Chapitre : Représentation graphique sous Matlab

MATLAB, outre de permettre de faire des calculs numériques de très haut niveau, il peut aussi produire des graphiques impressionnants, de type 2D ou 3D. Pour avoir un bref aperçu des possibilités graphiques de MATLAB, on a accès à des démos de graphiques.

>> demo matlab graphics

Dans ce chapitre, nous allons présenter les principes de base indispensables pour dessiner des courbes en MATLAB.

La commande plot

La commande **plot** permet de tracer un ensemble de points de coordonnées (x_i , y_i), $i=1, ..., N$.

La syntaxe est plot(x,y) où x est le vecteur contenant les valeurs x_i en abscisse et y est le vecteur contenant les valeurs y_i en ordonnée.

Bien entendu les vecteurs x et y doivent être de même dimension mais il peut s'agir de vecteurs lignes ou colonnes.

Par défaut, les points (x_i , y_i) sont reliés entre eux par des segments de droites.

```
>> x=[-2*pi:0.01:2*pi]; y = x.*sin(x);
>> plot(x,y)
```

La commande plot

Voici par exemple une autre façon de tracer le graphe de la fonction *h(x) = x sin(x)* entre -2π et 2π.

Essayez aussi :
*>> x=[-2*pi:1:2*pi]; y = x.*sin(x);*
>> plot(x,y)

Dans cet exemple on a défini un vecteur x de valeurs équi-réparties entre -2π et 2π (avec un pas de *0.01* dans le premier cas et de *1* dans le deuxième cas) et on a calculé l'image par la fonction *h* de ces valeurs (vecteur y). On affiche les points de coordonnées (x(i), y(i)).

La commande plot

On peut spécifier à MATLAB quelle doit être la couleur d'une courbe, quel doit être le style de trait et/ou quel doit être le symbole à chaque point (x_i , y_i). Pour cela on donne un troisième paramètre d'entrée à la commande plot qui est une chaîne de 3 caractères de la forme **'cst'** avec **c** désignant la couleur du trait, **s** le symbole du point et **t** le type de trait. Les possibilités sont les suivantes:

Couleur de la courbe		**Représentation des points**		**Style de la courbe**	
le caractère	**son effet**	**le caractère**	**son effet**	**le caractère**	**son effet**
b ou **blue**	courbe en bleu	**.**	un point	-	en ligne plein
g ou **green**	courbe en vert	**o**	un cercle	**:**	en pointillé
r ou **red**	courbe en rouge	**x**	le symbole **x**	**-.**	en point tiret
c ou **cyan**	entre le vert et le bleu	**+**	le symbole **+**	**--**	en tiret
m ou **magenta**	en rouge violacé vif	*****	une étoile *****		
y ou **yellow**	courbe en jaune	**s**	un carré		
k ou **black**	courbe en noir	**d**	un losange		
		v	triangle inferieur		
		^	triangle supérieur		
		<	triangle gauche		
		>	triangle droit		
		p	pentagramme		
		h	hexagramme		

La commande plot

Les valeurs par défaut sont c = b, s =. et t = - ce qui correspond à un trait plein bleu reliant les points entre eux. Il n'est pas obligatoire de spécifier chacun des trois caractères. On peut se contenter d'en spécifier un ou deux. Les autres seront les valeurs par défaut. La commande **grid** permet d'obtenir un quadrillage de la figure.

La commande plot

Il est possible de tracer plusieurs courbes sur la même figure en spécifiant plusieurs tableaux x1, y1, x2, y2, ..., comme paramètres de la commande plot. Si l'on souhaite que les courbes aient une apparence différente, on utilisera des options de couleurs et/ou de styles de traits distincts après chaque couple de vecteurs x, y.

Exemple :

```
>> x = [-5:0.01:5];
>> y = x.^2.*cos(x); z = x.*cos(x);
>> plot(x,y,'b-',x,z,'r:');
```

On trace sur l'intervalle [-5, 5] la fonction $x^2 \cos(x)$ en trait plein bleu et la fonction $x \cos(x)$ en trait pointillé rouge.

Améliorer la lisibilité d'une figure

Légender une figure

Dans une figure, il est préférable de mettre une description textuelle aidant l'utilisateur à comprendre la signification des axes et de connaitre le but ou l'intérêt de la visualisation concernée.

Il est très intéressant également de pouvoir signaler des emplacements ou des points significatifs dans une figure par un commentaire signalant leurs importances.

- Pour donner un titre pour l'axe horizontal des abscisses x, nous utilisons la fonction **xlabel** comme ceci :

>> xlabel('Ceci est l''axe des abscisses X')

- Pour donner un titre pour l'axe vertical des ordonnées y, nous utilisons la fonction **ylabel** comme ceci :

>> ylabel('Ceci est l''axe des ordonnées Y')

Améliorer la lisibilité d'une figure

Légender une figure

- Pour donner un titre à une figure contenant une courbe nous utilisons la fonction **title** comme ceci :

>> title('titre de la figure')

- Pour écrire un texte (un message) sur la fenêtre graphique à une position indiquée par les coordonnées **x** et **y**, nous utilisons la fonction **text** comme ceci :

>> text(x, y, 'Ce point est important')

- Pour mettre un texte sur une position choisie manuellement par la souris, nous utilisons la fonction **gtext**, qui a la syntaxe suivante :

>> gtext('Ce point est choisi manuellement')

Légender une figure

- Il est possible avec ces commandes d'afficher une valeur contenue dans une variable au milieu de texte. Pour cela on construit un tableau de type chaîne de caractères en convertissant la valeur contenue dans la variable en une chaîne de caractères grâce à la commande **num2str**.

>>title(['Exemple numero ', num2str(numex)]).

Exemple :

L'exemple suivant illustre l'utilisation des différentes commandes permettant de légender une figure.

>> P = 5;>> t = [0:.01:2];
*>> c = 12*exp(-2*t) - 8*exp(-6*t);*
>> plot(t,c); grid
>> xlabel('temps en minutes')
>> ylabel('concentation en gramme par litre')
>> title(['evolution de la concentration du produit ', num2str(P), ... ' au cours du temps '])
>> gtext('concentration maximale')

Afficher plusieurs courbes dans une même fenêtre
Il est possible d'afficher plusieurs courbes dans une même fenêtre graphique grâce à la commande **hold on**.

Les résultats de toutes les instructions graphiques exécutées après appel à la commande **hold on** seront superposés sur la fenêtre graphique active.

Pour rétablir la situation antérieure (le résultat d'une nouvelle instruction graphique remplace dans la fenêtre graphique le dessin précédent) on tapera **hold off**.

Par exemple pour dessiner la courbe des deux fonctions cos(x) et sin(x) dans la même figure, on peut écrire :

```
>> x=0:pi/12:2*pi;
>> y1=cos(x);
>> y2=sin(x);
>> plot(x,y1,'b-o')
>> hold on
>> plot(x,y2,'r-s')
```

Sauvegarder une figure

La commande print permet de sauvegarder la figure d'une fenêtre graphique dans un fichier sous divers formats d'images. La syntaxe de la commande print est:

print -f<num> -d<format> <nomfic>
où

<num> désigne le numéro de la fenêtre graphique. Si ce paramètre n'est pas spécifié, c'est la fenêtre active qui est prise en compte.

<nomfic> est le nom du fichier dans lequel est sauvegardée la figure. Si aucune extension de nom n'est donnée, une extension par défaut est ajoutée au nom du fichier en fonction du format choisi (.ps pour du PostScript, .jpg pour du jpeg, par exemple).

<format> est le format de sauvegarde de la figure. Ces formats sont nombreux. On pourra obtenir la liste complète en tapant help plot. Les principaux sont:

Sauvegarder une figure

Format	Signification
ps	PostScript noir et blanc
psc	PostScript couleur
eps	PostScript Encapsulé noir et blanc
epsc	PostScript Encapsulé couleur
jpeg	Format d'image JPEG
tiff	Format d'image TIFF

La commande fplot

La commande **fplot** permet de tracer le graphe d'une fonction sur un intervalle donné. La syntaxe est: fplot('nomf', [x_{min} , x_{max}])

où :
nomf est soit le nom d'une fonction MATLAB incorporée, soit une expression définissant une fonction de la variable x, soit le nom d'une fonction utilisateur.

[x_{min}, x_{max}] est l'intervalle pour lequel est tracé le graphe de la fonction.

La commande subplot

Il est possible de décomposer une fenêtre en sous-fenêtres et d'afficher une figure différente sur chacune de ces sous-fenêtres grâce à la commande subplot.

La syntaxe est subplot(m,n,i)

où
m est le nombre de sous-fenêtres verticalement;
n est le nombre de sous-fenêtres horizontalement;
i sert à spécifier dans quelle sous-fenêtre doit s'effectuer l'affichage. Les fenêtres sont numérotées de gauche à droite et de haut en bas.

L'exemple suivant illustre l'utilisation de la commande subplot.

```
>> figure
>> subplot(2,3,1), fplot('cos',[0 4*pi]), title('cosinus'), grid
>> subplot(2,3,2), fplot('sin',[0 4*pi]), title('sinus'), grid
>> subplot(2,3,3), fplot('tan',[-pi/3 pi/3]), title('tangente'), grid
>> subplot(2,3,4), fplot('acos',[-1 1]), title('arc-cosinus'), grid
>> subplot(2,3,5), fplot('asin',[-1 1]), title('arc-sinus'), grid
>> subplot(2,3,6), fplot('atan',[-sqrt(3) sqrt(3)]), title('arc-tangente'), grid
```

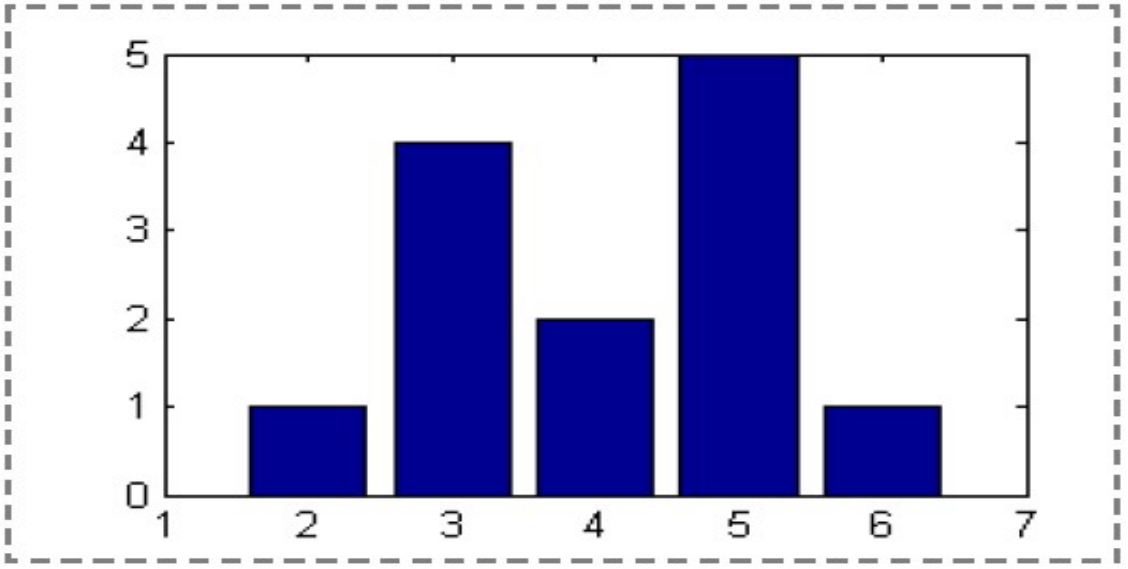
La commande bar

Le langage MATLAB ne permet pas uniquement l'affichage des points pour tracer des courbes, mais il offre aussi la possibilité de tracer des graphes à bâtons et des histogrammes.

Pour tracer un graphe à bâtons nous utilisons la fonction **bar** qui a le même principe de fonctionnement que la fonction **plot**.

Exemple :

```
>> X=[2,3,5,4,6];
>> Y=[1,4,5,2,1];
>> bar(X, Y)
```



Il est possible de modifier l'apparence des bâtons, et il existe la fonction **barh** qui dessine les bâtons horizontalement, et la fonction **bar3** qui ajoute un effet 3D.

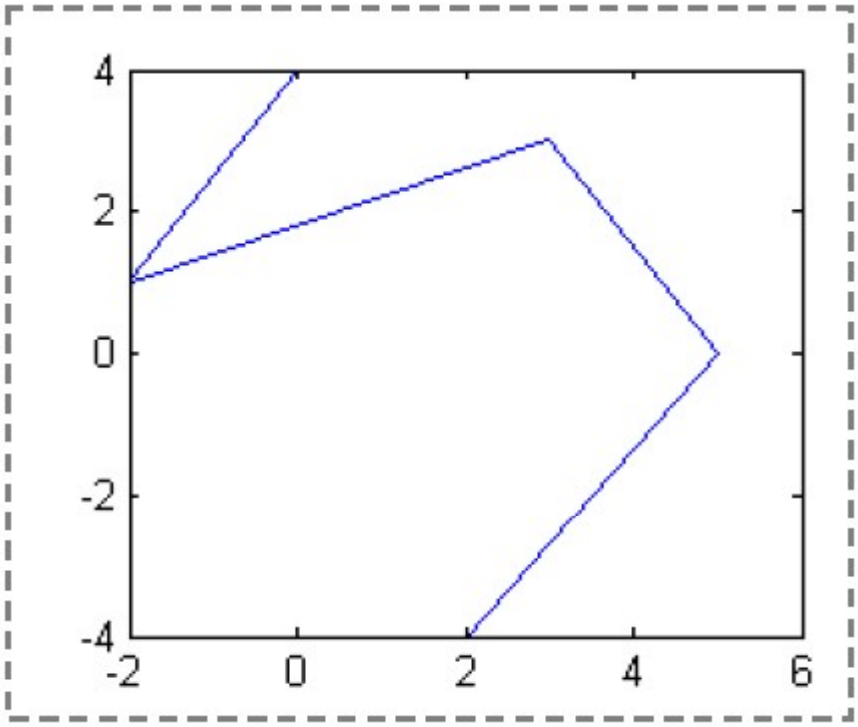
Exemples

Exemple :

```
>> A = [2, 5, 3,-2,0]
A =
       2      5      3     -2      0

>> B = [-4, 0, 3, 1,4]
B =
      -4      0      3      1      4

>> plot(A,B)
```

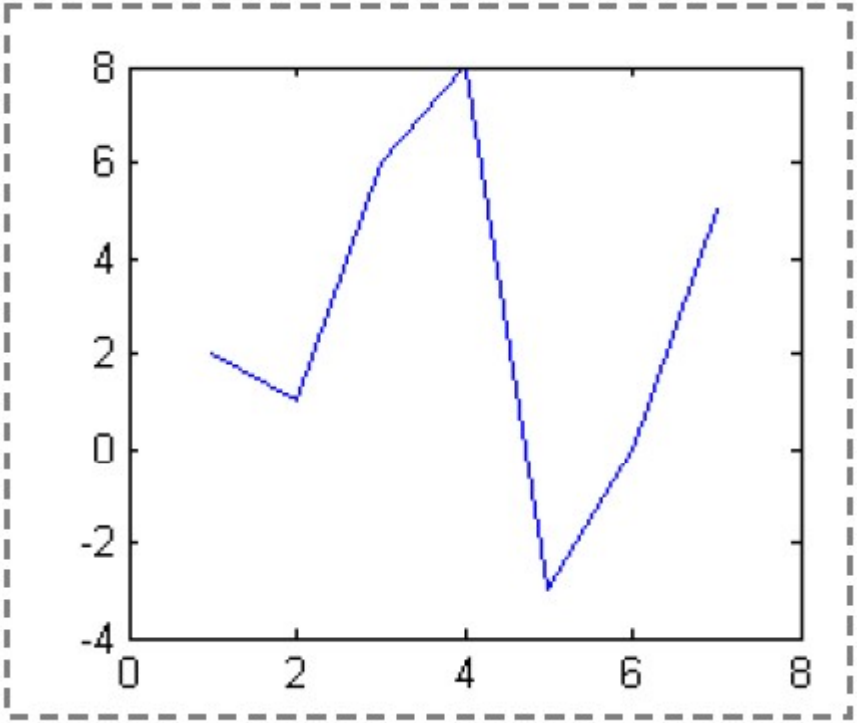


Exemples

Exemple :

```
>> V = [2, 1, 6, 8,-3, 0,5]
V =
       2    1    6    8   -3    0    5

>> plot(V)
```

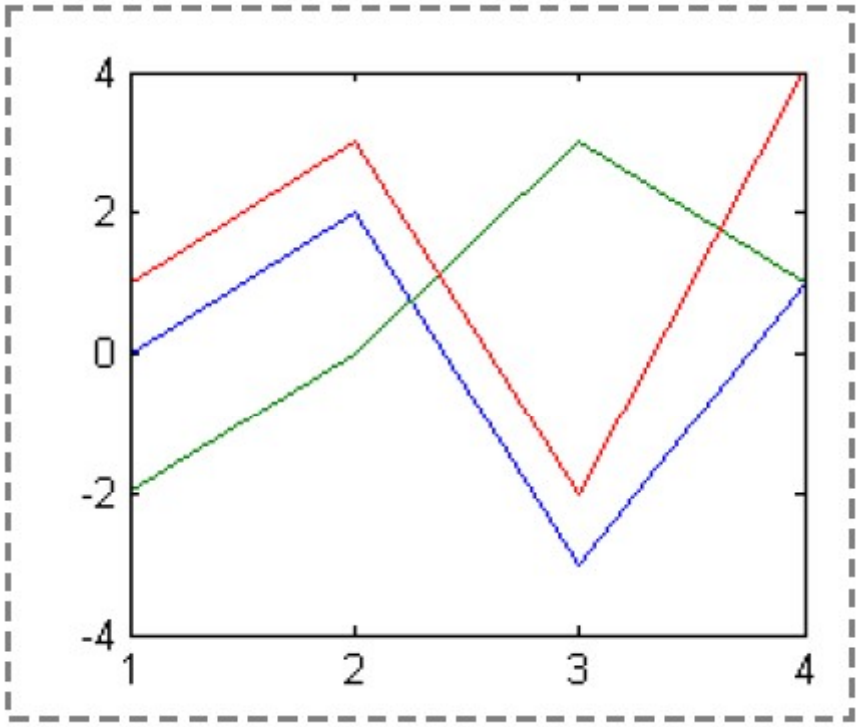


Exemples

Exemple :

```
>> M = [0 -2 1;2 0 3;-3 3 -2;1 1 4]
M =
     0    -2     1
     2     0     3
    -3     3    -2
     1     1     4

>> plot(M)
```



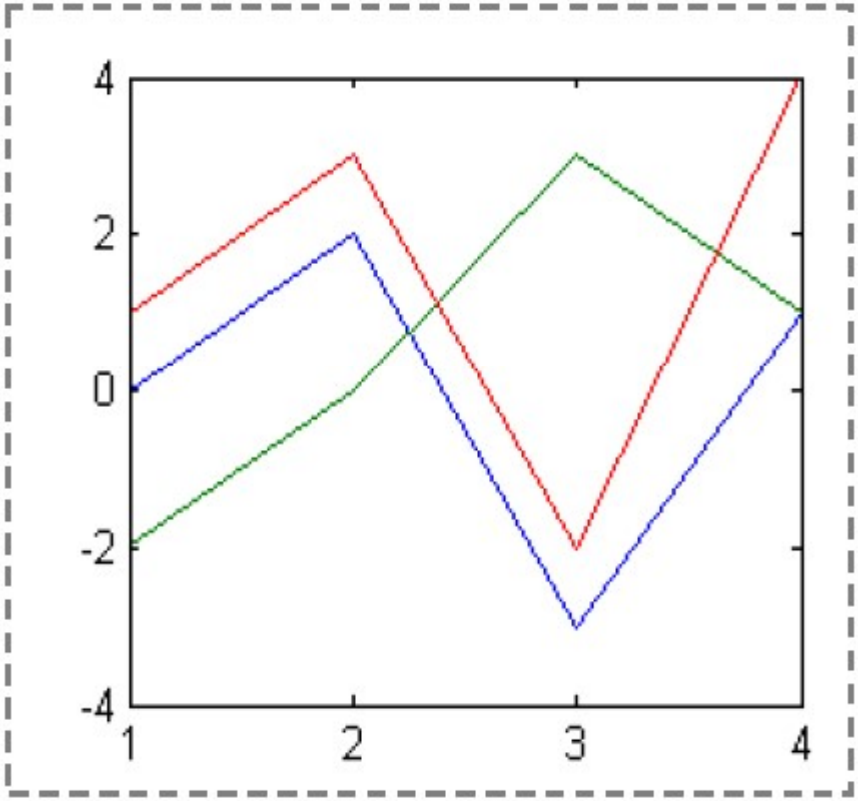
Exemples

Exemple :

```
>> K = [1 1 1;2 2 2; 3 3 3;4 4 4]
K =
     1     1     1
     2     2     2
     3     3     3
     4     4     4

>> M = [0 -2 1;2 0 3;-3 3 -2;1 1 4]
M =
     0    -2     1
     2     0     3
    -3     3    -2
     1     1     4

>> plot(K,M)
```

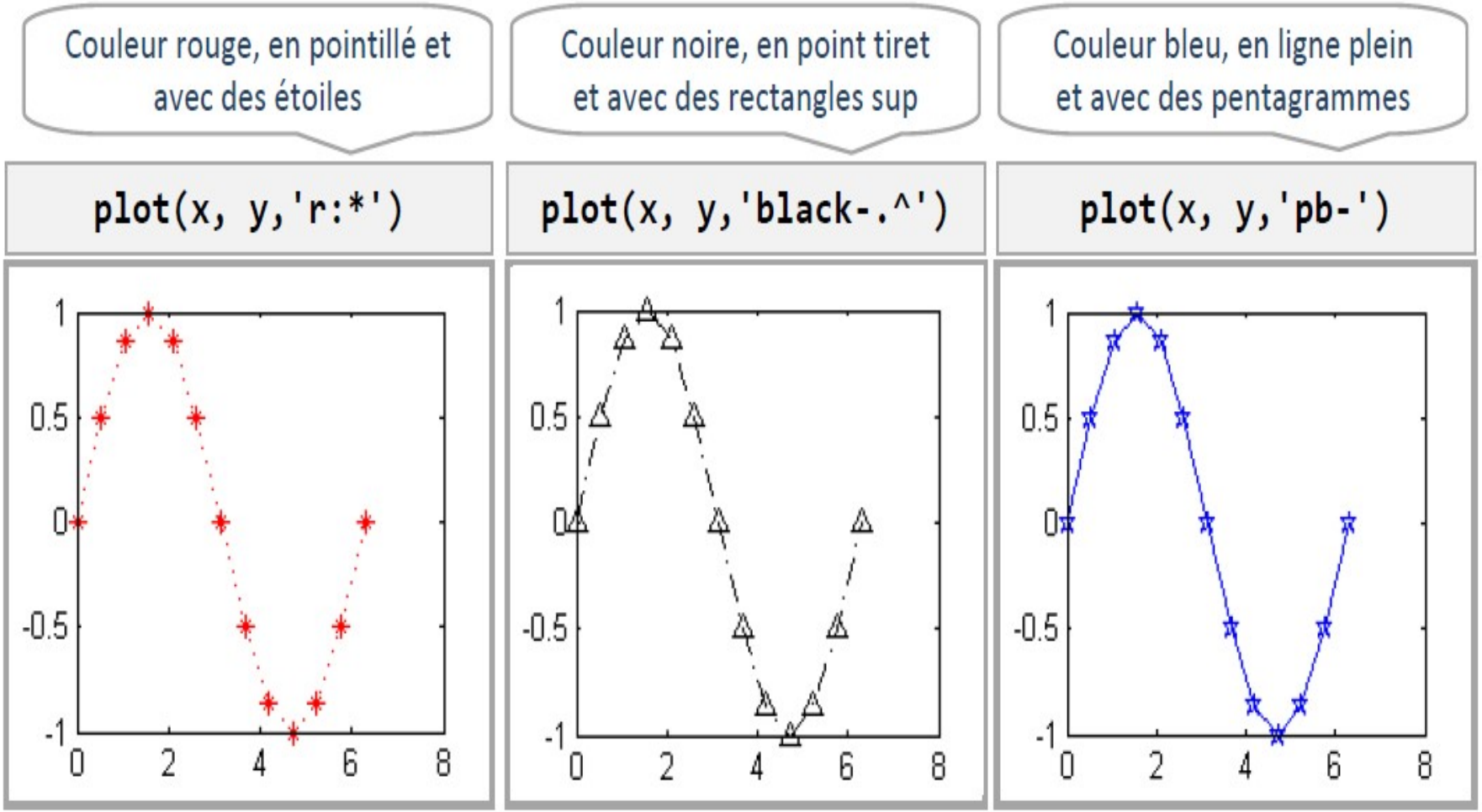


Exemples

Dessiner la fonction y = sin(x) pour x = [0 ... 2π] avec un pas = π/6.

>> x = 0:pi/6:2*pi;

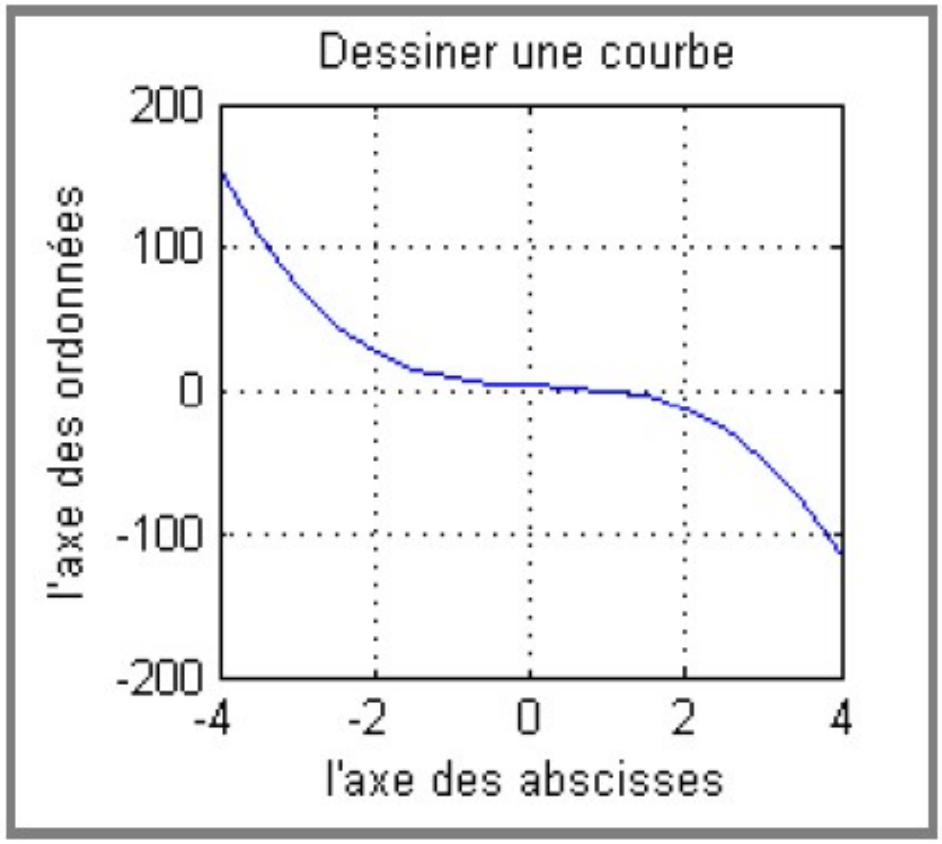
>> y = sin(x);



Exemples

Dessinons la fonction : y = -2x3+x2-2x+4 pour x varie de -4 jusqu'à 4, avec un pas = 0.5.

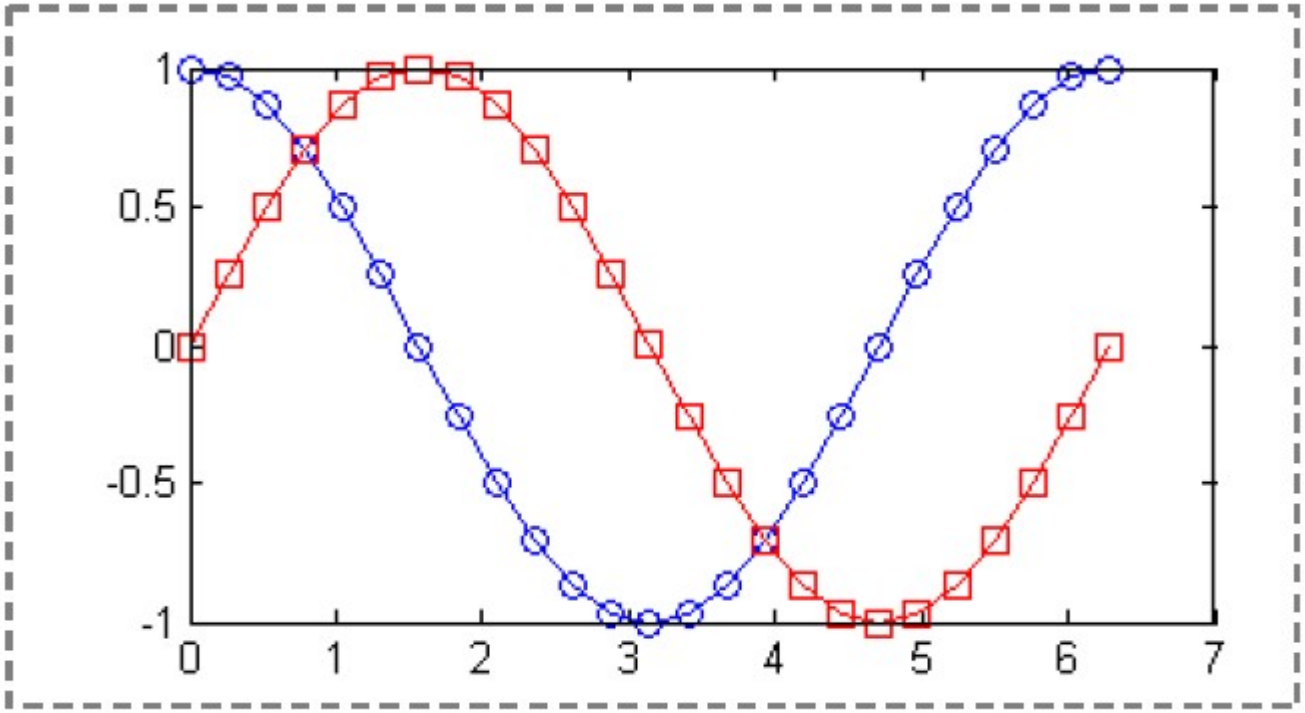
```
>> x = -4:0.5:4;
>> y = -2*x.^3+x.^2-2*x+4;
>> plot(x,y)
>> grid
>> title('Dessiner une courbe')
>> xlabel('l''axe des abscisses')
>> ylabel('l''axe des ordonnées')
```



Exemples

Dessiner la courbe des deux fonctions cos(x) et sin(x) dans la même figure.

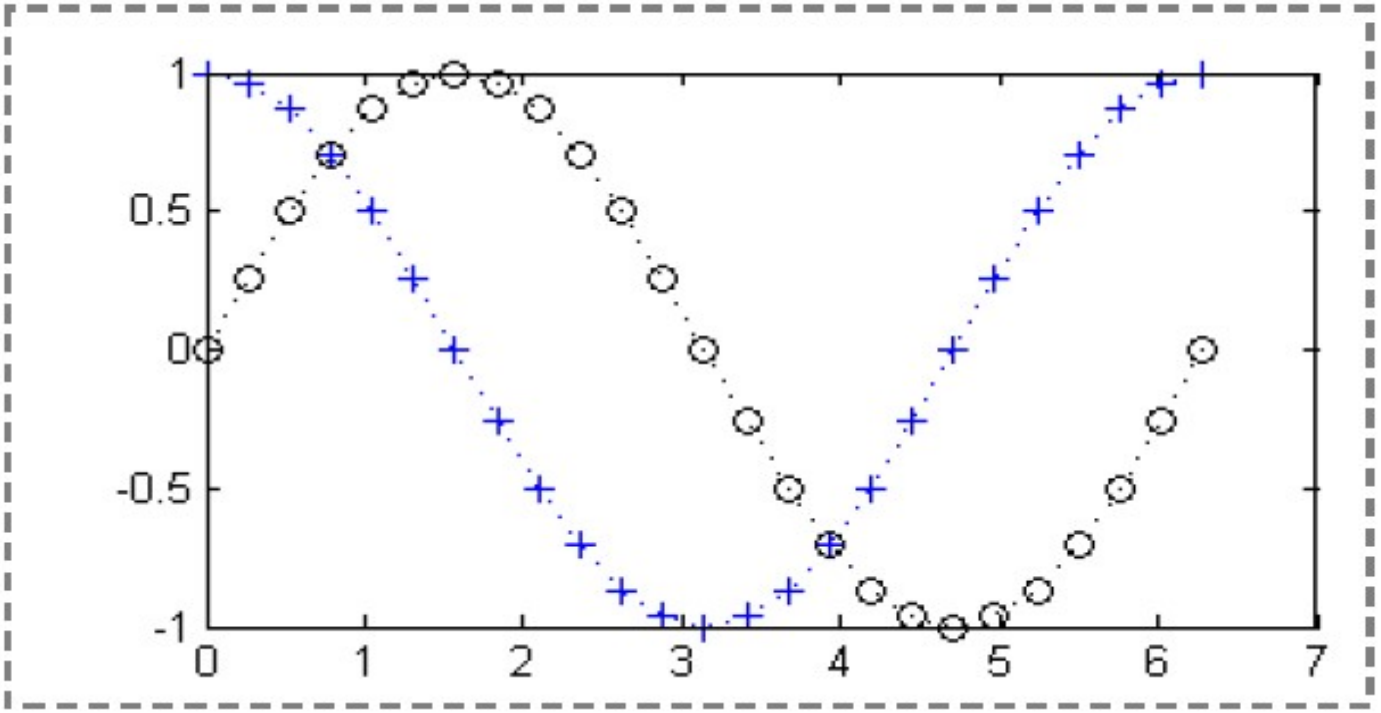
```
>> x=0:pi/12:2*pi
>> y1=cos(x);
>> y2=sin(x);
>> plot(x,y1,'b-o')
>> hold on
>> plot(x,y2,'r-s')
```



Exemples

Dessiner la courbe des deux fonctions cos(x) et sin(x) dans la même figure.

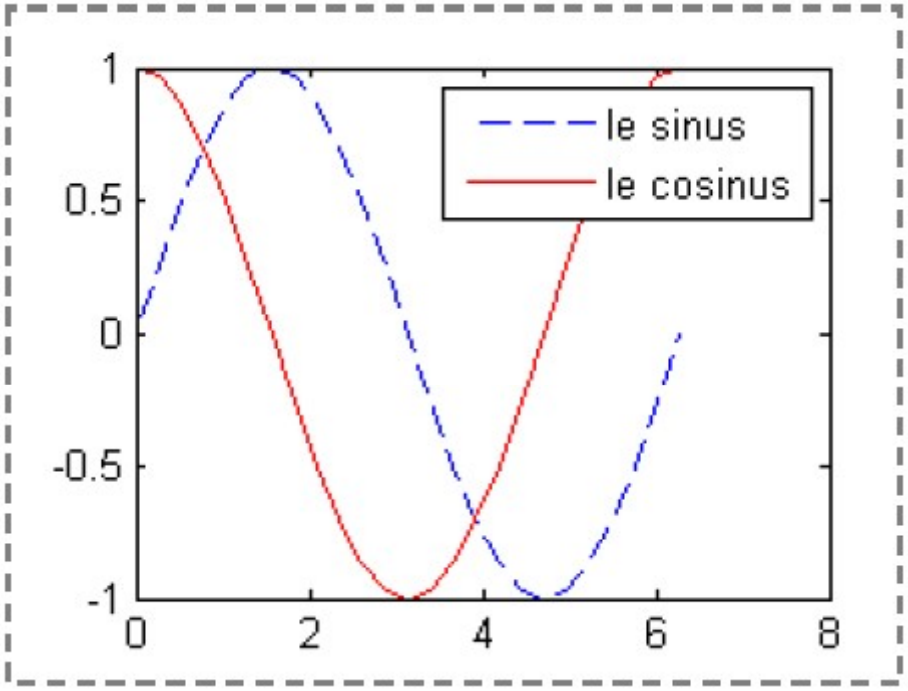
```
>> x=0:pi/12:2*pi;
>> y1=cos(x);
>> y2=sin(x);
>> plot(x,y1,'b:+',x,y2,'k:o')
```



Exemples

Dessiner la courbe des deux fonctions cos(x) et sin(x) dans la même figure.

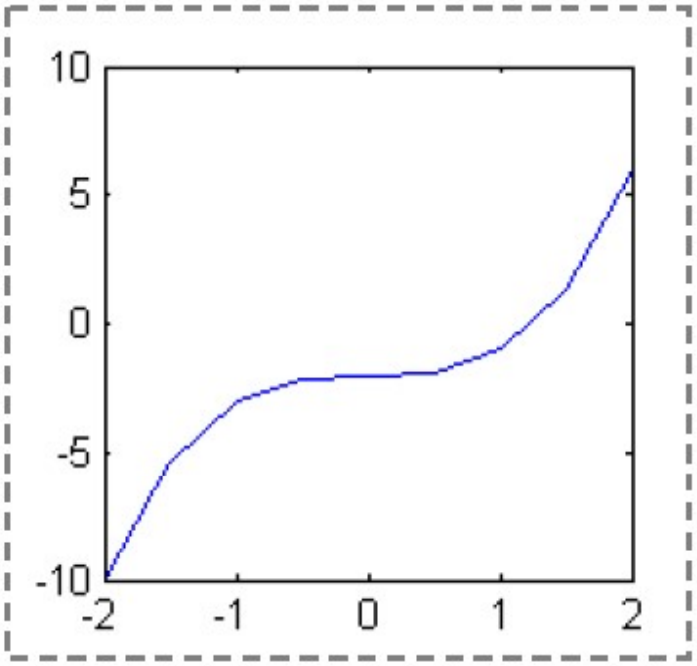
```
>> x=0:pi/12:2*pi;
>> y1=sin(x);
>> y2=cos(x);
>> plot(x,y1,'b--',x,y2,'-r')
>> legend('le sinus','le cosinus')
```



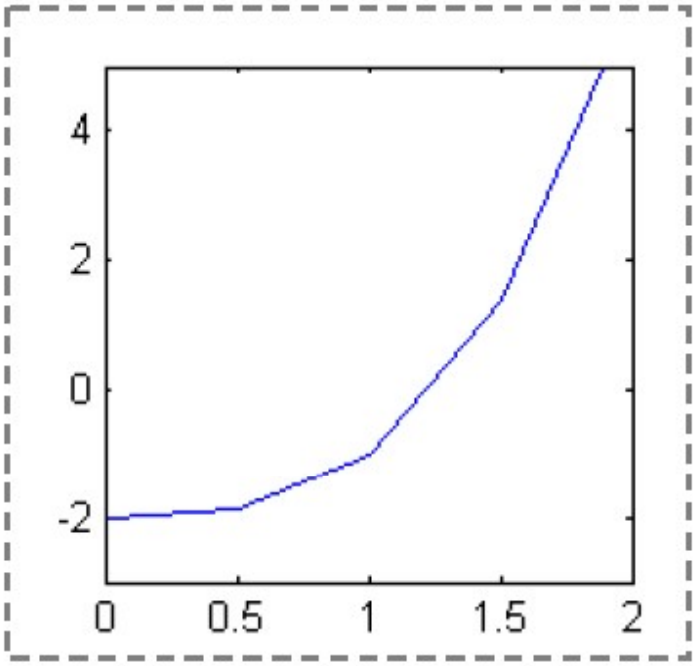
Exemples

Exemple :
F(x)= x^3-2

>> x = -2:0.5:2;
>> y = x.^3-2;
>> plot(x,y)



>> axis auto



>> axis([0,2,-3,5])