TP 7 : Graphique

Graphique 2D

Traçage de courbes :

Pour afficher des courbes dans une fenêtre graphique, voir les fonctions de help graph2d. On ne présente ici que les fonctions les plus courantes. L'utilisation d'une instruction d'affichage graphique crée automatiquement une fenêtre graphique si aucune fenêtre n'a déjà été créée. On peut également initialiser une fenêtre graphique en faisant appel à l'instruction figure(n) où n est le numéro de la fenêtre. Les fenêtres sont numérotées à partir de 1.

>> help graph2d % intro. au graphisme 2D et tableau des fonctions disponibles

Courbes: plot

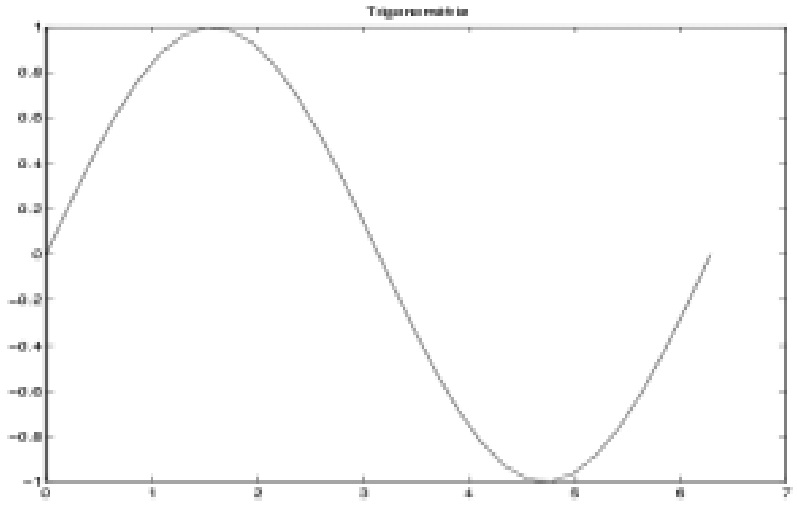
Exemple :

>> x=linspace(0,2*pi,30);

>> y=sin(x);

>> plot(x,y)

>> title('Trigonométrie') % Le titre de la figure



Quelques améliorations:

Un quadrillage (grille), donne plus de lisibilité au graphique. La commande grid off supprime le quadrillage du graphique courant.

>> grid on

Les valeurs de x entre 0 et 2*pi et les valeurs de y entre -1 et 1 : axis([0 2*pi -1 1]) , les deux premiers éléments caractérisent l'axe des abscisses et les deux deuxièmes caractérisent l'axe des ordonnés.

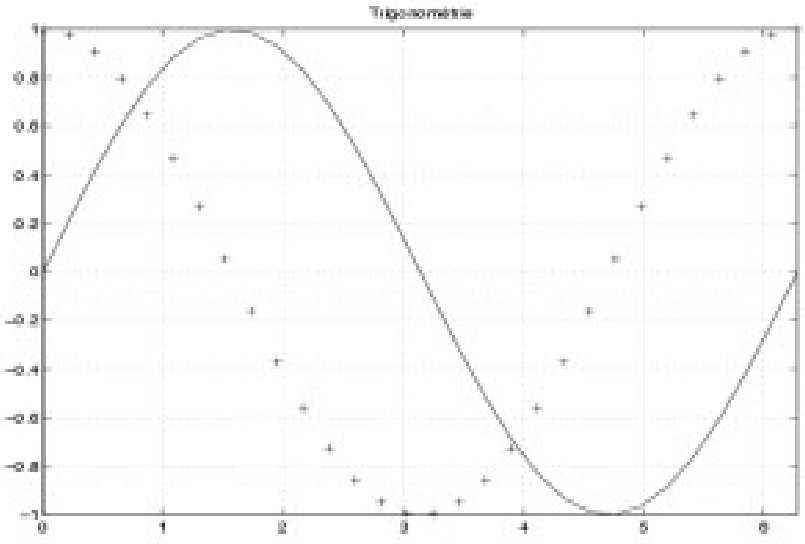
>> axis([0 2*pi -1 1])

% add a second curve

>> hold on % Pour tracer plusieurs courbes sur le même graphique

>> z=cos(x)

>> plot(x,z,'c+')



>> clf % efface la figure

semilogx, semilogy et loglog sont semblables à plot mais permettent de faire des plots log .

Exemple 2 :

Représenter la fonction f(x)=x2 (parabole) sur [-4,4],

>> a=[-4 : 4] ; b= a.^2 ; plot(a,b)

Une diminution du pas permet d'augmenter le nombre de points et avoir une courbe parabolique plus claire :

>> a=[-4 :0.01 : 4] ; b= a.^2 ; plot(a , b)

On peut ajouter à plot un troisième paramètre pour indiquer la couleur et le type de tracé (continu ou discontinu) de la courbe. Ce paramètre est une chaîne formée au plus de trois caractères.

Couleurs : y (jaune), m (violet), r (rouge), g (vert), b (bleu), w (blanc), k (noir)

Type de ligne : . o + * - -. –

Exemple :

>>x=[0 : 0.1 : 4*pi] ; y=sin(x) ; plot(x , y,'*k')

Le titre de l'axe des abscisses :

>> xlabel('x') ;

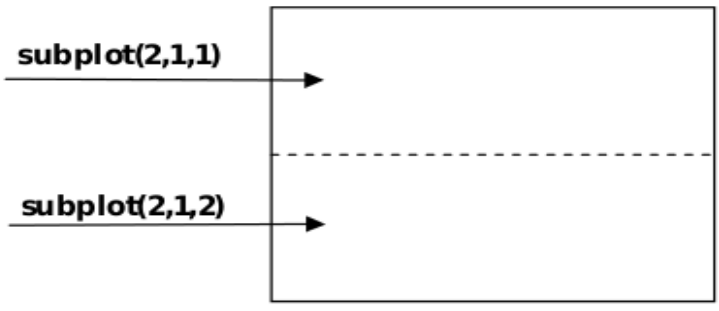
Le titre de l'axe des ordonnée :

>> ylabel('y') ;

Graphique multiple :

On peut tracer plusieurs graphiques dans la même fenêtre en utilisant l'instruction **subplot** pour diviser la fenêtre en plusieurs parties.

Diviser la fenêtre en deux parties (2 x 1) :

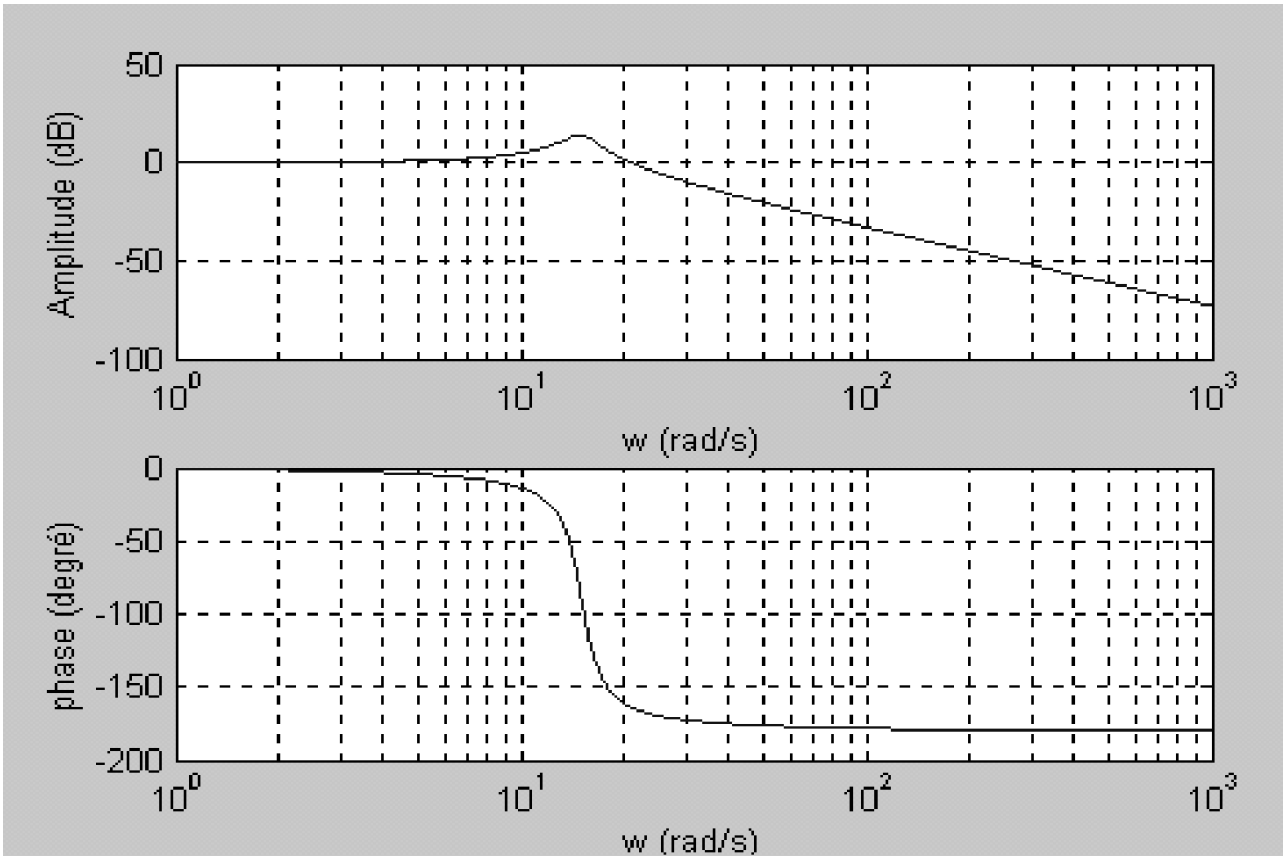


Exemple :

représenter les diagrammes de Bode du système linéaire ayant pour fonction de transfert : H(p) = 225/(p2 + 3.p +225) , ce qui revient à déterminer le gain (en Décibel) et la phase en fonction de la pulsation ω.

Taper une à une les instructions suivantes suivies d'une validation :

```
>>w = logspace(0,3,1000); p=j*w;
>>H=225./(p.^2+3*p+225); AdB=20*log10(abs(H)); phase=angle(H)*(180/pi);
>>subplot(2,1,1), semilogx(w,AdB),grid, xlabel('w (rad/s)') , ylabel('Amplitude (dB)')
>>subplot(2,1,2), semilogx(w,phase),grid, xlabel('w(rad/s)'), ylabel('Amplitude (dB)')
```



Remarques

- Pour des raisons d'explication, on est parfois ramené à ajouter du texte à un endroit bien

précis de la courbe, ceci peut être réalisé ou bien en utilisant la souris, la commande est alors **gtext('** Texte ou valeur à ajouter **')** ou bien en précisant les coordonnés du début du texte, la commande est alors **text(x,y, '** Texte à ajouter **')**.

- La commande **logspace(a,b)** génère un vecteur de 50 valeurs logarithmiques équidistants entre les deux décades 10^a et 10^b, on peut imposer le nombre de valeurs n : **logspace(a,b,n)**.
- **semilogx(w,AdB)** nous a permis de tracer AdB(w) avec une échelle log(w) (de même pour **semilogx(w,phase)**)

subplot(m , n , p) :

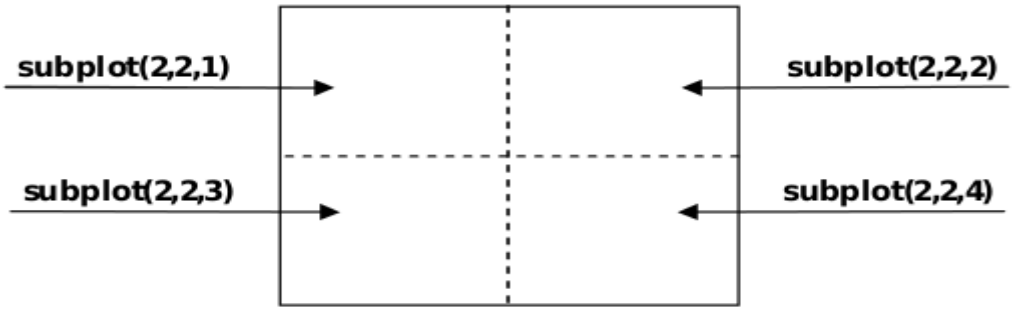
divise la fenêtre graphique courante en (m x n) zones graphiques (m lignes et n colonnes) et trace le graphique

qui suit cette instruction dans la zone de numéro p (la numérotation se fait de gauche

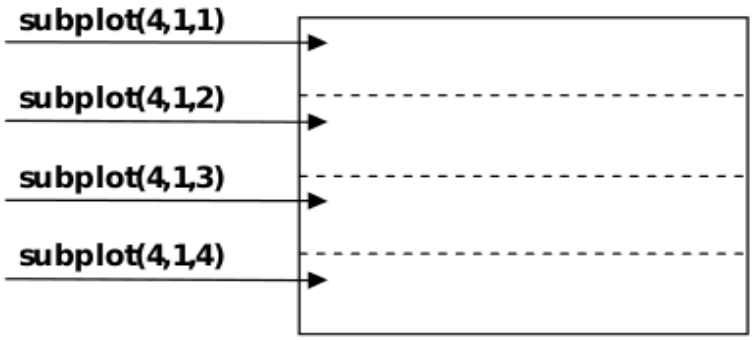
à droite et ligne par ligne).

Diviser la fenêtre en deux parties (1 x 2) :

Diviser la fenêtre en quatre parties (2 x 2) :



Diviser la fenêtre en quatre parties (4 x 2) :

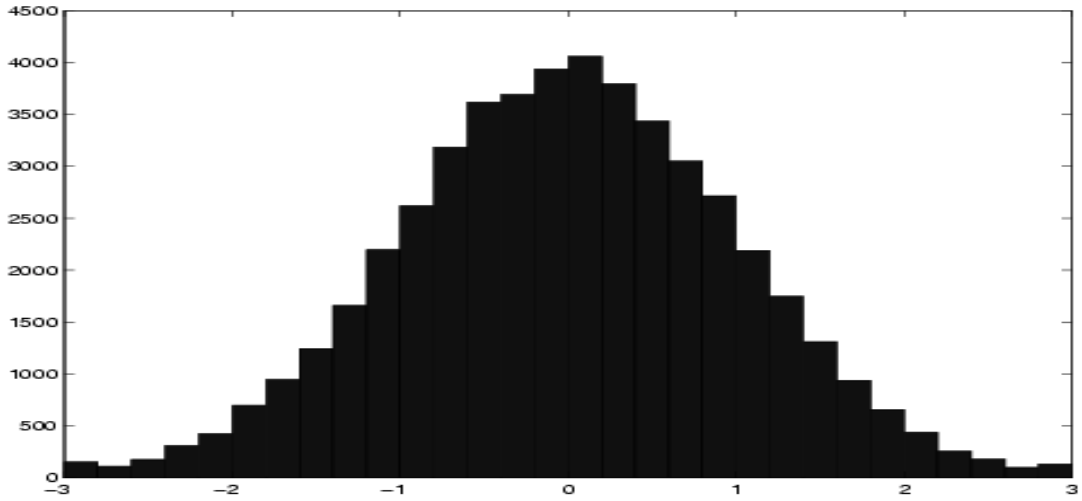


histogrammes: hist

>> x=-2.9 : 0.2 : 2.9; % défini l'intervalle et la largeurs des canaux de l'histogramme

>> y=randn(50000,1); % génère des nombres aléatoires répartis selon une distr. normale

>> hist(y,x) % dessine l'histogramme



Graphiques à 3D

>> help graph3d % introduction au graphisme 3D et tableau des fonctions disponibles

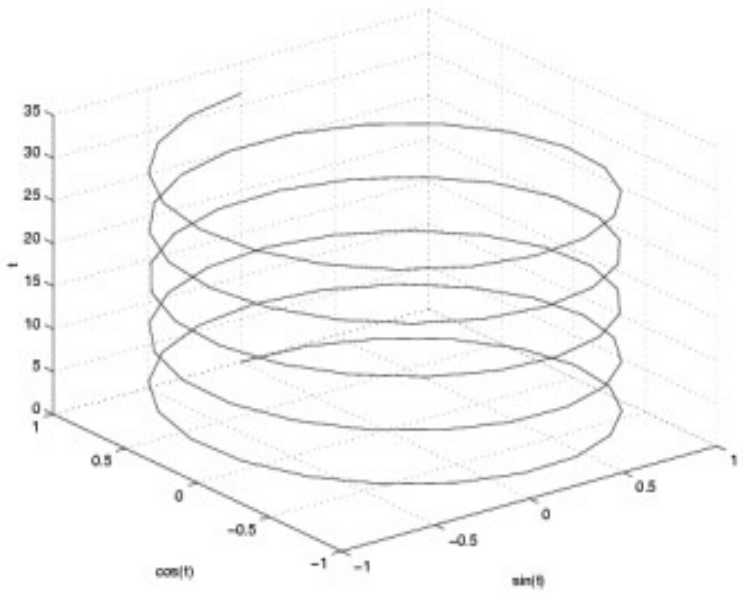
ligne dans l'espace: plot3

>> t = linspace(0, 10*pi);

>> plot3(sin(t), cos(t), t)

>> xlabel('sin(t)'), ylabel('cos(t)'), zlabel('t')

>> grid on



Construction d'un maillage dans le plan (x,y) : meshgrid

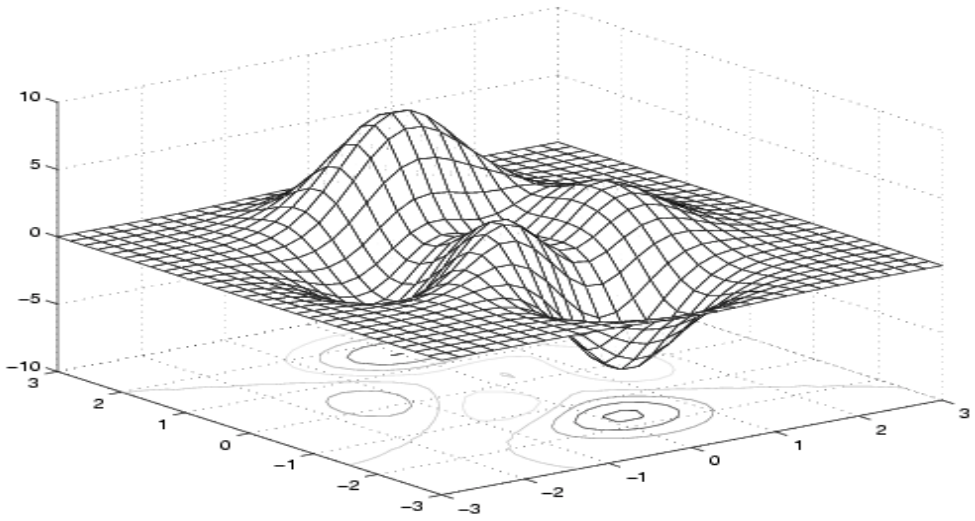
Pour la représention d'une surface f(x, y), on a besoin de connaître les triplets de coordonnées (xi, yj, Zi,j), avec Zi,j=f(xi,yj), pour un certain nombre de points (xi, yj) où i=1, ..., N, j=1, ..., M. On voit que Z a la structure d'une matrice {Zi,j}. Par soucis d'harmonisation, Matlab utilise également une représentation matricielle pour les coordonnées x et y dans le plan. Matlab fournit la fonction ***meshgrid*** pour générer les matrices {Xi,j} et {Yi,j} à partir des vecteurs {xi} et {yi}.

Exemple :

```
>> clear
>> x = logspace(0, 2, 3)
>> y = linspace(0, 100, 5)   % meshgrid fournit les matrices X et Y à partir des vecteurs x et y :
>> [X Y] = meshgrid(x, y)   % La maille (4, 2) est donnee par :
>> P_4_2 = [X(4,2) ; Y(4,2)]
```

grillage en perspective: mesh

```
>> clf
>> x = linspace(-3, 3, 30);
>> y = linspace(-3, 3, 30);
>> [X Y] = meshgrid(x,y);
>> Z = peaks(X, Y)   % peaks est une fonction à 2-D prédéfinie dans matlab
>> mesh(X, Y, Z)    % grillage 3D
>> meshc(X, Y, Z)   % grillage 3D avec les contours sur le plan de base
```



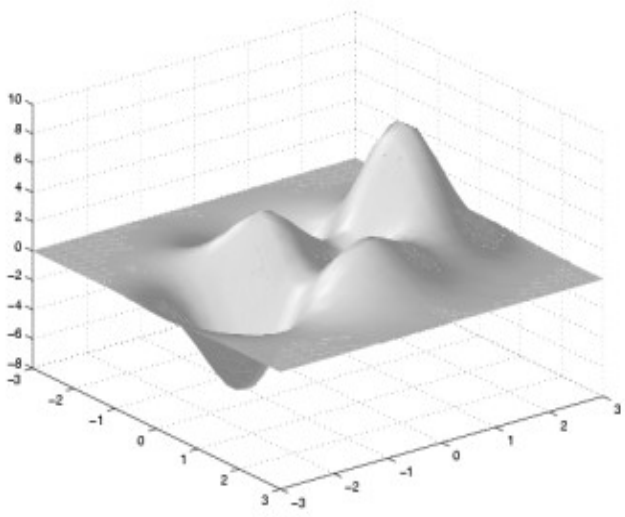
Surface avec illumination: surfl

```
>> surfl(X,Y,Z)   % graphique avec illumination
```

>> shading interp % meilleure interpolation

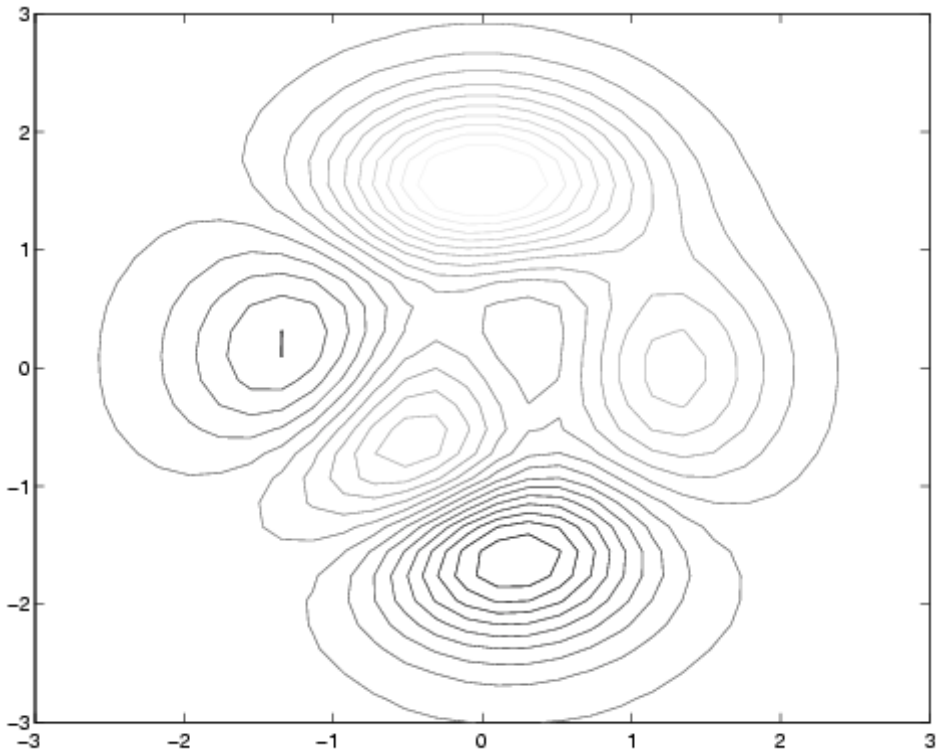
>> colormap pink %choix d'une palette de couleurs prédéfinie

>> view(-37.5+90, 30) % changement point de vue



Courbes de niveau: contour

>> contour(X,Y,Z,20) % 20 lignes de niveau



% pour transformer ce graphe en échelle de couleurs :

>> colormap(’gray’)

>> pcolor(X,Y,Z) % plot pseudo-couleur de Z sur la grille X,Y

>> shading interp

>> axis(’square’)

>> colormap(’default’)