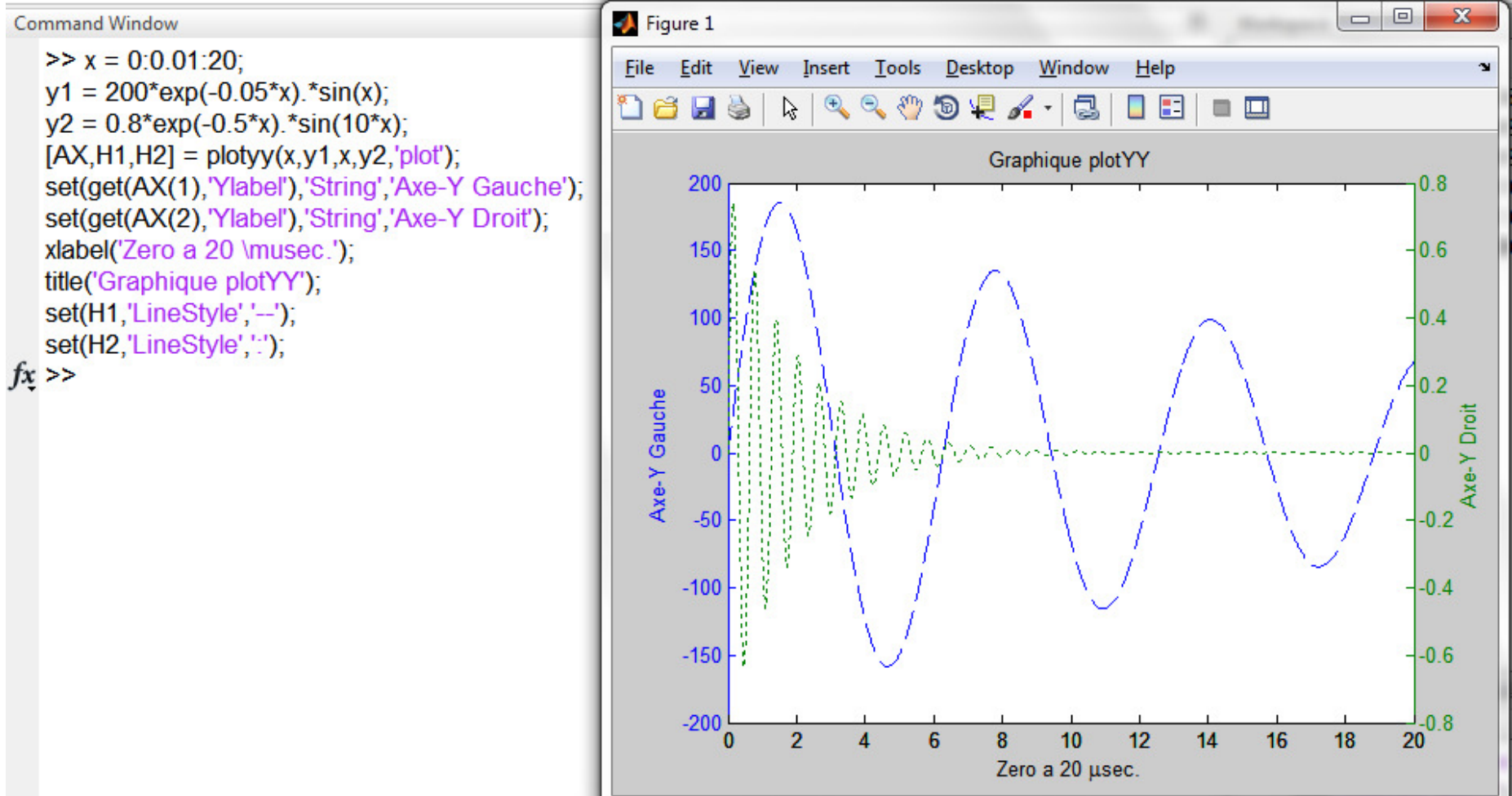
Différents graphiques – graphique avec 2 axes y

```
plotyy(X1,Y1,X2,Y2,'fonction1','fonction2')
```

- la fonction est trace un graphique à deux axes Y, une à droite et l'autre à gauche. Chaque axe a une échelle différente.

`fonctionn`: paramètres qui servent à définir la fonction utilisée pour tracer le graphique. Par exemple, il est possible de tracer des graphiques logarithmiques en donnant en paramètre `loglog()`. Si aucune fonction n'est spécifiée, la fonction `plot()` sera employée par défaut.

Différents graphiques – graphique avec 2 axes y



Différents graphiques – graphiques à barres

```
bar(x,y,width,'style','couleur')  
barh(x,y,width,'style','couleur')
```

- les fonctions tracent des graphiques à barres, pratiques pour illustrer, entre autre, des résultats financiers.

`bar()` dessine des barres verticales
`barh()` dessine des barres horizontales.

`x` : vecteur qui contient les données ou les noms des catégories qui apparaissent sur l'axe des X.(optionnel)

`y` : matrice qui contient les données qui seront représentées par des barres. Chaque donnée d'une même ligne est représentée par un groupe de barres. Il y a autant de groupes qu'il y a de lignes. Chaque groupe contient autant de barres que de colonnes.

Différents graphiques – graphiques à barres

```
bar(x,y,width,'style','couleur')  
barh(x,y,width,'style','couleur')
```

`width` : nombre - spécifie la largeur des barres. Par défaut définit à 0,8. Si définit à 1, les barres d'un même groupe se touchent.
(optionnel)

`style` : définit le style de configuration des groupes. Peut être définit à grouper (par défaut) (`grouped`) ou empiler (`stacked`).
(optionnel)

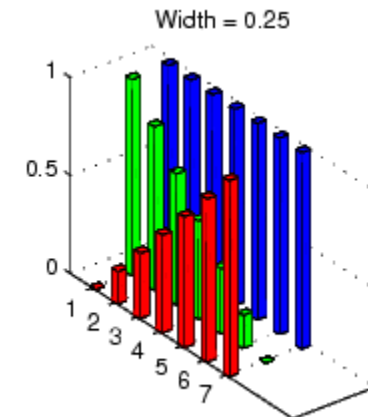
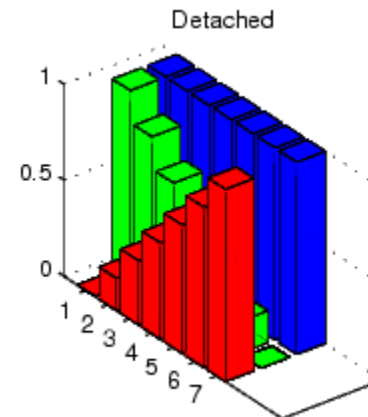
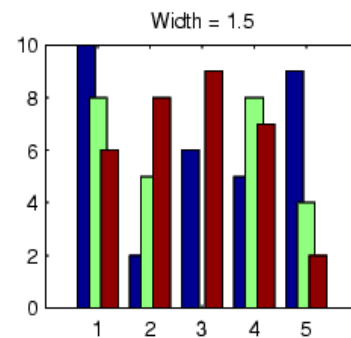
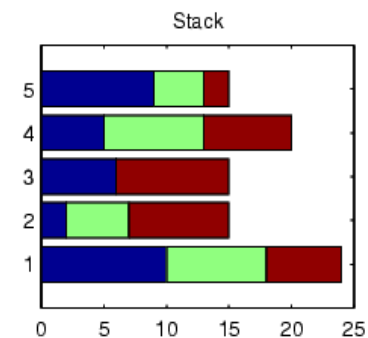
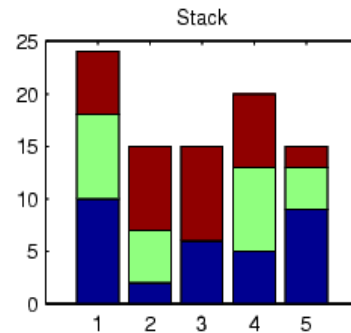
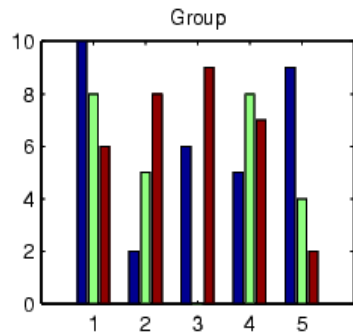
`couleur` : permet de définir la couleur que prendra chacun des barres du graphique. Utilise les mêmes abréviations de couleur que la fonction `plot()`. (optionnel)

Les versions trois dimensions de ces fonctions fonctionnent de la même façon que les versions deux dimensions (`bar3()` et `barh3()`).

Différents graphiques – graphiques à barres

fonctions bar et barh

fonctions bar3 et bar3h





Agenda



Traçage de graphiques



Graphiques – Ajouter des propriétés



Graphiques – fonctions utiles



Graphiques – exporter un graphique



Différents graphiques

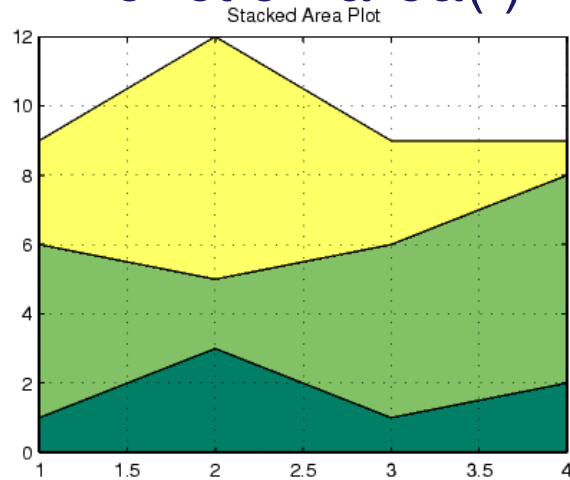
- Quelques graphiques intéressants

- Rappel de fonctions

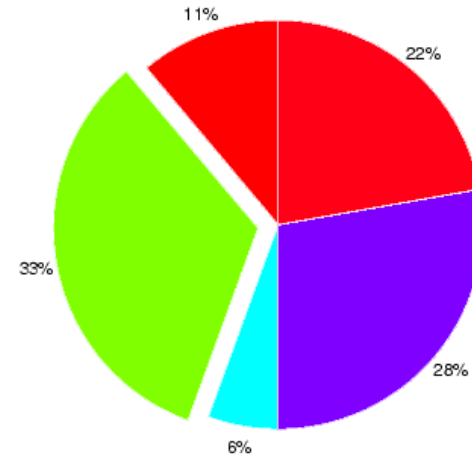


Quelques graphiques intéressants

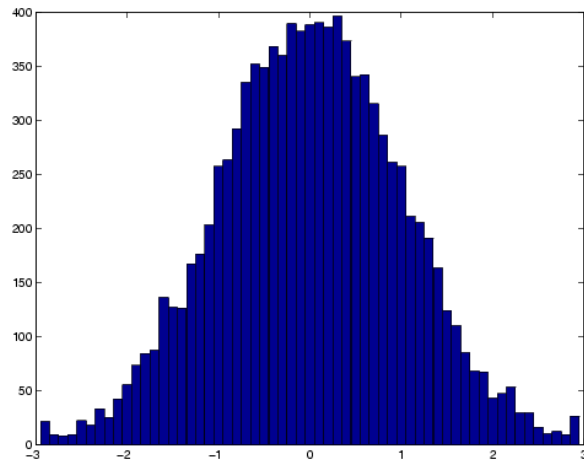
fonction area()



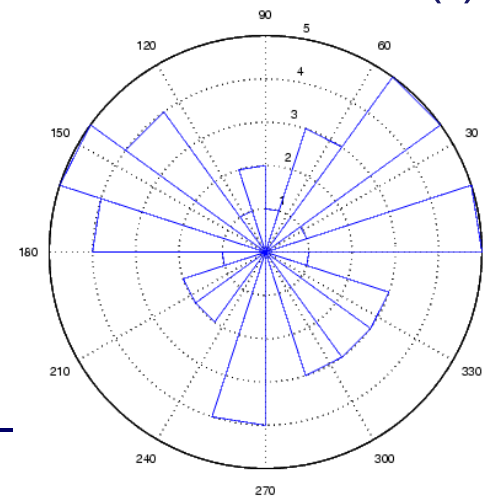
fonction pie()



fonction hist()



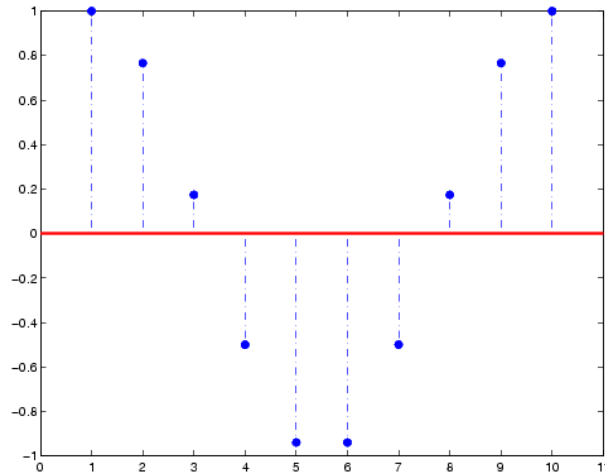
fonction rose()



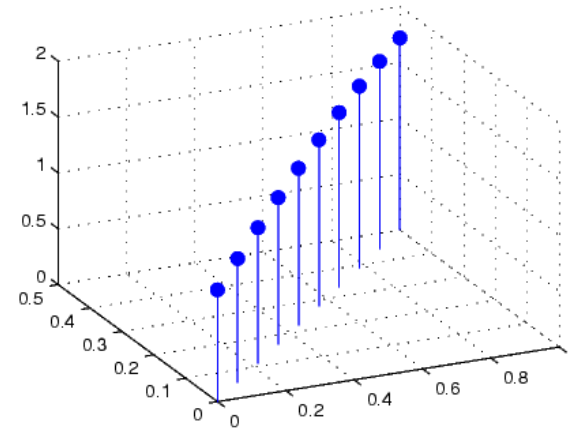


Quelques graphiques intéressants

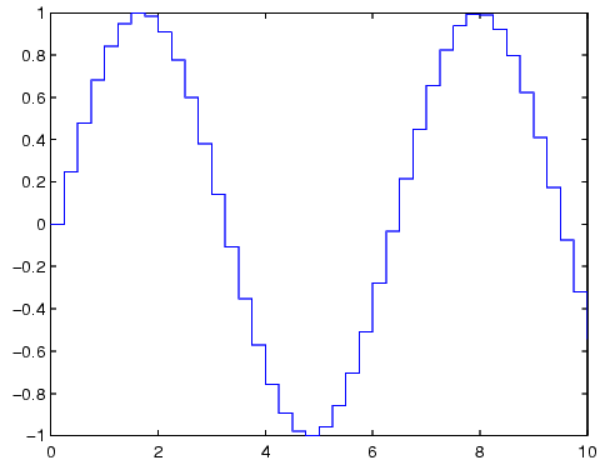
fonction stem()



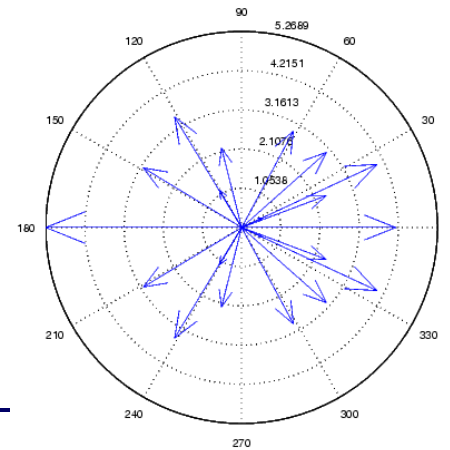
fonction stem3()



fonction stairs()



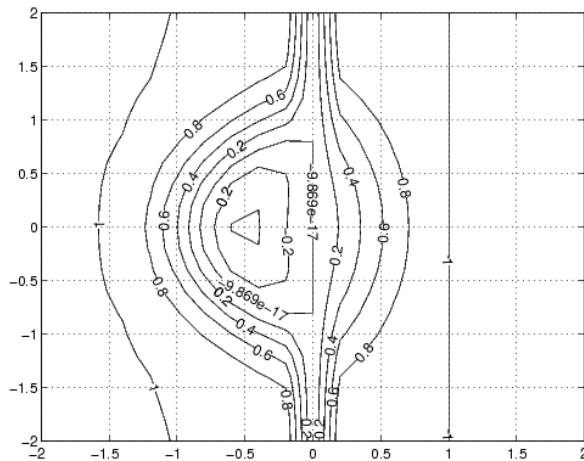
fonction compass()



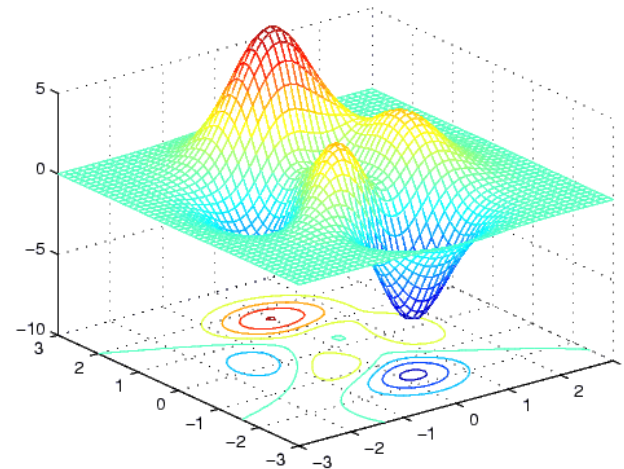


Quelques graphiques intéressants

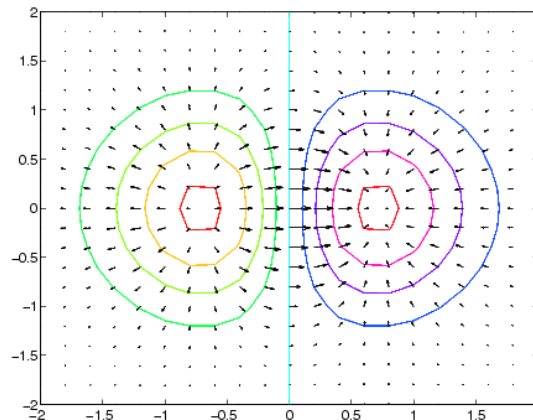
fonction clabel()



fonction mesh()



fonction quiver()





Agenda



Traçage de graphiques



Graphiques – Ajouter des propriétés



Graphiques – fonctions utiles



Graphiques – exporter un graphique



Différents graphiques



Quelques graphiques intéressants

- Rappel de fonctions

Graphique MATLAB – Rappel de fonctions

`figure (no_de_figure)`

`subplot (ligne, colonne, position)` – positionnement dans la fenêtre

`plot (X1, Y1, 'format_ligne1', X2, Y2, 'format_ligne2', ...)`
– fonction de traçage

`set (nograph, 'nom_propriété1', valeur_propriété1, ...)`

- définition des propriétés spécifiques pour le graphique.

`axis ([xmin xmax ymin ymax])` – format des axes

`xlabel('titre_axe_des_x')` et `ylabel('titre_axe_des_y')`

– annotation des axes

`title('titre_du_graphique_x')` – titre du graphique

`text(x, y, 'commentaire', 'format_commentaire')` – ajouter des commentaires

`legend('description1', 'description2', ..., position)`

- ajouter une légende

Graphiques MATLAB – Rappel de fonctions

`hold on / off` – ajouter les prochains graphiques créés au graphique déjà existant

`colormap('Type')` – personnaliser la palette des couleurs

`print -format_fichier -options nom_fichier` – exporter le graphique

`loglog(X1,Y1,'format_ligne1',X2,Y2,'format_ligne2',...)`
- échelle logarithmique

`semilogx(X1,Y1,'format_ligne1',X2,Y2,'format_ligne2',...)`
`semilogy(X1,Y1,'format_ligne1',X2,Y2,'format_ligne2',...)` – échelle semi-logarithmique

`plotyy(X1,Y1,X2,Y2,'fonction1','fonction2')` – graphique avec deux axes Y

`bar(x,y,width,'style','couleur'), barh(x,y,width,'style','couleur')` graphique à barres



Agenda



Traçage de graphiques



Graphiques – Ajouter des propriétés



Graphiques – fonctions utiles



Graphiques – exporter un graphique



Différents graphiques



Quelques graphiques intéressants



Rappel de fonctions



Sommaire

- 1 **Traçage de graphiques**
- 2 **Graphiques simples - ajouter des propriétés**
- 3 **Graphiques simples – fonctions utiles**
- 4 **Graphiques simples – exporter un graphique**
- 5 **Différents graphiques**
- 6 **Quelques graphiques intéressants**
- 7 **Rappel de fonctions**