

TP MATLAB 1

Visualisation de fonctions, courbes et surfaces

L'objectif de ce TP est de donner les commandes de base de **Matlab** pour tracer des fonctions, des courbes et surfaces en dimension 1, 2 et 3.

1 Quelques rappels Matlab

On peut taper les instructions directement dans l'espace de travail.

Exercice 1 Taper dans l'espace de travail les instructions suivantes (il y a des commentaires précédés de %) :

<pre>>> a=5 >> a+2 >> b=5; >> a+b >> clear all >> a+b >> 1:1:5 >> linspace(0,1,10) >> h=0.001, a=3, b=5 >> x=a:h:b >> clc >> f=@(x) sin(x)./x >> f(1) >> f(pi) >> f(0) >> f([1 2 3]) >> x=linspace(-4*pi,4*pi,1000); >> y=f(x); >> plot(x,y)</pre>	<p>"," permet de ne pas afficher la réponse</p> <p>efface toutes les variables en mémoire</p> <p>crée un vecteur</p> <p>crée un grand vecteur</p> <p>"," permet de taper plusieurs instructions à la suite</p> <p>ne pas oublier le point virgule! une autre manière de créer un grand vecteur</p> <p>efface toutes les instructions tapées mais conserve les variables en mémoire</p> <p>crée une fonction dont la variable est le réel ou vecteur x (attention au .)</p> <p>ne pas oublier le point virgule car c'est un vecteur de taille 1000!</p>
--	--

On peut aussi créer un script qui permet de taper plusieurs instructions à exécuter à la suite, dans un fichier *nom_du_script.m* **Tous les scripts d'une session doivent être placés dans un même répertoire dédié.**

2 Exemples de fonctions à une variable

Lorsque l'on souhaite tracer à l'aide de **Matlab** une fonction f sur un intervalle donné $I = [a, b]$, l'idée consiste à :

1. introduire un échantillonnage $a = x_1 < x_2 < \dots < x_N = b$ de l'intervalle $[a, b]$;
2. déterminer la valeur de la fonction $f(x_i)$ pour $i = 1, 2, \dots, N$;
3. tracer à l'aide de la commande **plot** le nuage de points reliés $\{(x_i, f(x_i))\}_{1 \leq i \leq N}$.

Exercice 2 Ouvrir le script **exercice2.m** qui contient les instructions permettant de tracer les fonctions $f : x \mapsto \frac{\sin(x)}{x}$ et $g : x \mapsto \frac{\cos(x) - 1}{x^2}$ sur l'intervalle $[-4\pi, 4\pi]$:

```
a = -4*pi; b = 4*pi; N = 1000;
x = linspace(a,b,N);
% on peut aussi créer x d'une autre manière :
% h=0.001;
% x=a:h:b;
f = @(x) sin(x)./x;
g = @(x) (cos(x)-1)./(x.^2);
plot(x,f(x),'linewidth',2); hold on; plot(x,g(x),'--m');
axis([-4*pi,4*pi,-1,1.5])
legend('x-> sin(x)/x', 'x -> (cos(x)-1)/x^2')
title('Exemples de fonctions')
```

Modifier le code précédent afin de comprendre l'influence des différents paramètres : résolution, couleurs et styles des graphiques...

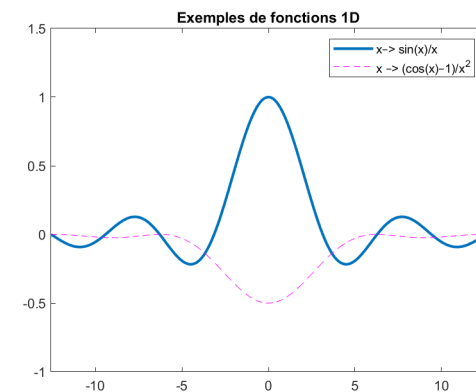


FIGURE 1 – Exemple d'utilisation de la commande **plot** en 1D

Commandes graphiques

- Commande `plot`. Syntaxe : `plot(x,y,'cst','options')`.
- L'option 'cst' spécifie une couleur, un symbole, un style de trait.

c	s	t
y jaune g vert	. point	- trait plein
m magenta b bleu	o cercle	: pointillé court
c cyan w blanc	x marque	-- pointillé long
r rouge k noir	* étoile	

- Un exemple d'option : 'linewidth' = épaisseur de la courbe.
- Autres commandes :
 - `clf` pour effacer une figure;
 - `axis([xmin xmax ymin ymax])` pour choisir la fenêtre d'affichage;
 - `grid` pour afficher une grille;
 - `hold on` pour superposer plusieurs courbes;
 - on peut aussi superposer des courbes avec par exemple `plot(x,cos(x),x,sin(x))`.
- La commande `fplot` : `fplot(@x sin(2*x))` trace la fonction $x \mapsto \sin(2x)$ sur l'intervalle par défaut $[-5, 5]$.
Syntaxe : `fplot(fonction, [xmin xmax])`.
- L'instruction `>> saveas(gcf,'figure1','png')` permet de sauvegarder la figure courante en fichier au format png nommé "figure1".

Exercice 3 (Une mystérieuse fonction T)

Commandes utiles : `fplot`, `exp` pour la fonction exponentielle, `log` pour la fonction \ln .

Soit T la fonction sur \mathbb{R} définie par $T(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}}$.

1. Définir la fonction T dans Matlab.
2. Créer un tableau de nombreuses valeurs dans x puis calculer le tableau $T(-x) + T(x)$. Que peut-on conjecturer sur la fonction T ?
3. Représenter graphiquement la fonction T sur l'intervalle $[-5, 5]$.
4. On définit la fonction g sur $] -1, 1[$ par $g : x \mapsto \frac{1}{2} \ln\left(\frac{1+x}{1-x}\right)$.

Tracer sur un même graphique la courbe représentant la fonction T en rouge, ainsi que celle représentant la fonction g en bleu et la droite d'équation $y = x$ en noir. Que remarque-t-on ? Quelle relation peut-on en déduire entre f et g ?

Les plus perspicaces d'entre vous auront sûrement reconnu la fonction th...

Exercice 4 (Courbes obtenues par translations, symétries)

Commandes utiles : `fplot`, `axis`. On pourra créer un petit script pour chaque question.

1. Représenter sur un même graphique $f : t \mapsto \frac{2 \cos(2t)}{1 + \cos(2t)}$ en noir et $g : t \mapsto \tan^2(t)$ en rouge sur un intervalle bien choisi. Conjecturer une relation entre f et g et la vérifier à l'aide de Matlab.
2. Même travail avec $F : t \mapsto \frac{2}{1 + \text{th}(t)}$ et $G : t \mapsto e^{2t}$.

Exercice 5 (Fonctions puissances/exponentielles)

Commande utile : une boucle `for`.

`for k=[0 1 2 1/2] fplot(@x x.^k); hold on ; end`

1. Tracer sur un même graphique les familles de fonctions puissances $x \mapsto x^k$ pour $x > 0$ et $k \in \{0, 1, 2, 4, \frac{1}{2}, \frac{1}{4}, -1, -2, -\frac{1}{2}\}$.
2. Tracer sur un même graphique les familles de fonctions exponentielles $x \mapsto a^x$ pour x réel et $a \in \{0, 1, 2, 4, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}\}$.

3 Courbes paramétrées

Une courbe \mathcal{C} en $2D$ peut aussi être définie à l'aide d'une paramétrisation $\mathcal{C} = \{(x_1(t), x_2(t)), t \in I\}$.

Avec **Matlab**, pour obtenir un cercle de rayon 1 centré à l'origine, on peut taper

- soit :

```
t=linspace(0,2*pi,100);  
plot(cos(t),sin(t))
```
- soit :

```
fplot(@t cos(t),@t sin(t))    ou    fplot(@cos,@sin)
```
- Le script suivant `courbe_param.m` permet de tracer une courbe paramétrée dynamique. Exemple pour une courbe de Lissajous :

```
for k=1:500  
t=linspace(2*(k-1)*pi/500,2*k*pi/500,2);  
plot(cos(3*t),sin(2*t));hold on  
axis equal;axis([-1.2 1.2 -1.2 1.2])  
pause(0.001)  
end
```

En rajoutant en début de script la commande `mafig = figure`, puis dans la boucle l'instruction complexe

```
im = frame2im(getframe(mafig));
[imind,cm] = rgb2ind(im,256);
if k == 1
    imwrite(imind,cm,'courbe_param.gif','gif','Loopcount',inf);
else
    imwrite(imind,cm,'courbe_param.gif','gif',...
        'WriteMode','append','DelayTime',0.1);
end
```

Matlab crée le fichier animé "courbe_param.gif".

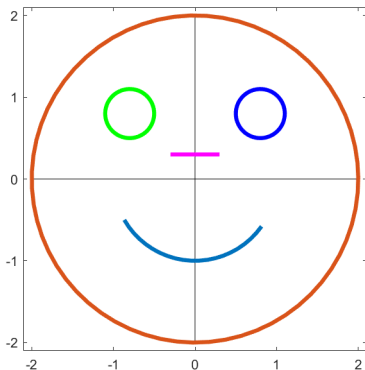
Exercice 6 En utilisant une représentation paramétrique, tracer :

1. le cercle centré en $(2, 1)$ de rayon 3;
2. la droite passant par le point $A(-1, 2)$ de vecteur directeur $\vec{u}(3, 1)$;
3. l'ellipse centrée au point $B(2, 1)$ telle que $a = 3$ et $b = 2$.

Exercice 7 Tracer les courbes paramétrées suivantes, conjecturer graphiquement les symétries vérifiées par les courbes et les vérifier par le calcul.

1. $\begin{cases} x(t) = \cos(t) \\ y(t) = (1 + \cos(t)) \sin(t) \end{cases}, t \in [-\pi, \pi];$
2. $\begin{cases} x(t) = 2 \sin(t) + \cos(t) \\ y(t) = \sin^3(t) + 2 \cos^3(t) \end{cases}, t \in [-\pi, \pi].$

Exercice 8 Fabriquer un smiley comme celui-ci (ou encore plus beau!) à base de courbes paramétrées.



4 Bonus 1 : exemple de fonction 2D

Nous nous intéresserons maintenant à la représentation d'une fonction 2D définie sur un ensemble $I = [a_1, b_1] \times [a_2, b_2]$. Comme précédemment, l'idée est d'afficher une interpolation de f définie sur un échantillonnage de I . Pour ce faire, on va construire deux échantillonnages respectivement de $[a_1, b_1]$ et $[a_2, b_2]$, puis utiliser la commande `meshgrid` qui permet d'obtenir un échantillonnage de I .

Voici un exemple de script **Matlab** qui permet de tracer sur l'ensemble $I = [-2, 2] \times [-2, 2]$ la fonction

$$f : (x_1, x_2) \rightarrow e^{-x_1^2 - x_2^2}.$$

```
colormap('jet')
a1 = -2; b1 = 2; a2 = -2; b2 = 2; N = 100;
x1 = linspace(a1,b1,N); x2 = linspace(a2,b2,N);
[X1,X2] = meshgrid(x1,x2);
f = @(x1,x2) exp(-x1.^2-x2.^2);
surf(x1,x2,f(X1,X2));
shading flat
title('(x_1,x_2) -> exp(-x_1^2-x_2^2)')
```

- Tester le code en variant les différents paramètres. Quelle est l'action de la commande `shading flat`?
- Tracer cette fonction en utilisant les commandes `imagesc`, `contour` ou `contourf` à la place de `surf` et en rajoutant la commande `colorbar`.

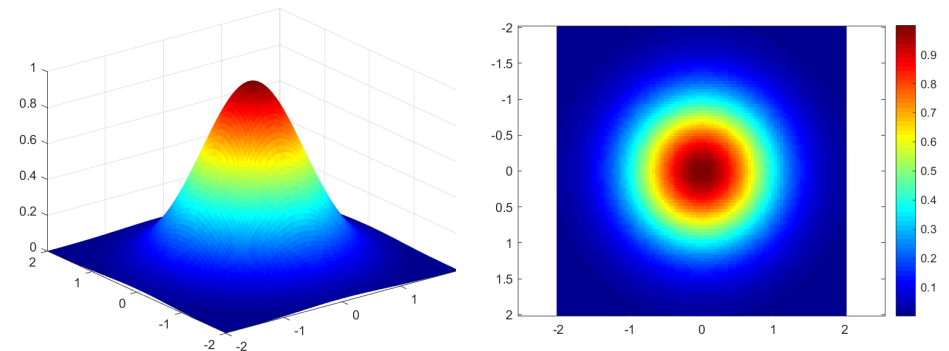


FIGURE 2 – Exemple d'utilisation des commandes `surf` et `imagesc` pour représenter une fonction 2D

5 Bonus 2 : exemple de surface en 3D

Une surface en 3D peut être définie de manière paramétrique ou de manière implicite à l'aide d'une équation cartésienne.

Par exemple, la sphère S de rayon 1 centrée à l'origine peut être définie à l'aide de coordonnées sphériques selon

$$S = \{(\sin(\phi) \cos(\theta), \sin(\phi) \sin(\theta), \cos(\phi)), (\theta, \phi) \in [0, 2\pi] \times [0, \pi]\}$$

ou encore par l'équation cartésienne

$$x_1^2 + x_2^2 + x_3^2 = 1.$$

Pour afficher la surface, on pourra alors utiliser respectivement les commandes `mesh` et `isonormal` pour chacune des deux représentations précédentes.

Voici un premier script **Matlab** qui permet de tracer une sphère en mode paramétrique :

```
clf;
theta = linspace(0,2*pi,50); phi = linspace(0,pi,25);
[Theta,Phi] = meshgrid(theta,phi);
x1 = @(theta,phi) sin(phi).*cos(theta);
x2 = @(theta,phi) sin(phi).*sin(theta);
x3 = @(theta,phi) cos(phi);
mesh(x1(Theta,Phi),x2(Theta,Phi),x3(Theta,Phi))
```

et un deuxième script pour la version implicite :

```
clf;
N = 100; x = linspace(-1,1,N);
[X,Y,Z] = meshgrid(x,x,x);
f = X.^2 + Y.^2 + Z.^2 - 1;
p = patch(isosurface(x,x,x,f,0));
isonormals(x,x,x,f,p)
set(p,'FaceColor','red','EdgeColor','none')
axis([-1.2 1.2 -1.2 1.2 -1.2 1.2])
camlight('infinite')
lighting gouraud
alpha(0.9)
```

- Exécuter le premier script en modifiant les paramètres du code.
- Exécuter le deuxième script en modifiant aussi les paramètres du code. À quoi servent les trois dernières lignes ?

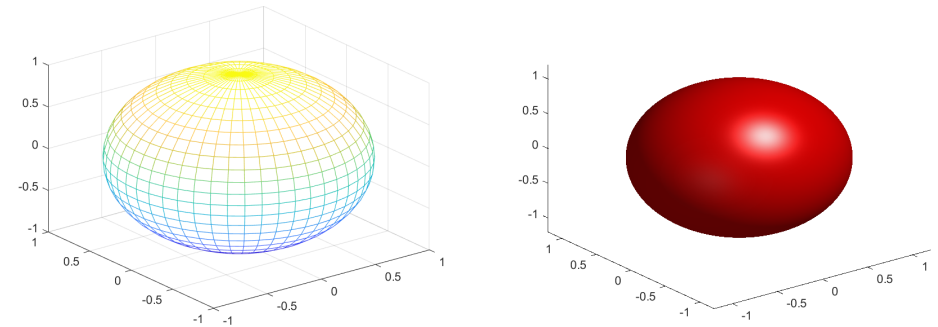


FIGURE 3 – Exemple d'utilisation des commandes `mesh` et `isonormal` pour représenter une surface en 3D