Université de Rennes 1
UFR Mathématiques

Master 1 CSA
Méthodes numériques

TP 3 - Interpolation polynomiale

Créer un répertoire `TP3` (dans le répertoire où vous regroupez tous les fichiers concernant ce cours) et vous y placer pour lancer Matlab.

Polynômes en Matlab

Un polynôme p de degré n est défini pour `Matlab` par le vecteur `p` contenant les coefficients du polynôme (dans l'ordre des puissances décroissantes) :

$$p(x) = p_1 x^n + p_2 x^{n-1} + \cdots + p_n x + p_{n+1}.$$

`>> y = polyval(p,x)` : évalue le polynôme défini par le vecteur `p` aux points `x`.

`>> p = polyfit(x,y,n)` : donne les $n+1$ coefficients du polynôme $p(x)$ de degré n qui approche au sens des moindres carrés les valeurs `y(i)` en `x(i)`. Si `x` et `y` sont de longueur $n+1$, `p` est le polynôme d'interpolation de Lagrange.

`>> r=roots(p)` : donne les racines du polynôme défini par le vecteur `p`.

`>> t=conv(p,q)` : multiplie les deux polynômes définis par les vecteurs `p` et `q`.

Exercice 1.

Phénomène de Runge pour l'interpolation de Lagrange sur des points équidistants

1) Écrire une fonction `[y]=f(x)` pour $f(x) = \frac{1}{1+x^2}$ en enregistrant le texte suivant dans le fichier de nom `f.m` :

```
function [y]=f(x)
% f(x) = 1/(1+x^2)
y=1 ./ (1+x.*x) ;
```

Noter le point avant les opérateurs `/` et `*` pour que la fonction puisse faire les calculs terme à terme dans le cas d'un `x` matriciel. Tester la fonction sur des valeurs bien choisies.

2) Pour `n=2, 4, 10` et `12`, calculer le polynôme d'interpolation de Lagrange de f sur l'intervalle $[-5, 5]$ aux points équidistants $x_i = -5 + i\frac{10}{n}$, représenter f et ses polynômes d'interpolation sur une même figure (Remarquer le phénomène de Runge aux bords du domaine), calculer l'erreur en norme infinie. Pour cela, écrire un script `tp3_1.m` en s'inspirant des instructions suivantes.

```
tn=[2,4,10,12] ; nbn=length(tn) ;
c=['y' 'm' 'c' 'r' 'g' 'b' 'w' 'k'] ; % Couleurs des courbes
xg=linspace(-5,5,10000) ; yf=f(xg) ;
plot(xg,yf) ; hold on ;
disp(' n    ||f-p||') ;
```

```
for j=1 :nbn
   n=tn(j) ; x=-5+10*[0 :n]/n ; y=f(x) ;
   p=polyfit(x,y,n) ; yp=polyval(p,xg) ;
   plot(xg,yp,c(j)) ;
   disp(sprintf('%3d %.7e',n,norm(yf-yp,inf))) ;
end
hold off
```

Exercice 2.

Phénomène de Runge pour l'interpolation de Lagrange sur les points de Chebishev

1) Reprendre l'étude précédente en remplaçant les points équidistants par les points de Chebishev : c'est-à-dire en remplaçant l'instruction `x=-5+10*[0 :n]/n ;` par `x=5*cos((2*[0 :n]+1)*pi/(2*n+2)) ;`. Puis, sans représenter le polynôme, calculer l'erreur pour `n= 20, 40, 60` et `80`. Regrouper les instructions dans un script `tp3_2.m`

Exercice 3. *Interpolation polynomiale de Lagrange par morceaux*

1) Enregistrer dans le fichier `lagrangem.m` la fonction suivante qui calcule le polynôme d'interpolation par morceaux de degré k de la fonction f sur des sous-intervales $[x_i, x_{i+1}]$ définis par le vecteur `x`.

```
function [p,n]=lagrangem(f,x,k)
% Interpolation de Lagrange par morceaux
% de degre k avec des points equidistants
% sur chaque sous-intervalle [x_{i}, x_{i+1}]
% pour la fonction f.
% p est une matrice dont la ligne i donne les k+1 coefficients
% du polynome d'interpolation sur [x_{i}, x_{i+1}]
n=length(x)-1 ;
for i=1 :n
   xint=linspace(x(i),x(i+1),k+1) ;
   p(i, :)=polyfit(xint,f(xint),k) ;
end
```

2) Enregistrer dans le fichier `pmval.m` la fonction suivante qui évalue le polynôme d'interpolation par morceaux `p` avec `nbpi` points sur chaque sous-intervale $[x_i, x_{i+1}]$ définis par le vecteur `x`.

```
function [y,xe]=pmval(p,x,nbpi)
% Evaluation d'une fonction
% polynomiale par morceaux sur les [x(i),x(i+1)].
% On suppose que x0 < x1 < ... < xn
% La ligne i de p contient les coefficients du polynome
% correspondant a l'intervalle [x(i),x(i+1)].
% Sur [x(i),x(i+1)] on evalue p en nbpi+2 points equidistants.
% y : vecteur des valeurs du polynome aux pts d'evaluation xe.
n=length(x)-1 ;xe=[] ;y=[] ;
```

```
for i=1 :n
   ptsi=linspace(x(i),x(i+1),nbpi+2) ;ptsi=ptsi(1 :nbpi+1) ;
   xe=[xe ptsi] ;y=[y polyval(p(i, :),ptsi)] ;
end
xe=[xe, x(n+1)] ; y=[y polyval(p(n, :),x(n+1))] ;
```

3) Pour `n=2, 4, 8, 16, 32, 64` et `128`, calculer le polynôme d'interpolation de Lagrange par morceaux de degré $k = 1$ sur la subdivision régulière définie à l'exercice 1, représenter f et ses polynômes d'interpolation sur une même figure et calculer l'erreur en norme infinie. Pour cela, écrire un script `tp3_3.m` en s'inspirant des exercices précédents et des instructions suivantes.

```
nbpi=100 ;
x=-5+10*[0 :n]/n ; k=1 ;
p=lagrangem(@f,x,k) ; [yp,xe]=pmval(p,x,nbpi) ;
plot(xe,yp,c(j)) ;
yf=f(xe) ;err(k,j)=norm(yf-yp,inf) ;
```

4) Refaire l'étude de la question précédente pour $k = 2$. On doit obtenir une erreur d'environ `1.77e-5` pour $n = 128$.

5) Estimer numériquement l'ordre de convergence de la méthode pour $k = 1$ et $k = 2$.

6) Estimer numériquement l'ordre de grandeur de la meilleure précision possible avec ces 2 méthodes (il suffit de chercher à partir de quel ordre de grandeur pour n voit-on apparaître le cumul des erreurs d'arrondis sur l'erreur en norme infinie ?).