Chapitre 3 : Graphique

Une courbe 2D est pour tout logiciel de tracé de courbes représentée par une série d'abscisses et une série d'ordonnées. Ensuite, le logiciel trace généralement des droites entre ces points. MATLAB n'échappe pas à la règle. La fonction s'appelle **plot**.

1. L'Instruction Plot

a. Tracer une courbe simple

Syntaxe :

L'utilisation la plus simple de l'instruction plot est la suivante :

`plot` **(** ***vecteur d'abscisses, vecteur d'ordonnées*** **)**

$$[x_1 x_2 \cdots x_n][y_1 y_2 \cdots y_n]$$

Les vecteurs peuvent être indifféremment ligne ou colonne, pourvu qu'ils soient tous deux de même type. En général, ils sont lignes car la génération de listes de valeurs vue au chapitre précédent fournit par défaut des vecteurs ligne.

Exemple :

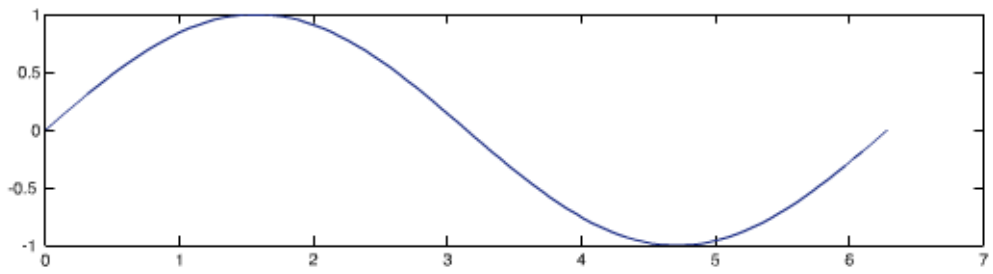
Si on veut tracer $sin(x)$ sur l'intervalle $[0,2\pi]$, on commence par définir une série (raisonnable, disons 100) de valeurs équidistantes sur cet intervalle :

```
>> x = 0: 2*pi/100 : 2*pi;
```

puis, comme la fonction **sin** peut s'appliquer terme à terme à un tableau :

```
>> plot(x, sin(x))
```

qui fournit le graphe suivant dans la fenêtre graphique :

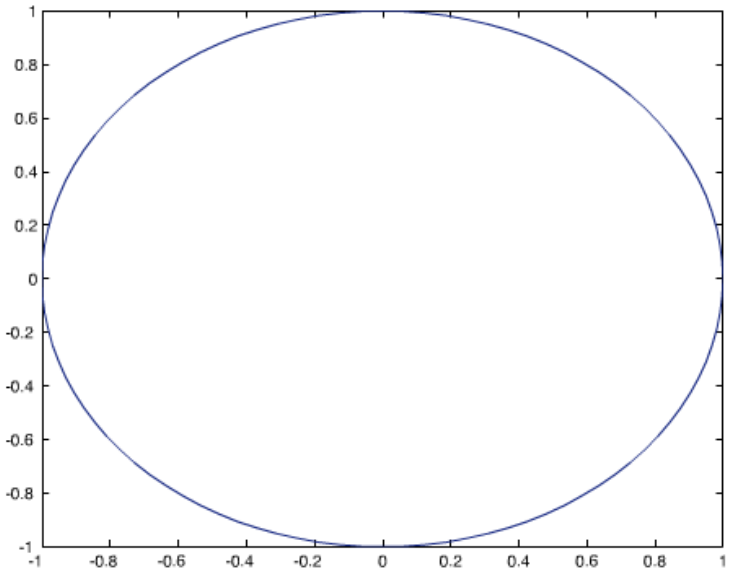


Courbe simple avec la commande plot

On voit que les axes s'adaptent automatiquement aux valeurs extrêmes des abscisses et ordonnées.

On remarquera que tout ce que demande `plot`, c'est un vecteur d'abscisses et un vecteur d'ordonnées. Les abscisses peuvent donc être une fonction de x plutôt que x lui-même. En d'autres termes, il est donc possible de tracer des courbes paramétrées :

```
>> x = 0: 2*pi/100 : 2*pi;
>> plot(cos(x), sin(x))
```

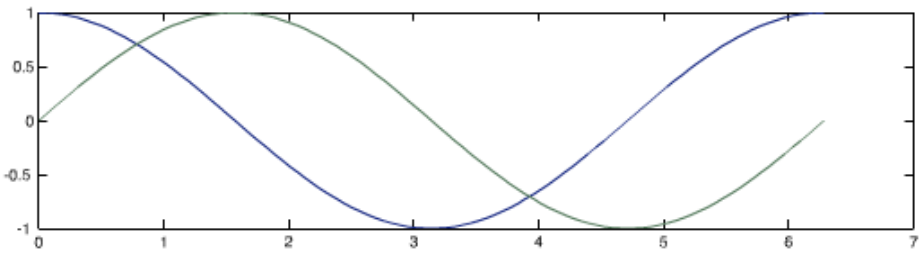


Courbe paramétrée avec la commande plot

b. Superposer plusieurs courbes

Il suffit de spécifier autant de couples (abscisses, ordonnées) qu'il y a de courbes à tracer. Par exemple pour superposer sin et cos :

```
>> x = 0: 2*pi/100 : 2*pi;
>> plot(x,cos(x),x, sin(x))
```



Superposition de courbes avec la commande plot

même si les abscisses des deux courbes ne sont pas les mêmes.

Dans le cas plus fréquent où les abscisses sont les mêmes, il existe un autre moyen de superposer les courbes. On fournit toujours le vecteur des abscisses, commun aux deux courbes, et on fournit autant de vecteurs d'ordonnées qu'il y a de courbes. Tous ces vecteurs d'ordonnées sont regroupés dans un même tableau, chaque ligne du tableau représentant un vecteur d'ordonnées :

`plot ( vecteur d'abscisses, tableau d'ordonnées )`

Exemple :

Par exemple, pour superposer sin et cos, on devra fournir à **`plot`** les arguments suivants :

```
>> plot(x, [cos(x);sin(x)])
```

c. Le piège classique

Vous tomberez rapidement dessus, et rassurez-vous, vous ne serez pas le premier. Essayez par exemple de tracer le graphe de la fonction $x \to xsin(x)$ sur $[0,2\pi]$. Fort des deux sections précédentes, vous serez tenté d'écrire :

```
>> x = 0:0.1:1;
>> plot(x, x*sin(x))

??? Error using ==> *
Inner matrix dimensions must agree.
```

L'erreur générée par MATLAB provient du fait que pour MATLAB vous manipulez des tableaux, et ces tableaux sont de taille différente (des vecteurs lignes). Il aurait donc fallu utiliser la multiplication terme à terme, soit :

```
>> plot(x, x.*sin(x))
```

et ça marche ! On peut donc énoncer la règle suivante :

Rappel :

Lorsque l'on écrit une expression arithmétique dans les argument de `plot`, il faut utiliser systématiquement les opérateurs terme à terme `.* ./` et `.^` au lieu de `* /` et `^`

d. Attributs de courbes

Vous aurez remarqué que MATLAB attribue des couleurs par défaut aux courbes. Il est possible de modifier la couleur, le style du trait et celui des points, en spécifiant après chaque couple (abscisse, ordonnée) une chaîne de caractères (entre quotes) pouvant contenir les codes suivants (obtenus en tapant **help plot**) :

Couleurs	
y	yellow
m	magenta
c	cyan
r	red
g	green
b	blue
w	white
k	black

Styles de points	
.	point
o	circle
x	x-mark
+	plus
*	star
s	square
d	diamond
v	triangle (down)
^	triangle (up)
<	triangle (left)
>	triangle (right)
p	pentagram
h	hexagram

Styles de lignes	
-	solid
:	dotted
-.	dashdot
-	dashed

Codes à spécifier pour modifier les attributs de courbes

Lorsque l'on utilise seulement un style de points, MATLAB ne trace plus de droites entre les points successifs, mais seulement les points eux-même. Ceci peut être pratique par exemple pour présenter des résultats expérimentaux.

Les codes peuvent être combinés entre eux. Par exemple

```
>> plot(x,sin(x),':',x,cos(x),'r-.')
```

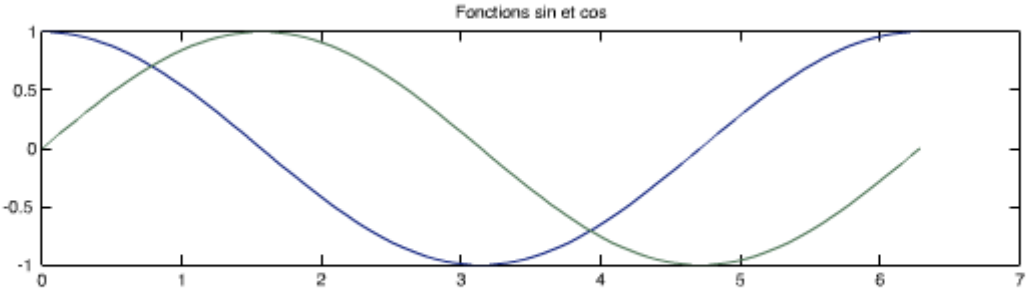
---

2. Décoration des graphiques

a. Titre

C'est l'instruction `title` à laquelle il faut fournir une chaîne de caractères. Le titre apparaît en haut de la fenêtre graphique :

```
>> x = 0: 2*pi/100 : 2*pi;
>> plot(x,cos(x),x, sin(x))
>> title('Fonctions sin et cos')
```

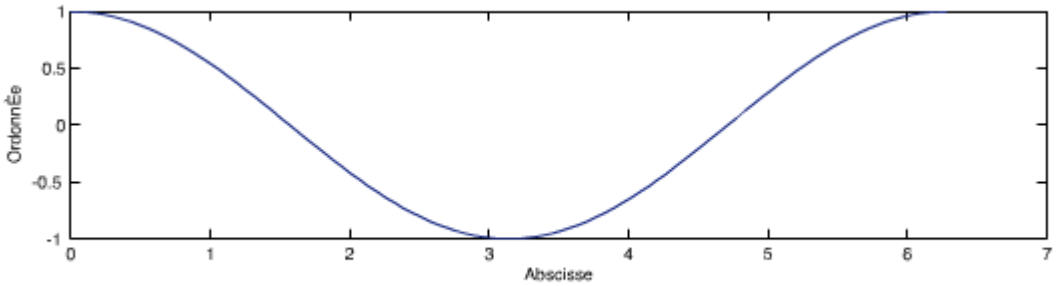


Courbe titrée avec la commande title

Labels

Il s'agit d'afficher quelque chose sous les abscisses et à coté de l'axe des ordonnées :

```
>> x = 0: 2*pi/100 : 2*pi;
>> plot(x, cos(x))
>> xlabel('Abscisse')
>> ylabel('Ordonnée')
```

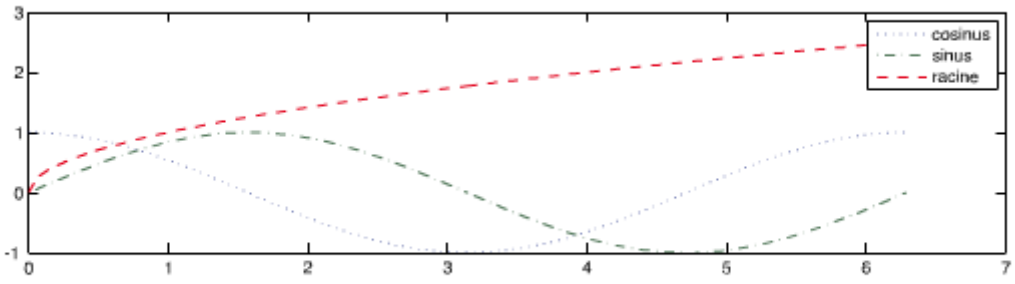


Labels d'une courbe avec les commandes xlabel et ylabel

b. Légendes

C'est l'instruction **legend**. Il faut lui communiquer autant de chaînes de caractères que de courbes tracées à l'écran. Un cadre est alors tracé au milieu du graphique, qui affiche en face du style de chaque courbe, le texte correspondant. Par exemple :

```
>> x = 0: 2*pi/100 : 2*pi;
 >> plot(x,cos(x),':',x,sin(x),'-.',x,sqrt(x),'--')
 >> legend('cosinus','sinus','racine')
```



Légendes des courbes avec la commande legend

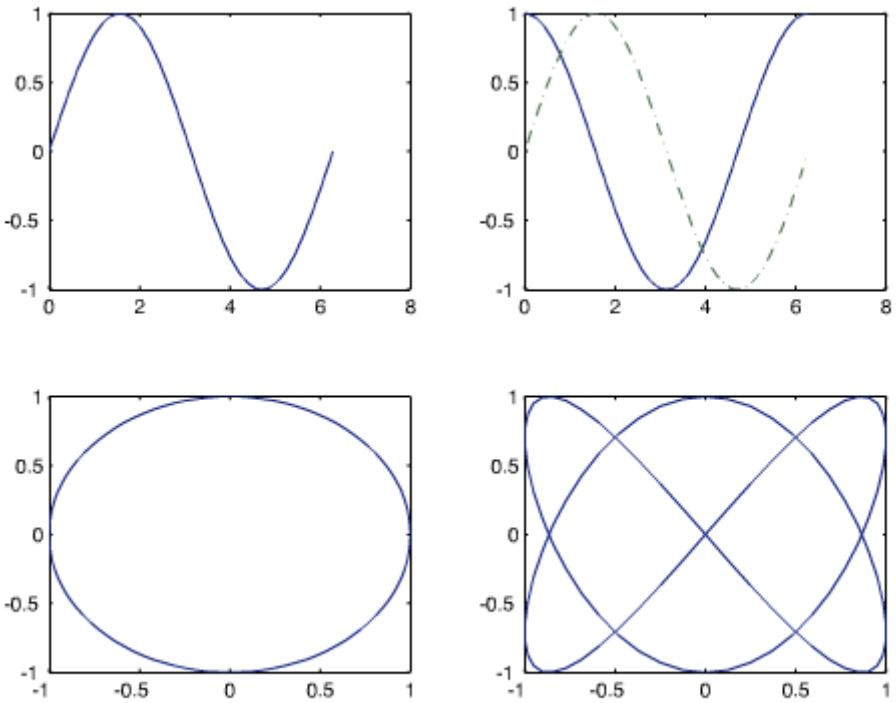
c. Tracer un quadrillage

C'est l'instruction **grid**, qui utilisée après une instruction **plot** affiche un quadrillage sur la courbe. Si on tape à nouveau **grid**, le quadrillage disparaît.

---

3. Afficher plusieurs graphiques (subplot)

Voilà une fonctionnalité très utile pour présenter sur une même page graphique un grand nombre de résultats, par exemple :



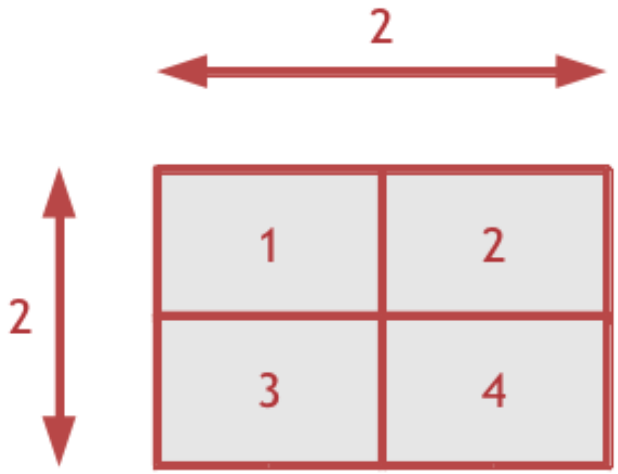
Plusieurs graphiques avec la commande subplot

L'idée générale est de découper la fenêtre graphique en pavés de même taille, et d'afficher un graphe dans chaque pavé. On utilise l'instruction **`subplot`** en lui spécifiant le nombre de

pavés sur la hauteur, le nombre de pavés sur la largeur, et le numéro du pavé dans lequel on va tracer :

subplot (*Nbre pavés sur hauteur, Nbre pavés sur largeur, Numéro pavé*)

La virgule peut être omise. Les pavés sont numérotés dans le sens de la lecture d'un texte : de gauche à droite et de haut en bas :



Schématisation du découpage de la fenêtre graphique avec la commande **subplot**

Une fois que l'on a tapé une commande **subplot**, toutes les commandes graphiques suivantes seront exécutées dans le pavé spécifié. Ainsi, le graphique précédent a été obtenu à partir de la suite d'instructions :

```
>> x = 0: 2*pi/100 : 2*pi;
>> subplot(221)
>> plot(x,sin(x))
>> subplot(222)
>> plot(x,cos(x),x,sin(x),'-.')
>> subplot(223)
>> plot(cos(x),sin(x))
>> subplot(224)
>> plot(sin(2*x),sin(3*x))
```

---

4. Axes et zoom

Il y a deux manières de modifier les valeurs extrêmes sur les axes, autrement dit de faire du zooming sur les courbes. La plus simple est  de la fenêtre graphique. Vous pouvez alors :

- encadrer une zone à zoomer avec le bouton de gauche de la souris

- cliquer sur un point avec le bouton de gauche. Le point cliqué sera le centre du zoom, ce dernier étant effectué avec un facteur arbitraire
- cliquer sur un point avec le bouton de droite pour dézoomer

Pour faire des zooms plus précis,utiliser la commande **axis** (voir la doc).

---

5. Instructions graphiques diverses

a. Maintien du graphique

Par défaut une instruction **plot** efface systématiquement le graphique précédent. Il est parfois utile de le conserver et de venir le surcharger avec la nouvelle courbe. Pour cela on utilise la commande **hold**. Pour démarrer le mode surcharge, taper **hold on**, pour revenir en mode normal, **hold off**.

Il est conseillé de ne pas abuser de cette commande.

b. Effacement de la fenêtre graphique

Tapez simplement **clf**. Cette commande annule également toutes les commandes **subplot** et **hold** passées.

c. Saisie d'un point à la souris

La commande **ginput(N)** permet de cliquer N points dans la fenêtre graphique, et la commande renvoie deux vecteurs, l'un contenant les abscisses, l'autre les ordonnées. Utilisée sans le paramètre N, la commande tourne en boucle jusqu'à ce que la touche «Entrée» soit tapée.