**TP2 : Interpolation**, **Polynômes de Lagrange …**

1. **Opérations sur les polynômes dans MATLAB**
2. **Ecriture des fonctions personnelles**

Dans Matlab, les programmes qui sont sauvegardés comme des fichiers.m sont équivalents à des sous programmes et des fonctions dans d’autres langages.

Exemple1 :

1. Sauvegarder le programme suivant sous le nom de : demof.m

function y= demof(x)

y=2\*x+3

1. Pour déterminer f(x=3) par exemple, il suffit d’exécuter la commande suivante :

>> y = demof(3)

La réponse sera :

>> y = 9

Exemple2 : sauvegarder sous le nom de : demof1.m

function [z y] = demof1(x1,x2)

y=2\*x1-3

z= 2\*x2+3

>> y = 3

 z= 13

NB : y,z sont des variables de sortie, x1, x2 sont les variables d’entrée.

**2)** **Ecriture d’un programme dans Matlab**

En Matlab, les programmes se terminent par une extension ‘.m’ dans le nom du fichier programme.

Exemple1 : Calcul de la valeur moyenne

Ecrire le programme suivant :

x=[1 2 3 4]

n=length(x) % la longueur de x

somme =sum(x)

mean = somme /n

% sauvegarder le programme sous le nom de : demof.m et exécuter.

Exemple 2 : calcul de n !

input(‘donner la valeur de n,n=’)

nfact=1

i=1 :1 :n % boucle sur i avec un pas de 1

nfact=nfact\*i ;

nfact % le résultat final de n factoriel

Exemple3 : Calcul du produit de :

$$y= \prod\_{i=1}^{n}\sin((i\*x))$$

input (‘x en degrés, x=’)

input (‘n, n=’)

x=x\*pi/180 ; % changement de x en radians

y=1,

i=1 :1 :n

y= y\*sin(i\*x)

y % affiche le résultat final

1. **Opérations sur les polynômes**

Dans matlab, les polynômes son représentés sous forme de vecteurs lignes dont les composants sont données par ordre des puissances décroissantes. Un polynôme de degré n est représenté par un vecteur de taille (n+1).

Exemple : le polynôme : f(x)= 8x5+2x3-3x2-4x-2 est représenté par :

>> f=[8 0 2 -3 4 -2]

f=8 0 2 -3 4 -2

1. **Multiplication des polynômes**

On utilise la fonction ‘conv’

Exemple :

f(x) = 3x3+2x2-x+4

g(x)= 2x4-3x2+5x-1

Le produit de convolution : h(x) = f(x).g(x) est donné par :

>> f = [3 2 -1 4]

>> g = [2 0 -3 5 -1]

>> h = conv(f,g)

h=

6 4 -11 17 10 -19 21 -4

C’est à dire : h(x) =6x7+4x6-11x5+17x4+10x3-19x2+21x-4

1. **Division des polynômes**

On utilise la fonction ‘deconv’

Exemple :

f(x)= 3x3+2x2-x+4

g(x)=2x4-3x2+5x-1

h(x)=g(x)/f(x)

>> h=deconv(g,f)

>>h=

.6667 -.444

C’est à dire : h(x)= 0.6667x – 0.4444

1. **Manipulation de fonctions polynomiales dans Matlab**
2. **Racines d’un polynôme**

P(x) = 2x3 +x2+4x+5

>> P = [2 1 4 5]

>> r = roots(P)

>> r

0.25 + 1.5612i

0.25-01.5612i

-1.00

1. **Evaluation d’un polynôme**

Exemple :

P(1)=

>> polyval(P,1)

Ans

12

Exemple : y = 3x4-7x3 +2x2 +x+1

>> C = [3 -7 2 1 1]

>> x = 2.5

>> y = polyval (C,x)

>> y=

23.8125

1. **Interpolation linéaire et non linéaire**

Une interpolation consiste à relier les points expérimentaux par une courbe sous forme de segments de droites ou de courbes polynomiales. Ceci peut être réalisé par la fonction ‘interp1’. La commande ‘interp1(x,y,xi,’type’) retourne un vecteur de même dimensions que xi et dont les valeurs correspondent aux images des éléments de xi déterminées par interpolation sur x et y. Si f est l’interpolation de y, la chaine ‘type’ spécifie alors le type d’interpolation qui doit être parmi les suivants :

‘linear’ → interpolation linéaire

‘spline’ → interpolation par splines cubiques

‘cubic’ → interpolation cubiques

Si on ne spécifie pas le type, l’interpolation linéaire est choisie par défaut.

Exemple1 :

>> x=0 :10 ;

>> y = sin(x) ;

>> yi=0 :.25 :10 ;

>> yi = interp1(x,y,xi,’cubic’) ;

>> plot(x,y,’r’,xi,yi)

Dans le cas suivant, on étuduiera les différents types d’interpolation sur un exemple de valeurs discrètes de la fonction cosinus. On appellera l’algorithme : ‘interpol1.m’

% Utilisation de la fonction interpol1

clear all

clc

x=0 :10 ;

y=cos(x) ;

z= 0 :.25 :10 ;

f= interp1(x,y,z) ;

plot (x,y,’\*r’,z,f) ;

grid on

% interpolation par splines cubiques

hold on

f= interp1(x,y,z,’spline’) ;

plot(z,f,’-m’) ;