**Nom :** **Date :**
**Prénom :**

---

Faculté des sciences et ingénierie (Toulouse III) Année universitaire
Département de mathématiques – L3 MMESI 2012-2013
Analyse numérique I

# TP n° 4 – Tracé de courbes et approximation de fonctions

---

Aller dans le dossier `L3_analyseNumerique` et créer le dossier `TP04`, dans lequel seront placés les fichiers relatifs à ce TP.

## 1 Découverte de Matlab (4) : outils graphiques

### 1.1 Tracé de courbes en Matlab

Maintenant que nous savons créer et calculer les valeurs de fonctions, reste à savoir tracer leurs courbes représentatives. C'est ce que l'on se propose de découvrir au cours de ce TP. Et comme la vie est bien faite, cela correspond aussi au cours que nous commençons : comment, étant donné une famille d'abscisses $x_0, \dots, x_N$ et d'images $f(x_0), \dots, f(x_N)$, peut-on approcher $f$ par une fonction régulière qui passe par les points $(x_i, f(x_i))$ ?

Vous savez maintenant que Matlab ne fait pas de calcul formel, c'est-à-dire que, pour lui, seules les données numériques permettent de représenter des objets mathématiques. Ainsi, pour Matlab, le tracé de la courbe représentative d'une fonction $f$ ne se fait pas à partir de la définition de la fonction $x \mapsto f(x)$ en tant qu'application de la variable $x$. Souvenez-vous de votre vieille calculette du lycée... vous lui donniez un intervalle pour $x$, elle calculait plein de valeurs de $f(x)$ et plaçait sur un graphe les couples $(x, f(x))$ calculés ; comme elle en calculait un grand nombre, en "reliant les points", cela faisait une jolie courbe...

Pour Matlab c'est presque pareil sauf que, cette fois, *vous* lui donnez un vecteur $x = (x_1, x_2, \dots, x_N)$ d'abscisses [1], puis :
- il calcule les images $f(x_1), f(x_2), \dots, f(x_N)$, que l'on convient de noter $f(x)$ ;
- il place les points $(x, f(x))$ dans un repère ;
- il relie ces points par des segments.

Ainsi, si vous prenez peu de points [2], votre "graphe de $f$" ne ressemblera à rien !

Pour ces tracés, on va utiliser la commande **plot** dont on va tester la syntaxe et les options.

---

1. C'est-à-dire que chacun des $x_i$ est une abscisse.
2. C'est-à-dire un $N$ très petit.

*Exemple* 1. Reproduire (à main levée !) et expliquer le résultat observé sur ces quatre tracés :

```
x=[0,pi/2,pi]
y=sin(x)
plot(x,y)
```

```
N=100
x=linspace(0,pi,N)
y=sin(x)
plot(x,y)
```

```
x=[0,pi,2*pi]
plot(x,sin(x))
```

```
x=linspace(0,2*pi,N);
plot(x,sin(x))
```

Que se passe-t-il si on refait les mêmes graphes avec $N = 10$ ?

**Attention !** Dans la syntaxe **plot**(x,**sin**(x)), le premier argument (ici x) ne correspond pas au nom d'une variable par rapport à laquelle on va tracer le graphe de la fonction sin. Il correspond à une liste d'abscisses auxquelles on va placer les points du graphe.

Autrement dit, ce n'est pas "Matlab trace la fonction sinus sur $[0,\pi]$" mais "étant donné un vecteur $x = (x_1, \dots, x_N)$, Matlab place les points $(x_i, \sin(x_i))$ puis les relie par des segments".

## 1.2 Options graphiques

*Exemple* 2. Reproduire les commandes suivantes en ligne de commande et expliquer le résultat obtenu.

```
x=linspace(0,2*pi,100)
y=sin(x)
z=cos(x)
plot(x,y)
plot(x,z)
plot(x,y,x,z)
clf
hold on
plot(x,y)
plot(x,z)
clf
hold on
plot(x,z)
plot(x,y,'r')
clf
plot(x,y,'b--',x,z,'*')
```

Quel est l'intérêt de la commande `hold on` ?

Comment annuler son effet ?

Comment faire apparaitre des axes orthonormés ? une grille ?

Comment faire apparaitre une légende sur les axes ? un titre ?

*Commandes à utiliser :*

```
axis on    axis off    axis equal    grid on    grid off
legend    xlabel    ylabel    title
```

```
close all    figure
```

## 1.3 Un petit exercice

Pour gérer la liste des options et éviter de tout recopier à chaque fois en ligne de commande, on va se servir de scripts.

**Exercice 1.** Soit la fonction $f : x \mapsto x\sin(\frac{1}{x})$.

1. Supposons le vecteur des abscisses de taille 3 (*i.e.* `x=[x1,x2,x3]`) : exprimer alors (en fonction de `x1`, `x2`, `x3`) le vecteur des ordonnées permettant de tracer $f$.
2. Créer un script `exercice1.m` dans le répertoire `TP04` qui permet de tracer dans deux figures séparées :
   – la fonction $f$ pour $x \in [-1, 1]$ ;
   – la fonction $f$ pour $x \in [-0.05, 0.05]$ ainsi que les deux droites $y = x$ et $y = -x$.
   Faire varier le nombre de points et commenter le résultat observé.

**Exercice 2** (Courbes paramétrées)**.** Un des intérêts principaux de la manière dont Matlab trace les courbes (en plaçant des points (abscisse,ordonnée)), est la simplicité avec laquelle on peut tracer des courbes paramétrées. En effet, pour tracer une courbe paramétrée, on calcule (grâce à différentes valeurs du paramètre) les abscisses des points de la courbe séparément des ordonnées. Ainsi, avec Matlab, il suffit : de fabriquer le vecteur des abscisses à partir du paramètre, de fabriquer le vecteur des ordonnées à partir du même paramètre et enfin de placer les points sur un graphe.

On demande ici de créer un script (`exercice2.m`) dans lequel tracer les courbes paramétrées suivantes :

1. $x(\theta) = 2\cos(\theta)$, $y(\theta) = \sin(\theta)$ lorsque $\theta$ varie dans $[0, \pi/2]$, puis $[0, \pi]$, $[0, 3\pi/2]$ et enfin $[0, 2\pi]$. Qu'observez-vous ?

2. $x(t) = \exp(-0.1t)\cos(t), \quad y(t) = \exp(-0.1t)\sin(t)$ lorsque $t \in [0, 20]$ puis $t \in [0, 200]$. Qu'observez-vous ?

## 2 Interpolation de Lagrange d'une fonction

On considère un échantillon de $n+1$ points (des "mesures") $(x_0, y_0), \ldots, (x_n, y_n)$. On cherche à construire une fonction "régulière" qui passe par tous les points. La première idée (un peu comme on prendrait une droite si l'on n'avait que deux points), est de chercher s'il existe un tel polynôme.

**Idée** Chercher $P \in \mathbb{R}[X]$ tel que $\forall 0 \leq i \leq n,\ P(x_i) = y_i$.

### 2.1 Base de Lagrange associée à un vecteur $X = (x_0, \ldots, x_n)$

**Théorème :** Étant donné $n+1$ réels distincts $x_0, \ldots, x_n$ et $n+1$ réels $y_0, \ldots, y_n$, il existe un unique polynôme $P$ tel que :

$$P \in \mathbb{R}_n[X] \quad \text{et} \quad \forall 0 \leq i \leq n,\ P(x_i) = y_i. \tag{1}$$

**Base de Lagrange associée à $X = (x_0, \ldots, x_n)$ :** Soit $0 \leq i \leq n$, considérons le polynôme $\ell_i$ défini par :

$$\forall t \in \mathbb{R}, \quad \ell_i(t) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{t - x_j}{x_i - x_j}.$$

Alors, pour tout $0 \leq i \leq n$ :

1. $\ell_i \in \mathbb{R}_n[X]$.
2. $\ell_i(x_j) = 0$ si $i \neq j$, $\quad \ell_i(x_j) = 1$ si $i = j$.
3. $(\ell_0, \ldots, \ell_n)$ constitue une base de $\mathbb{R}_n[X]$.

Par conséquent, il est clair que le polynôme $P(t) = \sum_{i=0}^{n} y_i \ell_i(t)$ satisfait les deux conditions de (1). Par unicité, c'est donc le polynôme d'interpolation de Lagrange associé aux points $(x_0, y_0), \ldots, (x_n, y_n)$.

**Exercice 3.** Écrire une fonction lagrange(X,i,t) qui, étant donné un vecteur $X$, un entier $i$ et un réel $t$ renvoie $\ell_i(t)$, où $\ell_i$ est le $i$-eme polynôme de la base de Lagrange associée à $X = (x_0, \dots, x_n)$.
*Important :* Pour pouvoir tracer le graphe de $\ell_i$ par la suite, il est important que si $t$ est un vecteur $t = (t_0, \dots, t_p)$ pour $p \geq 1$, alors lagrange(X,i,t) renvoie le vecteur des $\ell_i(t_k)$ pour $0 \leq k \leq p$.

## 2.2 Interpolation de Lagrange d'une fonction $f$

**Principe** On suppose cette fois que l'on connait uniquement quelques valeurs d'une fonction $f$ régulière et l'on veut la reconstituer (ou du moins l'approcher). On suppose donc que :

- l'on connait $n+1$ points du graphe de la fonction $f$ : $(x_0, f(x_0)), \dots, (x_n, f(x_n))$ ;
- l'on ne connait pas $f$,

et on cherche le polynôme de Lagrange associés aux points $(x_0, f(x_0)), \dots, (x_n, f(x_n))$.

**Exercice 4.** Écrire une fonction qui, étant donné deux vecteurs $X = (x_0, \dots, x_n)$ et $Y = (y_0, \dots, y_n)$, construit le polynôme de Lagrange associé ces points. On pourra pour cela utiliser le script qui permettait de définir la base de Lagrange

**Exercice 5.** Écrire une fonction qui, étant donné un vecteur $X = (x_0, \dots, x_n)$ et une fonction $f$ prédéfinie, construit la fonction polynomiale de Lagrange de $f$ associée aux points $x_0, \dots, x_n$.

**Exercice 6.** Calculer le polynôme d'interpolation de Lagrange $\Pi_n$ de la fonction sin définie sur l'intervalle $[0, 2\pi]$ associé à $n+1$ points équidistants de $[0, 2\pi]$, avec $x_0 = 0$ et $x_n = 2\pi$ pour $n = 3, 4, 5, 6$. Le tracer en superposant les résultats pour les différentes valeurs de $n$ et tracer sur le même graphique la fonction sin. Faire de même pour la fonction $x \mapsto \frac{1}{1+x^2}$ sur l'intervalle $[-5, 5]$ (qu'observez-vous sur les bords lorsque $n$ augmente ?).

## 2.3 La commande polyfit

Comme la vie est bien faite, il se trouve que la commande polyfit, préprogrammée en Matlab, permet de calculer les coefficients du polynôme d'interpolation de Lagrange associé à des points $(x_0, y_0), \dots, (x_n, y_n)$ où $X = (x_0, \dots, x_n)$ et $Y = (y_0, \dots, y_n)$ sont donnés.

**Exercice 7.** Expliquer la syntaxe de la commande polyfit et ce qu'elle renvoie.

Écrire un script qui permet, grâce aux commandes polyfit et polyval, étant donné $X = (x_0, \dots, x_n)$ et $Y = (y_0, \dots, y_n)$ de tracer le polynôme d'interpolation de Lagrange associé aux points $(x_0, y_0), \ldots, (x_n, y_n)$. Faire apparaitre les points d'interpolation sur le même graphe.

Écrire un script qui permet de tracer le polynôme d'interpolation de Lagrange de la fonction sin associé à 7 points équidistants. Tracer sur le même graphe le polynôme obtenu à l'exercice précédent ainsi que la fonction sin.

## 2.4 Interpolation et population américaine

**Exercice 8.** Notre but est d'examiner si on peut prédire à partir d'une interpolation polynomiale la valeur de la population américaine en 2010 alors qu'on ne dispose que de ses valeurs sur les changement de dizaines entre 1900 et 2000.

1. Le fichier `census.dat` de Matlab contient l'évolution de la population américaine de 1900 à 2000. Pour le charger, tapez `load census` en ligne de commande. Le vecteur $p$ contient les valeurs de la population américaine de 1900 à 2000 tous les 10 ans.

2. Retenir les valeurs correspondant aux changements de dizaines entre 1900 et 1980. On notera ce vecteur pop9. Retenir parmi les 9 valeurs déjà prises les valeurs associées aux années 1900, 1920, 1940, 1960 et 1980. Ce vecteur sera noté pop6. Enfin, retenir parmi ces 6 dernières valeurs uniquement les valeurs correspondant aux années 1900, 1940 et 1980. Ce vecteur sera noté pop3.

3. Écrire un script `population.m` qui permet de calculer le polynôme de Lagrange passant par $n+1$ points $(x_0, y_0), \ldots, (x_n, y_n)$ à l'aide des commandes **polyfit** et **polyval**.

4. Tracer maintenant les polynômes d'interpolation correspondant à pop3, pop6 et pop9 en faisant apparaitre les données sur la population.

   Que constatez-vous ? Que dire de la population américaine en 2010 ?