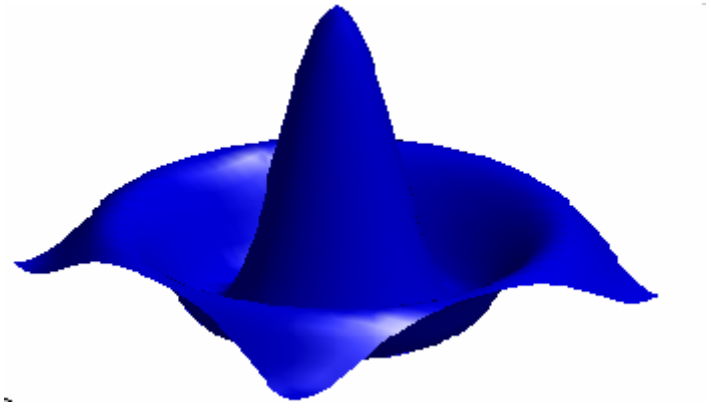
Graphisme sous Matlab



Florence Nicol
Unité de Neurosciences Cognitives
Université Catholique de Louvain
Louvain-la-Neuve, Belgique
florence.nicol@psp.ucl.ac.be
08 Avril 2004

Fonctions graphiques

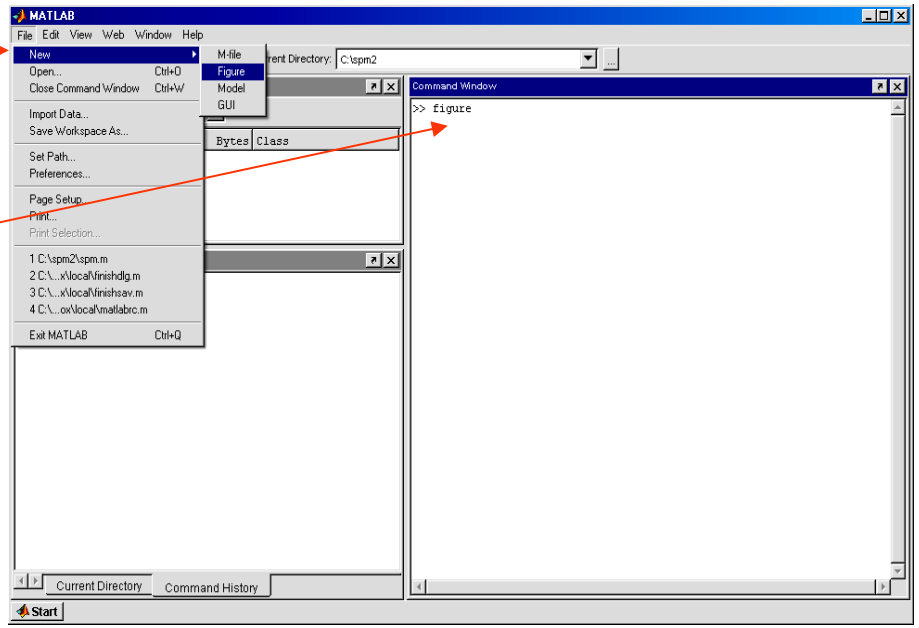
- Interface graphique puissante :
  - Visualiser les vecteurs et matrices sous forme de graphiques en utilisant des commandes pré installées dans Matlab
  - Possibilité d'annoter et d'imprimer ces graphiques
- Graphiques en 2D et en 3D
  - 2D : commandes *plot*, *fplot*, *loglog*, …
  - 3D : commandes *mesh*, *surf*, *image*, *contourslice*, …

Gestion des fenêtres graphiques

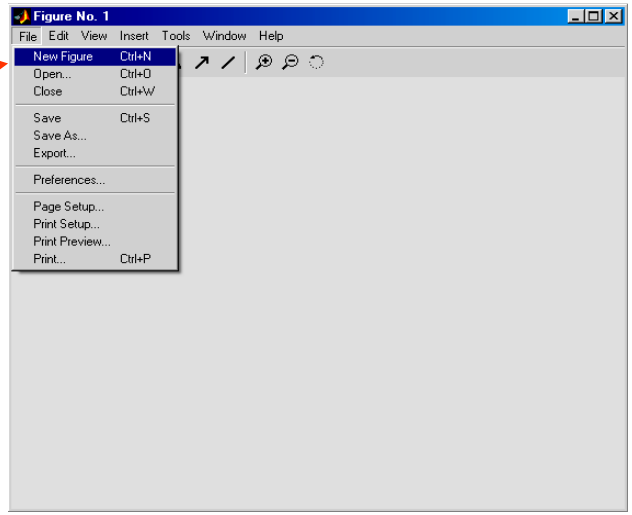
- Une commande graphique ouvre une fenêtre dans laquelle sera affichée le graphique voulu.
- Si une fenêtre est déjà ouverte, une nouvelle instruction graphique écrasera le graphique précédent.
- Si plusieurs fenêtres sont ouvertes, Matlab utilise la dernière fenêtre active.
- Chaque fenêtre graphique ouverte se voit affectée un numéro (visible dans le bandeau de la fenêtre).

Ouvrir une nouvelle fenêtre

- Menu File>New>Figure
- Commande *figure*



- Fenêtre graphique :
  Menu File>New Figure



Ouverture/fermeture d'une fenêtre graphique

- Ouverture d'une fenêtre :
  - Ouvrir une nouvelle fenêtre :
    - Commande *figure*
    - Commande *figure*(*n*) où *n* est le numéro de la fenêtre
- Activation :
  - Cliquer sur la fenêtre graphique
  - Commandes *figure* et *figure*(*n*) : activent une fenêtre existante
- Fermeture d'une fenêtre :
  - Fermer la fenêtre active :
    - Commande *close*
  - Fermer une fenêtre ouverte :
    - Commande *close*(*n*) où *n* désigne le numéro de la fenêtre
  - Fermer toutes les fenêtres graphiques :
    - Commande *close all*

Graphiques 2D : commande *fplot*

- Graphe d'une fonction : commande *fplot*

  trace le graphe d'une fonction sur un intervalle donné

  Syntaxe : *fplot*('*nomf*',[*xmin*, *xmax, ymin, ymax*])

  nomf = le nom d'une fonction matlab incorporée

  ou une expression définissant une fonction

  xmin, *xmax* déterminent l'intervalle des abscisses pour lequel est tracé le graphe

  ymin, *ymax* déterminent l'intervalle des ordonnées

  par défaut : les valeurs minimum et maximum prises par la fonction sur l'intervalle [*xmin*, *xmax*]

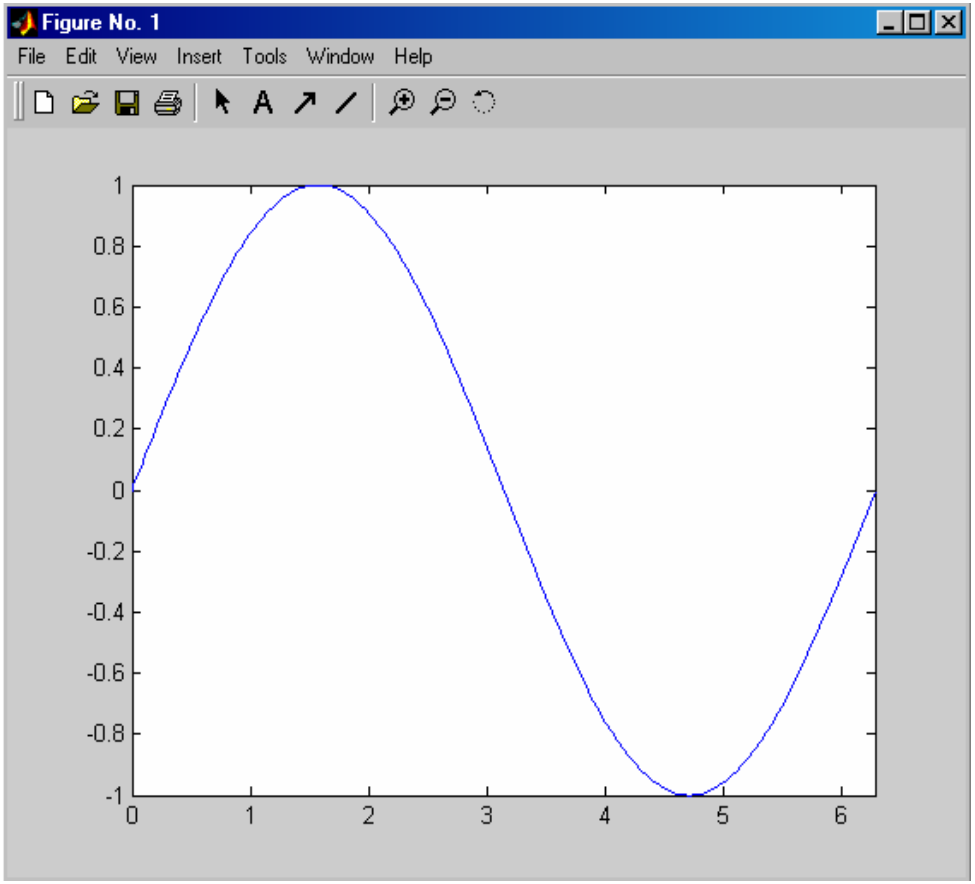
Exemple : commande *fplot*(1)

- Fonction sinus :

  int = [0,2*pi];
  fplot('sin',int)

ou

  fplot('sin',[0,2*pi])



Exemple : commande *fplot*(2)

- Fonction x*sinus(x) :

  fplot('x*sin(x)',[-2*pi,2*pi])

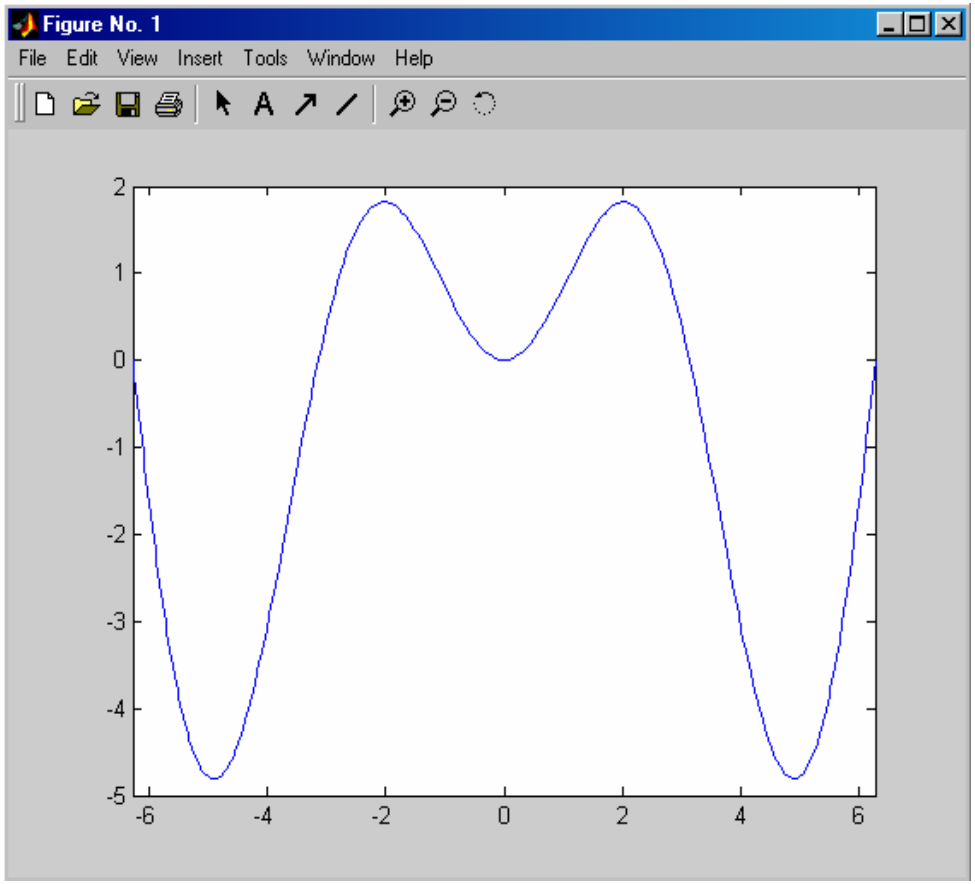
- Définir une fonction :

  – Ecrire une fonction h.m :

    function y=h(x)

    y=x.*sin(x);

  – fplot('h',[-2*pi,2*pi])



Graphiques 2D : commande *plot*

- Graphe d'une fonction : commande *plot*
  - trace le graphe d'un vecteur *y* pour un ensemble de valeurs *x* spécifiées
  - par défaut, relie les points par des segments de droite

Syntaxe : *plot*(*x*,*y*)

y = vecteur contenant les valeurs en ordonnées

x = vecteur contenant les valeurs en abscisses

x et *y* sont des vecteurs (ligne ou colonne) de même longueur !

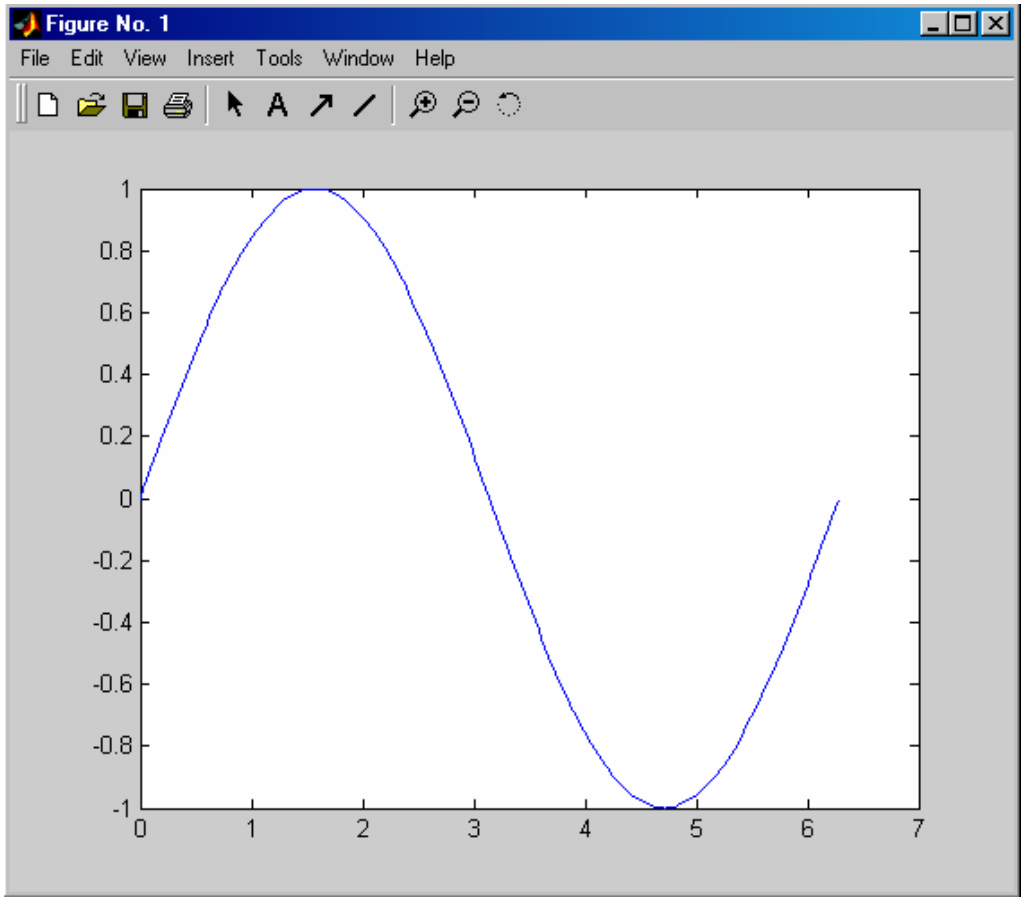
Exemple : commande *plot*(1)

- Fonction sinus :

Intervalle [0,2pi], 201 valeurs

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

- Sélection automatique de l'étendue des axes



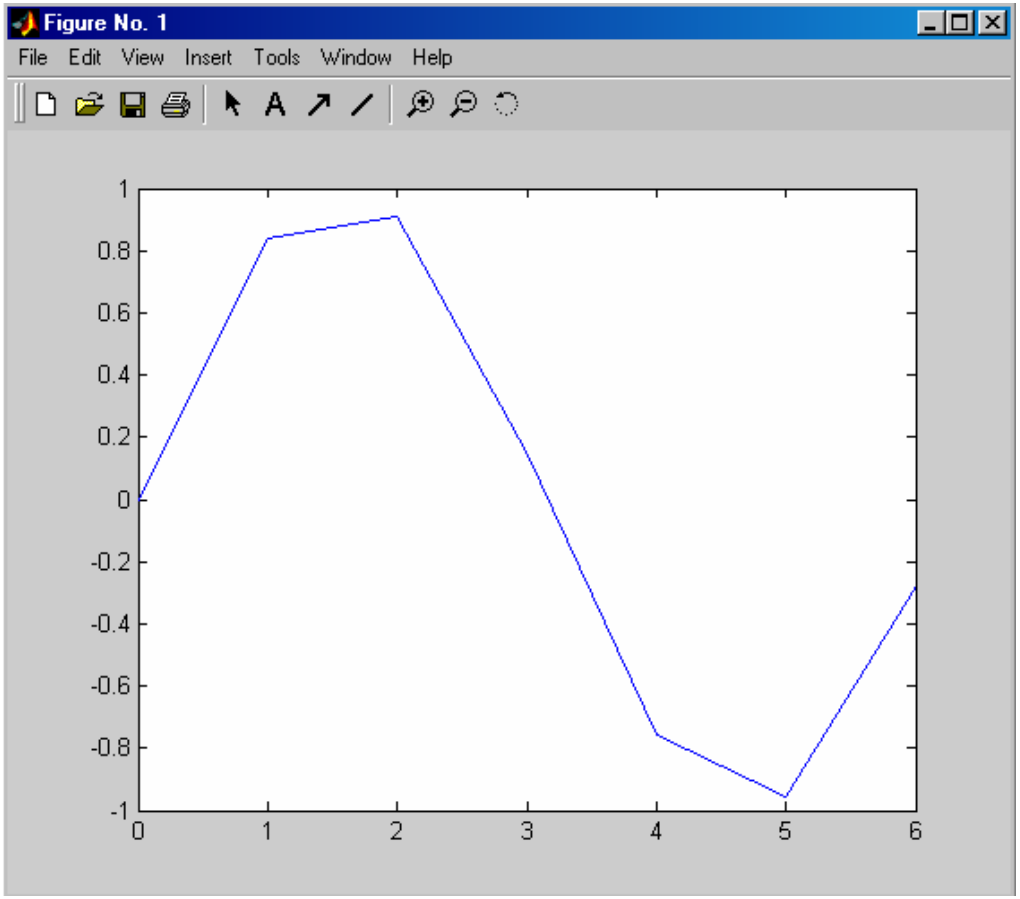
Exemple : commande *plot*(2)

- Fonction sinus :

Intervalle [0,2pi], 7 valeurs

```
x= 0:1:2*pi;
y = sin(x);
plot(x,y);
```

- Points reliés linéairement



Plusieurs courbes sur un seul graphique(1)

- Plusieurs courbes pour le même vecteur d'abscisse *x* :
  - commande *fplot* :

    fplot('[*nomf1*, *nomf2*]',[*xmin*, *xmax*, *ymin*, *ymax*])

    xmin, *xmax* déterminent l'intervalle des abscisses pour lequel les deux graphes sont tracés

  - commande *plot* :

    plot(x,y_1,x,y_2)

    x = vecteur d'abscisses pour les courbes y_1 et y_2

Plusieurs courbes sur un seul graphique(2)

- Plusieurs courbes pour différents vecteurs d'abscisse *x* :
  - commande *plot* :

$plot(x_1,y_1,x_2,y_2)$

x_1 = vecteur d'abscisses pour la courbe y_1
x_2 = vecteur d'abscisses pour la courbe y_2

  - commandes *hold on* et *hold off :*

hold on
……..
instructions graphiques
……..
hold off

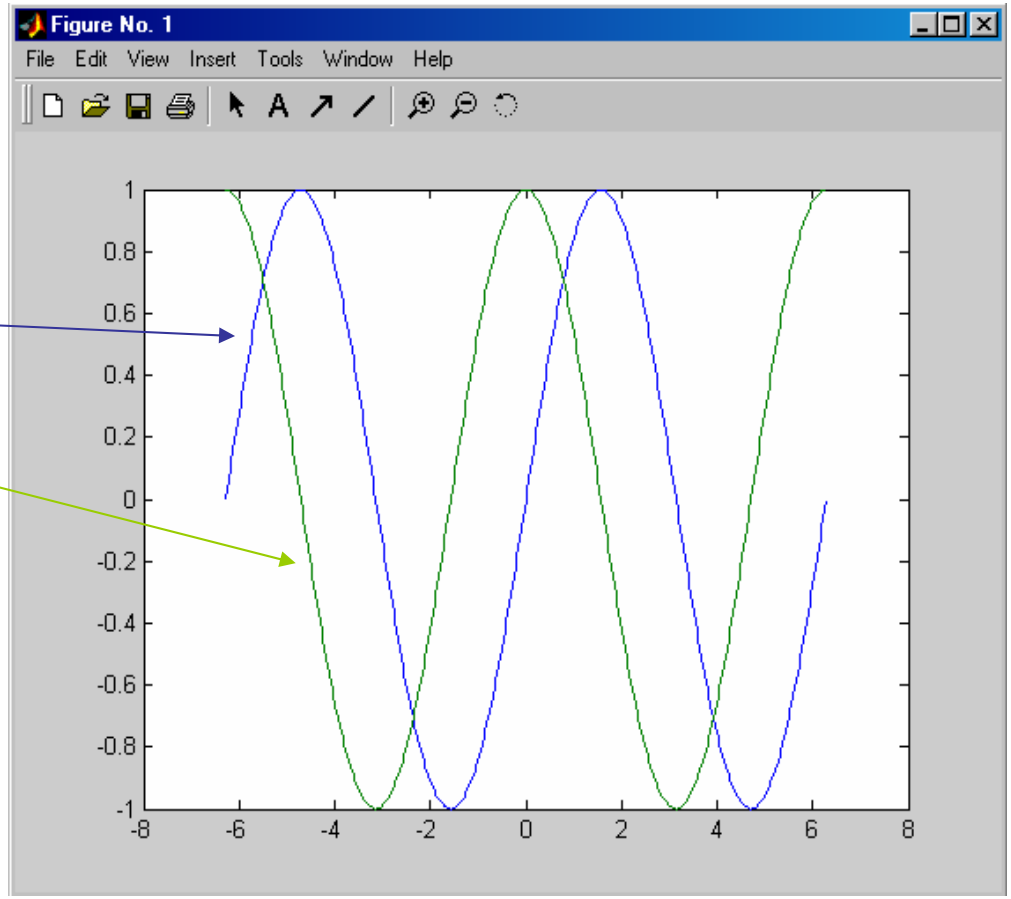
} graphes superposés dans la même fenêtre active

» utile dans une boucle (cf. programmation)
» utile pour la commande *fplot*

Plusieurs courbes sur un seul graphique : exemple(1)

- Fonctions sin et cos :
  intervalle [-2pi,2pi], 201 valeurs

```
x = -2*pi:pi/100:2*pi;
y1 = sin(x);
y2 = cos(x);
plot(x,y1,x,y2)
```



Plusieurs courbes sur un seul graphique : exemple(2)

- Fonctions sin et cos :
  intervalle [-2pi,2pi], 201 valeurs
  intervalle [0,2pi], 101 valeurs

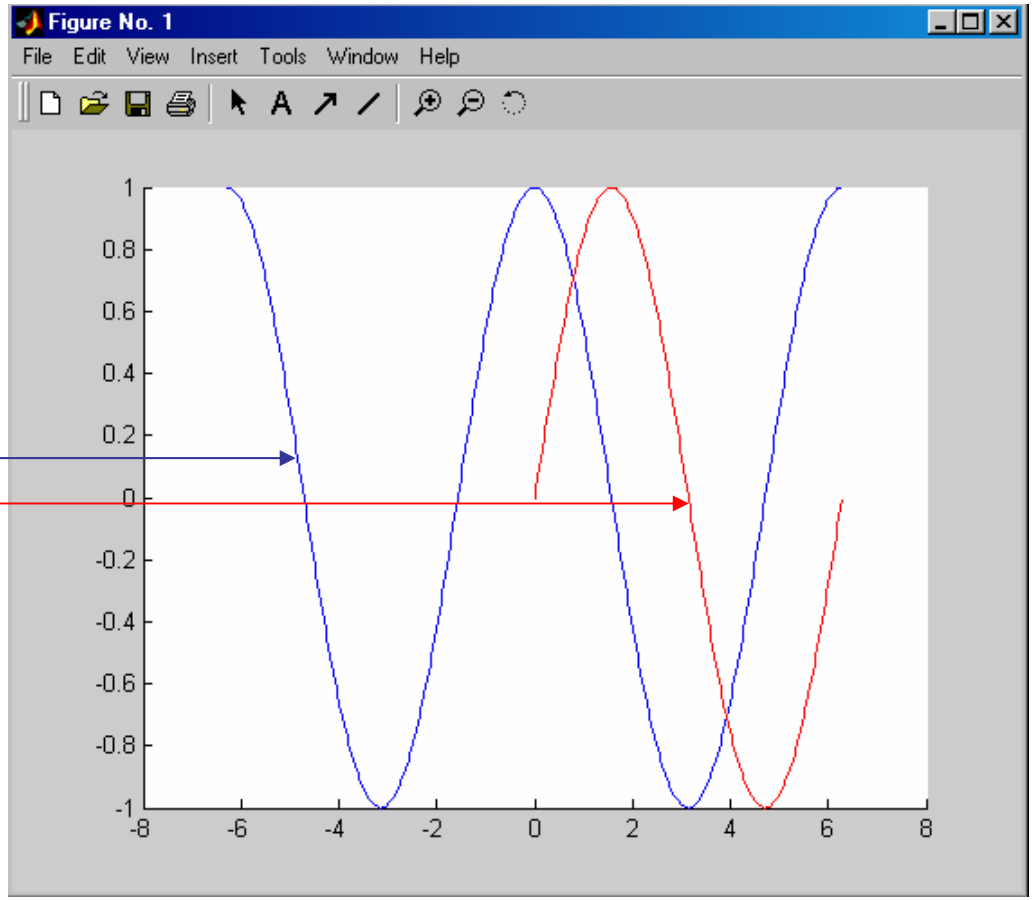
```
x1 = -2*pi:pi/100:2*pi;
y1 = cos(x1);
x2 = 0:pi/100:2*pi;
y2 = sin(x2);

hold on;
plot(x1,y1);
plot(x2,y2,'r');
hold off;
```

ou

```
hold on;
fplot('cos',[-2*pi,2*pi];
fplot('sin',[0,2*pi],'r');
hold off;
```

ou

```
plot(x1,y1,x2,y2,'r');
```



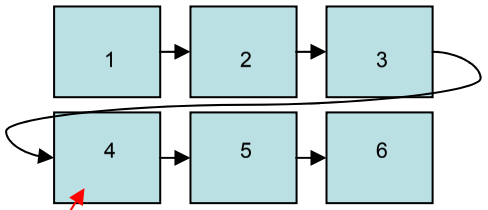
Plusieurs graphiques dans une fenêtre

- Commande *subplot* : décomposer une fenêtre en sous-fenêtres

subplot(*m*,*n*,*i*);
instruction graphique

m = nombre de sous-fenêtres verticalement
n = nombre de sous-fenêtres horizontalement
i = numéro de la sous-fenêtre dans laquelle le graphique s'affiche

Numérotation de gauche à droite, de haut en bas

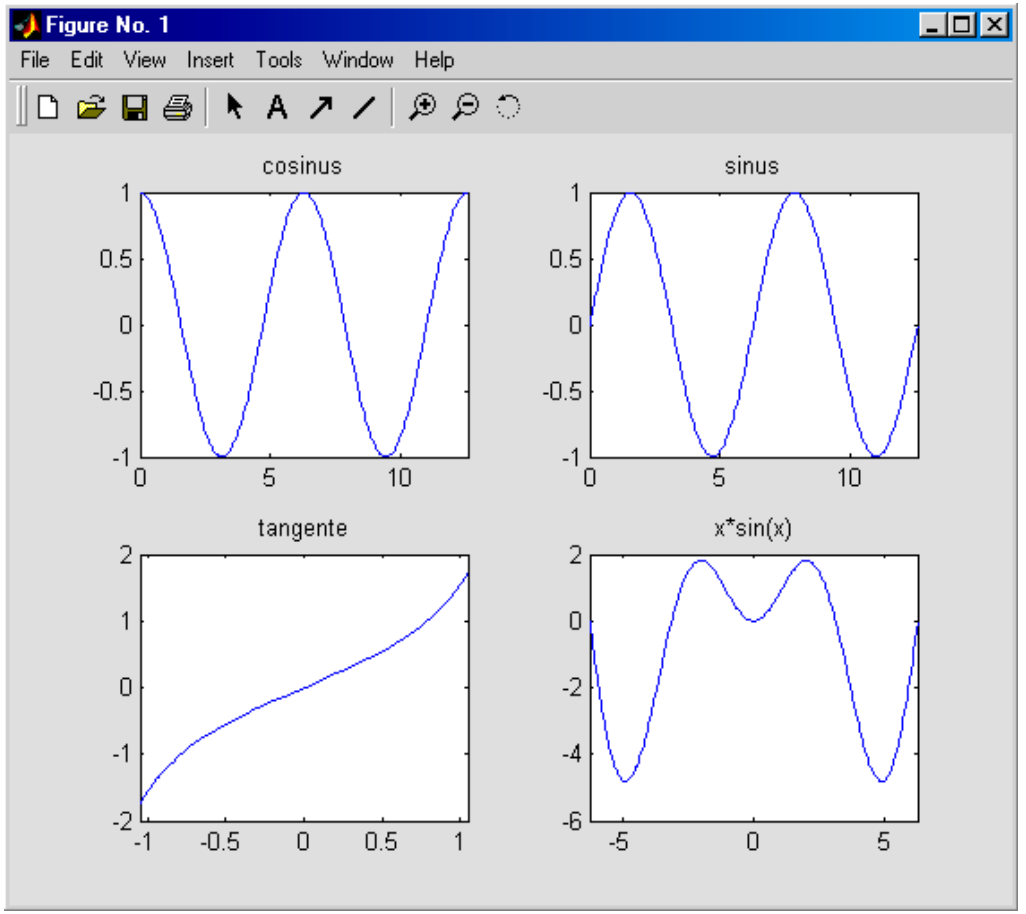


Ex : subplot(2,3,4);

Plusieurs graphiques dans une fenêtre : exemple

- Fonctions cos(x), sin(x), tangente(x), x*sin(x):

```
figure;
subplot(2,2,1);
fplot('cos',[0,4*pi]);
title('cosinus');
subplot(2,2,2);
fplot('sin',[0,4*pi]);
title('sinus');
subplot(2,2,3);
fplot('tan',[-pi/3,pi/3]);
title('tangente');
subplot(2,2,4);
fplot('x*sin(x)',[-2*pi,2*pi]);
title('x*sin(x)');
```



Améliorer la lisibilité

- Créer des légendes et annotations
- Contrôler les axes
- Options du tracé des courbes

Textes et Légendes(1)

- Légendes des axes
  - Commande *xlabel* :

    xlabel(‘légende de l’axe x’)
  - Commande ylabel :

    ylabel(‘légende de l’axe y’)
- Titre du graphique
  - Commande *title* :

    title(‘titre du graphique’)

Textes et Légendes(2)

- Texte dans la figure :
  - Commande *text* : écrit un texte à une position précise sur la figure

    text(*posx*,*posy*,'un texte')

    posx et *posy* = coordonnées du point de début de texte
  - Commande *gtext* : écrit un texte à une position choisie à l'aide de la souris

    gtext('un texte')

  Remarque : texte entre côtes avec la notation *Tex*

Textes et Légendes(3)

- Identification des courbes dans un même graphique :
  - Commande *legend* : légende permettant d'identifier les courbes

legend(*'légende 1'*,*'légende 2'*,*'légende 3'*)

Contrôler les axes(1)

commande *axis* = orientation et échelle des plots

- Limiter les axes : par défaut, l'étendue varie du minimum au maximum

  axis([*xmin xmax ymin ymax*])

  axis auto = revenir à la sélection par défaut

- Aspect des axes : changement d'échelle

  axis square = axes *x* et *y* de même longueur

  axis equal = pas de même longueur sur les axes *x* et *y*

  axis normal = revenir à la sélection par défaut

- Autres options : *help axis*

Contrôler les axes(2)

- Visibilité :

  axis on = rend les axes visibles (par défaut)

  axis off = rend les axes invisibles

- Grille de lignes :

  grid on = affiche une grille de lignes

  grid off = efface la grille de lignes (par défaut)

Options du tracé des courbes(1)

- Spécifier les couleurs
- Spécifier le style de trait
- Spécifier le symbole à chaque point

plot(*x*,*y*,'*color_style_marker*','*LineWidth*',n)

color_style_marker = chaîne de 3 à 4 caractères définissant la couleur, le style du trait et le symbole

n = épaisseur du trait (par défaut 1)

- Couleur du fond définis par les axes : commande *whitebg('couleur')*
  - Modifie la couleur les propriétés du graphique (couleur des axes, du tracé, …) pour maintenir un contraste adéquat

Options du tracé des courbes(2)

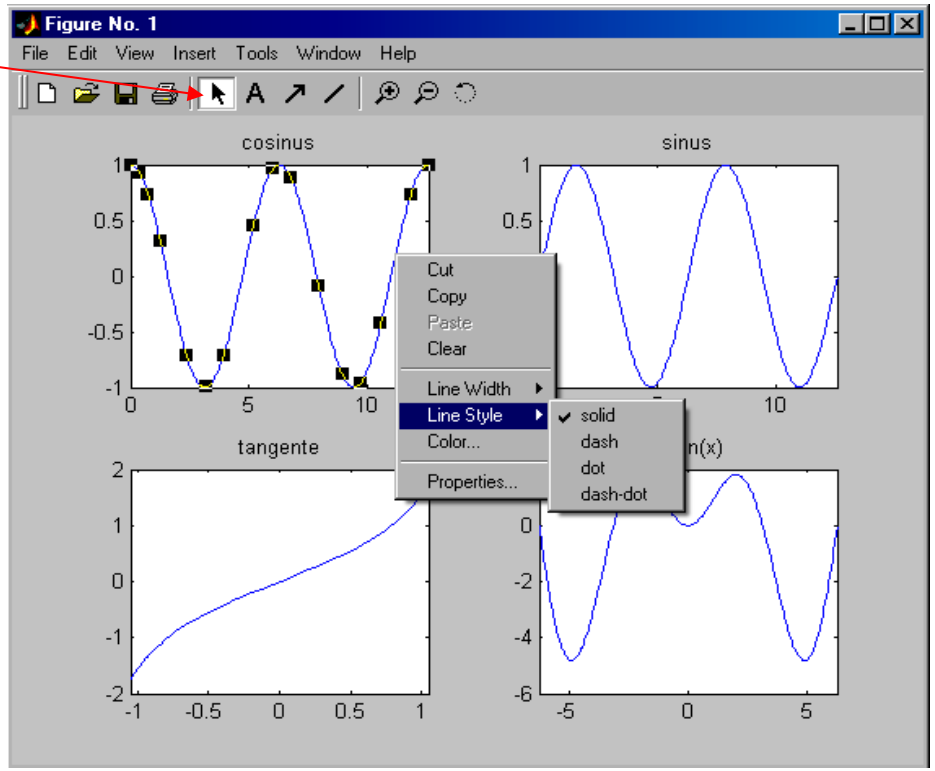
Couleurs	Style	
y = jaune	-	trait plein
m = magenta	:	pointillé court
c = cyan	--	pointillé long
r = rouge	-.	pointillé mixte
g = vert	none	pas de ligne
b = bleu		
w = blanc		
k = noir		

Symbole	
.	point
o	cercle
x	croix
+	plus
*	étoile
s	carré
d	losange
v	triangle (bas)
^	triangle (haut)
<	triangle (gauche)
>	triangle (droite)
p	pentagone
h	hexagone
none	aucun

Valeur par défaut : ‘b-.’ = bleu, trait plein, point

Modifier les plots après création

- Mode édition interactive
  - Clic droit sur l'objet à modifier :
    - Épaisseur de ligne
    - Style de ligne
    - Couleur
    - Propriétés de l'objet



- Commande *set*

  h = *plot*(*x*,*y*);
  set(*h*,'*nomprop*',*valprop*)

  nomprop = désigne la propriété du graphique à modifier
  ex : 'Color' modifie la couleur du fond d'écran
  valprop = valeur de la propriété
  ex : 'r'
  n = numéro de la figure à modifier

- Commande *get(h)* : retrouver les propriétés graphiques

Sauvegarder une figure(1)

- Format figure Matlab « *.fig » :
  - Dans la fenêtre graphique : Menu File>Save nomfig.fig

- Exporter
  - Menu File>Export (tiff, jpeg, eps, …)
  - Commande *saveas* :

```
h=plot(…);  (*)
saveas(h,'nomfich.ext')
saveas(h,'nomfich','format')
```

format ou *ext* =

ai = Adobe Illustrator
eps = EPS level1
tif = TIFF
jpg = JPEG
fig = fichier binaire « nomfich.fig »
m = fichier « nomfich.fig » et crée une fonction matlab de même nom qui permettra d'appeler la figure via l'instruction *nomfich*

(*) Remarque : on peut utiliser alternativement l'instruction *h*=*figure*(*n*) pour sauvegarder l'ensemble des graphiques d'une figure (par exemple, dans le cas de « subplots »)

Sauvegarder une figure(2)

- Exporter
  - Commande *print* :

print –f<*num*> -d<*format*> <*nomfich*>

num = numéro de la fenêtre graphique (par défaut la fenêtre active)

format = format de sauvegarde (tiff, jpeg, ps, eps, epsc, …)

nomfich = nom du fichier du graphique sauvegardé

Exporte le graphique dans le répertoire de travail actif

Imprimer des graphiques

- Imprimer directement ou après exportation du graphique
- Par le menu File :
  - Page Setup = mise en page
  - Print setup = configuration de l'impression
  - Print preview = visualiser l'impression
  - Print = imprimer le graphique
- Commande *print*
  - Commande *print* : imprime directement sur l'imprimante connectée

Graphiques 3D

- Représenter les lignes de niveaux d'une fonction $z = g(x,y)$ sur le domaine plan $[a,b] \times [c,d]$
- Représenter une surface d'équations $z = g(x,y)$ sur le domaine plan $[a,b] \times [c,d]$
- Représenter une surface paramétrée d'équations
- Représenter des volumes d'équations $v = g(x,y,z)$

Préparer les données

- Commande *meshgrid* : création d'un maillage du domaine *[a,b]*$\times$ *[c,d]*, de maille de pas *h*

  [X,Y] = *meshgrid*(*a*:*h*:*b*, *c*:*h*:*d*)

- Evaluer la fonction *g* aux nœuds de maillage

  Z = g(X,Y)

Ex : fonction $z=x \exp(-x^2-y^2)$ sur le domaine [-2,2]$\times$ [-2,2]

avec un maillage de pas *h*=0.2

```
[X,Y] = meshgrid(-2:0.2:2, -2:0.2:2);
Z = X.*exp(-X.^2-Y.^2);
```

Lignes de niveaux(1)

- Syntaxe

  commande *contour*(*X*,*Y*,*Z*,*n*)

  » *n* = nombre de lignes de niveaux à afficher

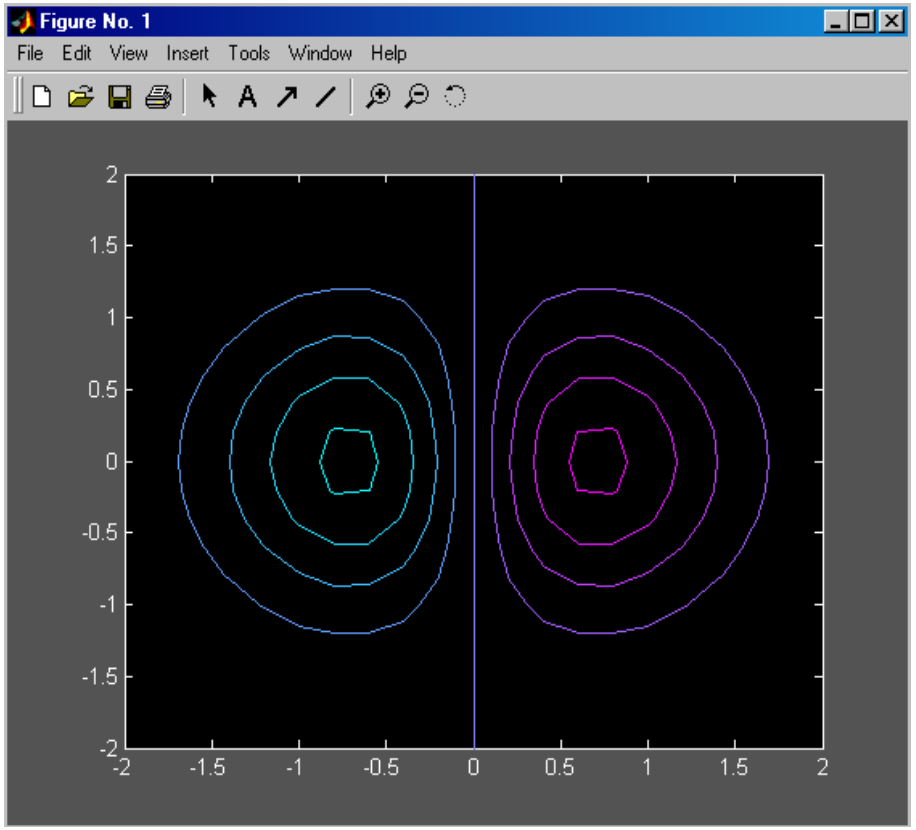
  commande *contour*(*X*,*Y*,*Z*)

  » Sélection automatique du nombre de lignes de niveaux

- Afficher les valeurs des lignes de niveaux : commande *clabel*

  – Toutes les lignes de niveaux :
  [*C*,*h*] = *contour*(*X*,*Y*,*Z*,*n*);
  clabel(*C*,*h*)
  – Quelques lignes de niveaux :
  [*C*,*h*] = *contour*(*X*,*Y*,*Z*,*n*);
  clabel(*C*,*h*,'*manual*')

  Sélection des lignes de niveaux avec la souris



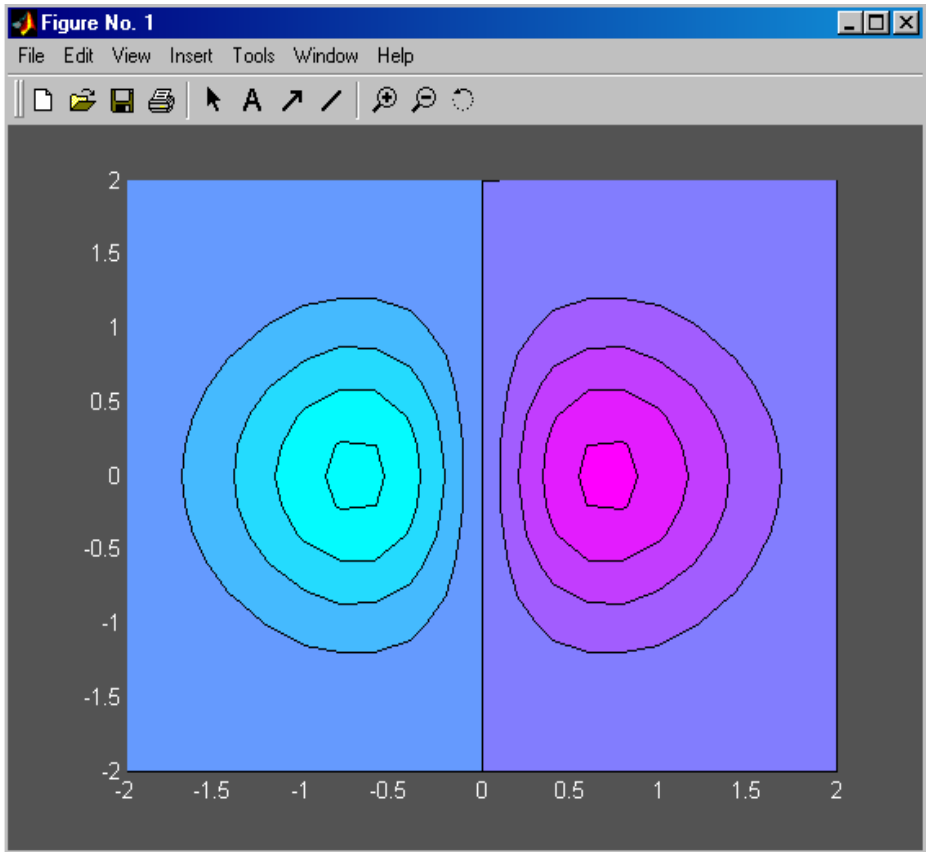
Lignes de niveaux(2)

- Syntaxe

  commande *contourf*(*X*,*Y*,*Z*,*n*)

  n = nombre de lignes de niveaux à afficher

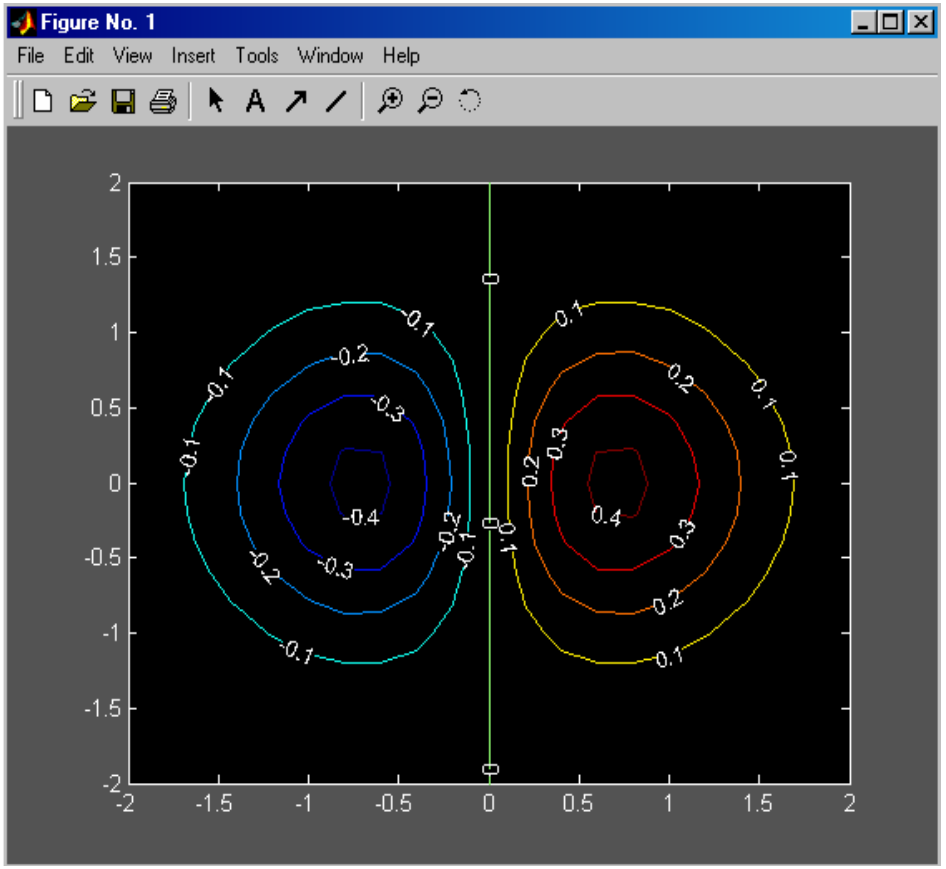
- Affiche en plus de lignes de niveaux, un dégradé continu de couleurs qui varient en fonction des valeurs de la fonction



Lignes de niveaux : exemple

- fonction $z=x \exp(-x^2-y^2)$ sur le domaine $[-2,2]\times[-2,2]$
  avec un maillage de pas h=0.2

```
[X,Y] = meshgrid(-2:0.2:2, -2:0.2:2);
Z = X.*exp(-X.^2-Y.^2);
[C,h] = contour(X,Y,Z);
clabel(C,h);
```

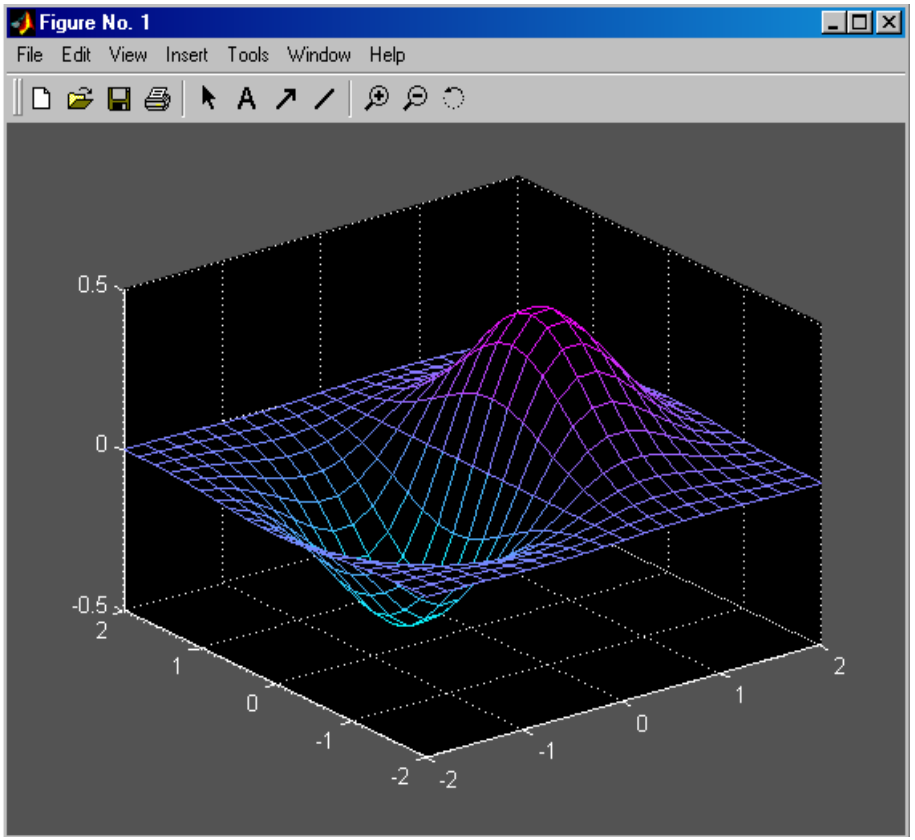


Surface d'équations(1)

- commande *mesh*(*X*,*Y*,*Z*)
  surface de la fonction Z=g(X,Y)

- fonction $z=x\exp(-x^2-y^2)$ sur le domaine $[-2,2]\times[-2,2]$
  avec un maillage de pas h=0.2

  [X,Y] = *meshgrid*(-2:0.2:2, -2:0.2:2);
  Z = X.*exp(-X.^2-Y.^2);
  mesh(X,Y,Z);

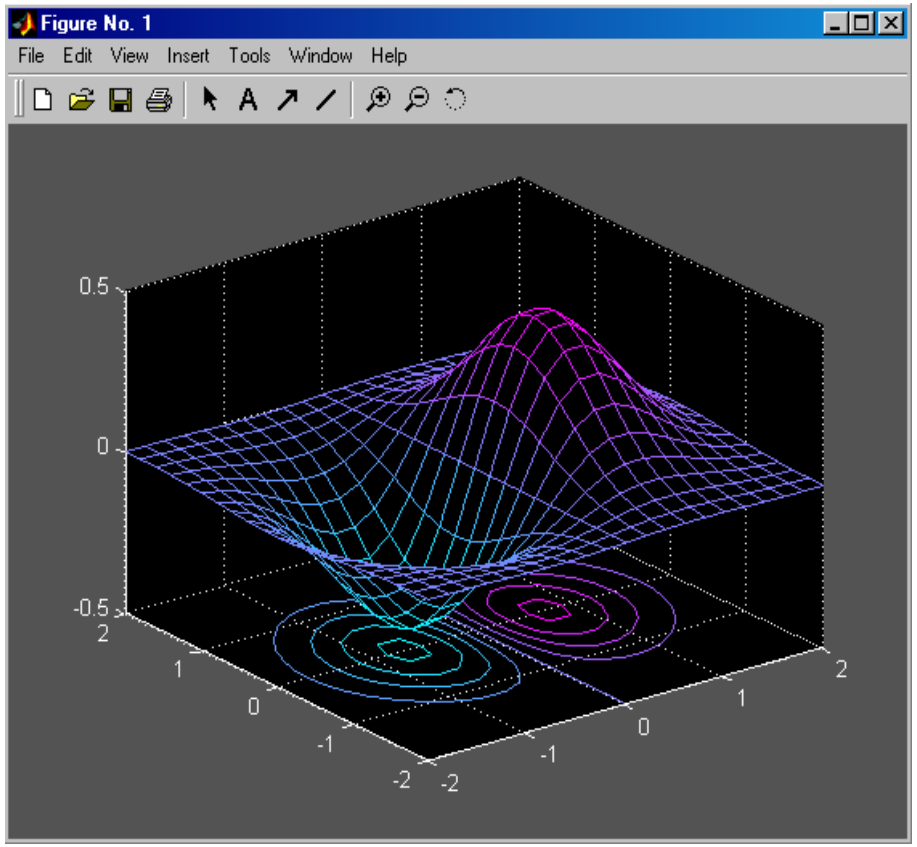


Surface d'équations(2)

- commande *meshc(X,Y,Z)*
  surface de la fonction Z=g(X,Y)
  et projection des contours sur le
  plan défini par [a,b]× [c,d]

- fonction z=x exp(-x^2-y^2) sur le domaine
  [-2,2]× [-2,2]
  avec un maillage de pas h=0.2

  [X,Y] = *meshgrid*(-2:0.2:2, -2:0.2:2);
  Z = X.*exp(-X.^2-Y.^2);
  meshc(X,Y,Z);

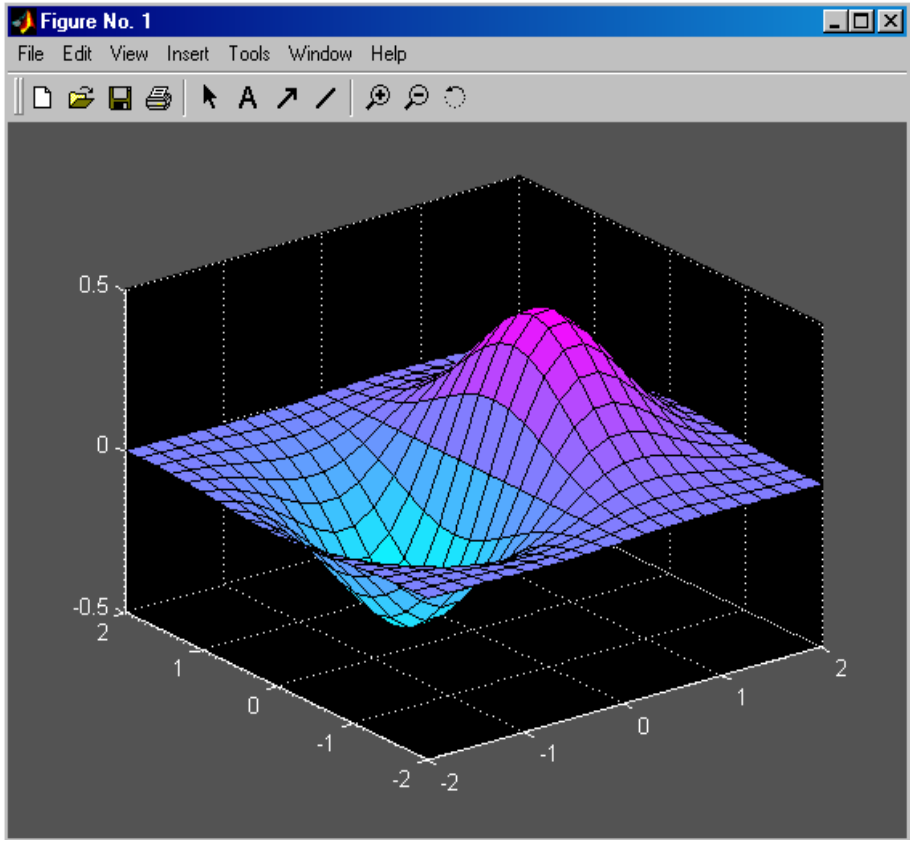


Surface d'équations(3)

- commande surf(*X*,*Y*,*Z*)
  surface de la fonction Z=g(X,Y)
  et colore la surface

- fonction $z=x\exp(-x^2-y^2)$ sur le domaine $[-2,2]\times[-2,2]$
  avec un maillage de pas h=0.2

  ```
  [X,Y] = meshgrid(-2:0.2:2, -2:0.2:2);
  Z = X.*exp(-X.^2-Y.^2);
  surf(X,Y,Z);
  ```



Options d'apparence

voir *help graph3d*

- Palette de couleurs : commande *colormap*(*palette*)

  Ex : *colormap*(*gray*) = palette de dégradés de gris

  colormap(*winter*) = palette de dégradés de bleu et vert

- Couleur unique : *surf*(*X*,*Y*,*Z*,'*FaceColor*','*couleur*')

  Ex : *surf*(*X*,*Y*,*Z*,'*FaceColor*','r') colorie la surface en rouge

- Couleur du maillage de la surface : *surf*(*X*,*Y*,*Z*,'EdgeColor','*couleur*')

  Ex : *surf*(*X*,*Y*,*Z*,'EdgeColor','none') supprime le maillage

Options d'apparence

voir *help graph3d*

- Contrôle des axes :
  - commande *axis [xmin xmax ymin ymax zmin zmax]*
- Point de vision :
  - commande *view*(*visionhor*,*visionvert*)

    Ex : view(30,65) = angle (30°,65°)
    » Orientation 3D par défaut : commande *view(3*)
    équivalent à *view(-37.5,30)*

  - commande *rotate3d* = rotation interactive
- Barre de couleurs : commande *colorbar*
- Eclairage d'une surface :
  - Position de l'éclairage : commande *camlight* <position>
    Ex : *camlight left*
  - Mode d'éclairage : commande *lighting* <mode>
    Ex : *lighting phong*

Visualisation de volumes

- Visualisation graphique de volumes :
  - Tableaux 3D :

    [x y z v] avec v : valeurs prises aux coordonnées x, y ,z
- Quelques fonctions :
  - Image
  - Contourslice
  - Slice
  - Isonormals
  - …

Données MRI

- Données MRI : fichier mri.mat
  - 27 coupes d'un cerveau humain
  - Tableau D: objet 4D 128×128×1×27 données
  - Map : couleurs associées

- Préparer les données :
  - Récupérer les données : *load* mri
  - Transformer le tableau 4D en 3D : D = *squeeze*(D);

    D = tableau 3D 128×128×1×27

    Commande *squeeze* = supprime les dimensions 1 d'un tableau

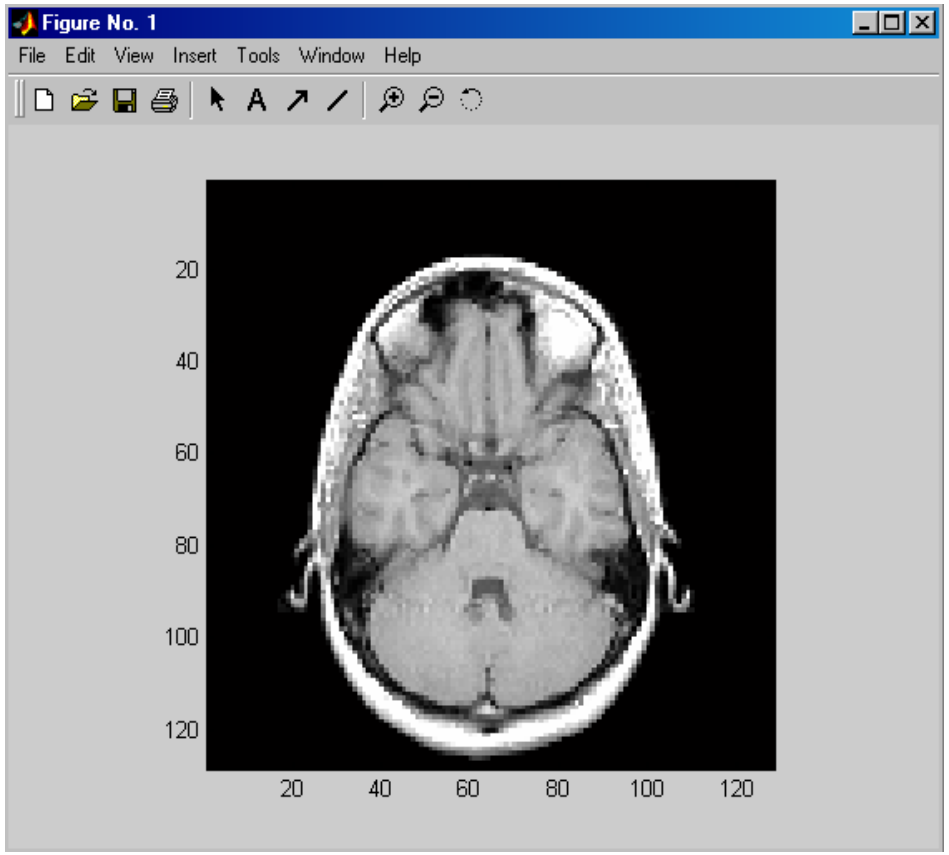
Image MRI

- Commande *image* :
  - *image*(*C*) : visualise une matrice C dont les valeurs spécifient les couleurs de l'image

- Ex : image de la coupe 8

```
image_num = 8;
image(D(:,:,image_num))
axis image
colormap(map)
x=xlim
y=ylim
```

axis image = la boîte s'adapte autour des données



Contour 2D d'une coupe

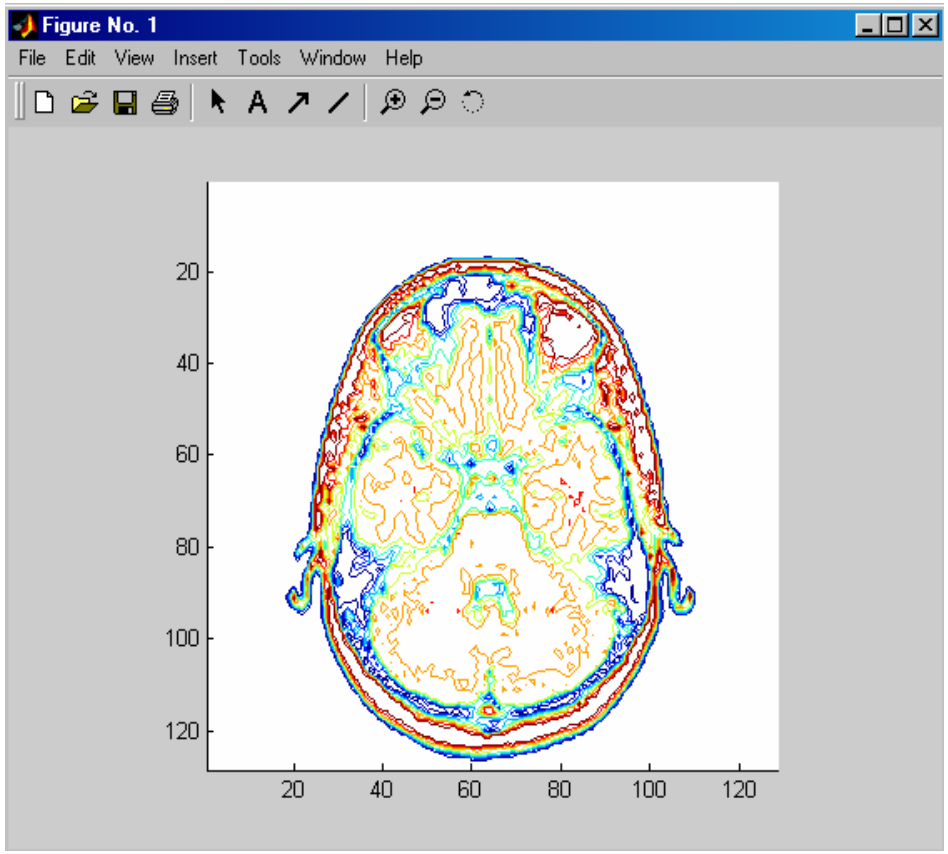
- Commande *coutourslice :*
  - *contourslice(X,Y,Z,V,X_I,Y_I,Z_I,n)*
    lignes de niveaux de la surface
    du volume *V* de dimensions N*M*P
    sur la grille définie par X_I,Y_I,Z_I
    n = nombre de lignes de niveaux
    grille de maillage : *X,Y, Z*
  - *contourslice(V,XI,YI,ZI)*
    par défaut [X Y Z]=meshgrid(1:N,1:M,1:P)

- Ex :

```
contourslice(D,[ ],[ ],image_num)
axis ij
xlim(x)
ylim(y)
daspect([1,1,1])
colormap('default')
```

axis ij = mode « matrice » des axes
l'origine du système de coordonnées se trouve au coin supérieur gauche

daspect = échelle relative des axes
daspect([1,1,1]) = aspect réel
équivalent à axis square
daspect([a,b,c]) = ratio d'aspect des données
a unités de x = b unités de y = c unités de z



Contour 3D de coupes

- Commande *coutourslice* :
  - *contourslice(X,Y,Z,V,X_I,Y_I,Z_I,n)*
    lignes de niveaux de la surface
    du volume *V* de dimension N*M*P
    sur la grille définie par X_I,Y_I,Z_I
    n = nombre de lignes de niveaux
    grille de maillage : *X,Y, Z*
  - *contourslice(V, X_I,Y_I,Z_I)*
    par défaut [X Y Z]=meshgrid(1:N,1:M,1:P)

- Ex :

```
phandles=contourslice(D,[],[],[1,12,10,27],8);
view(3);
axis tight;
set(phandles,'LineWidth',2)
```

view(3) = permet d'avoir une vue 3D
axis tight = fixe les limites des axes à l'étendue des données