Interpolation, Polynômes de Lagrange et Splines

C. Dossal

Décembre 2011

1 Polynômes de Lagrange

1. **Convergence uniforme.**
   Montrer que pour tout intervalle $[a,b] \subset \mathbb{R}$, et quelque soit la suite $(x_n)_{n\in\mathbb{N}}$ les interpolations de Lagrange de la fonction $f : x \mapsto \sin(x)$ convergent uniformément vers f sur $[a,b]$.
2. **Comparaison entre interpolation de Lagrange et interpolation composite linéaire**
   On considère la fonction $f(x) = e^{2x}$, avec $x \in [0,1]$. Soit N un nombre entier, on pose $H = \frac{1}{N}$, et on note $\Pi_1^H f$ le polynôme composite linéaire par morceaux qui interpole f aux noeuds $x_i = iH$, pour $i = 1,...N$.
   (a) Déterminer un entier N de sous-intervalles tel que l'erreur d'interpolation $E_1^H(f) = \max_{x\in[0,1]} |f(x) - \Pi_1^H f(x)|$ soit inférieure à 10^{-4}.
   (b) Soit $\Pi_n f$ le polynôme d'interpolation de Lagrange de degré n qui interpole f aux noeuds $x_i = iH$, pour $i = 1,...n$.
   Est-ce que l'erreur d'interpolation $E_n(f) = \max_{x\in[0,1]} |f(x) - \Pi_n^H f(x)|$ tend vers zéro quand $n \to +\infty$?
   (c) Est-ce que le nombre de noeuds nécessaires pour que $E_n(f)$ soit inférieure à 10^{-4} est du même ordre de grandeur que celui de la première question ?
3. Expliquer ce que renvoie la fonction matlab décrite ci dessous et la recopier dans votre répertoire de travail.

```
function P=Lagrange2(a,b,x,y,N)
t=a+(b-a)*[0:1/(N-1):1];
P=0*t;
n=length(y);
for k=1:n
    Lk=ones(1,N);
    for l=1:n
        if abs(l-k)>0.5
           Lk=Lk.*(t-x(l))/(x(k)-x(l));
        end
    end
    P=P+y(k)*Lk;
end
plot(t,P);hold on;
plot(x,y,'or');
```

4. Expliquer ce que renvoie la fonction matlab décrite ci dessous et la recopier dans votre répertoire de travail.

```
function Lagrange3(f,a,b,N,x)
t=a+(b-a)*[0:1/(N-1):1];
I=round((x-a)/(b-a)*(N-1)+1);
y=f(I);
P=Lagrange2(a,b,x,y,N);
hold on;
plot(t,f,'k');
hold off;
```

5. **Exemples d'interpolation.**

   (a) Tester la fonction suivante pour réaliser des interpolations de la fonction sinus sur l'intervalle $[-2\pi, 2\pi]$ en utilisant plusieurs manières de répartir les noeuds afin d'illustrer le premier exercice.
   (b) Réaliser l'interpolation décrite dans l'exercice 2 avec des points équirépartis et évaluer numériquement $\|f - p_n\|_\infty$ pour $n = 5$ et $n = 10$.

(c) Le script suivant permet de comparer l'interpolation de Lagrange en utilisant des points équidistants et les racines des polynômes de Tchebichev sur la fonction $f : x \mapsto \frac{1}{1+x^2}$ sur l'intervalle [-5,5] :

```
n=13;
a=-5;
b=5;
N=1000;
t=a+(b-a)*[0:1/(N-1):1];
f=1./(1+t.^2);
x=a+(b-a)*[0:1/(n-1):1];
close all;
Lagrange3(f,a,b,N,x);
for k=1:n
   z(k)=(a+b)/2+(b-a)*cos((2*k-1)*pi/(2*n))/2;
end
figure;
Lagrange3(f,a,b,N,z);
```

i. Quelle approximation est la meilleure ?

ii. Augmenter le nombre de points et comparer les interpolations.

iii. On observe que l'interpolation avec des noeuds équidistants ne semble pas converger ponctuellement, alors que celle obtenue en utilisant les racines des polynômes de Tchebichev converge uniformément. De fait, il existe un intervalle en dehors duquel la première interpolation ne converge nulle part. C'est cve qu'on appelle le phénomène de Runge. De plus, quelque soit la fonntion continue, l'interpolation de Tchebichev converge uniformément vers la fonction f.

6. **Formule de Newton**
Soient $(x_i, y_i)_{i=0,\dots n}$ un ensemble de points donnés. On définit les différences finies divisées par les notations suivantes :

$$
\begin{aligned}
y[x_i] &= y_i \\
\partial_y[x_i, x_j] &= \frac{y[x_j] - y[x_i]}{x_j - x_i} \\
\partial_y^2[x_i, x_j, x_k] &= \frac{\partial_y[x_j, x_k] - \partial_y[x_i, x_j]}{x_k - x_i} \\
&\vdots \\
\partial_y^n[x_{i_0}, x_{i_1}, \dots, x_{i_n}] &= \frac{\partial_y^{n-1}[x_{i_1}, \dots, x_{i_n}] - \partial_y^{n-1}[x_{i_0}, \dots, x_{i_{n-1}}]}{x_{i_n} - x_{i_0}}
\end{aligned}
$$

Montrer par récurrence que le polynôme d'interpolation de degré n qui passe par les $n+1$ points $(x_0, y_0), \dots, (x_n, y_n)$, avec les x_i distincts, est unique et donné par :

$$p(x) = y[x_0] + (x - x_0)\partial_y[x_0, x_1] + (x - x_0)(x - x_1)\partial_y^2[x_0, x_1, x_2] + \dots + (x - x_0)\dots(x - x_{n-1})\partial_y^n[x_0, x_1, \dots, x_n]$$

Quel est l'avantage de cette formule par rapport à la définition habituelle du polynôme d'interpolation de Lagrange ?

7. **Convergence uniforme.**
Soit $n \in \mathbb{N}$. Soit $(P_k)_{k \in \mathbb{N}}$ une suite de polynômes de degré n convergeant simplement vers f sur l'intervalle $[a, b]$. En utilisant l'interpolation de Lagrange sur $(n+1)$ points distincts de $[a, b]$, montrer que f est un polynôme de degré $\leq n$ et que la convergence est uniforme.

2 Splines

8. **Propriétés des splines.**
Soient des points (x_i, y_i), avec les x_i ordonnés : $x_0 < x_1 < \dots < x_n$. Soit s une fonction $[a, b] \to \mathbb{R}$, avec $a = x_0, b = x_n$, polynômiale de degré 3 sur chaque intervalle $[x_i, x_{i+1}]$, vérifiant
– $s(x_i) = y_i$,pour $i = 0, \dots, n$
– s est de classe C^2

(a) A quel type d'interpolation correspond cette fonction s ?

(b) Montrer que $\forall f : [a, b] \to \mathbb{R}$ vérifiant les deux propriétés précédentes, et telle que

$$s''(b)(f'(b) - s'(b)) = s''(a)(f'(a) - s'(a))$$

on a :

$$\int_a^b s''(x)^2 dx \leq \int_a^b f''(x)^2 dx$$

9. **Construction des Splines.**
Soit $[a,b]$ un intervalle de $\mathbb{R}$, $(x_k)_{1\leqslant k\leqslant n}$, n points ordonnés de $]a,b[$. Soit $(y_k)_{1\leqslant k\leqslant n}$ n réels, on se propose construire la fonction spline de classe C^2 sur $]a,b[$, polynomiale par morceaux sur chaque intervalle $[x_i,x_{i+1}]$ telle que pour tout $k\leqslant n$, $s(x_k)=y_k$ et qui soit affine sur les intervalles $[a,x_1]$ et $[x_n,b]$. Dans la suite on note $\delta x_i = x_{i+1}-x_i$ et $\delta y_i = y_{i+1}-y_i$. On va chercher à montrer dans un premier temps que la fonction s est entièrement définie par les valeurs de s a dérivée seconde aux noeuds x_i. On note $z_i = s^{(2)}(x_i)$. Dans un second temps on va chercher à les calculer.
   (a) Justifier le fait que $z_1 = z_n = 0$.
   (b) Montrer que sur chaque intervalle $[x_i,x_{i+1}]$, $1\leqslant i\leqslant n-1$, on a
   $$s^{(2)}(x) = z_{i+1}\frac{x-x_i}{\delta x_i} + z_i\frac{x_{i+1}-x}{\delta x_i}. \tag{1}$$
   (c) En déduire que sur chaque intervalle, il existe b_i et a_i tels que
   $$s(x) = z_{i+1}\frac{(x-x_i)^3}{6\delta x_i} + z_i\frac{(x_{i+1}-x)^3}{6\delta x_i} + b_i(x-x_i)+a_i. \tag{2}$$
   (d) Déterminer a_i et b_i en utilisant les valeurs de s en x_i. En déduire que les z_i détermine entièrement s sur $[a,b]$.
   On va maintenant chercher les valeurs de z_k pour $2\leqslant k\leqslant n-1$.
   (e) En écrivant la continuité de s' aux points $(x_i)_{3\leqslant i\leqslant n-2}$, montrer que pour $3\leqslant i\leqslant n-2$ on a
   $$\frac{\delta x_i}{6}z_{i-1} + \left(\frac{\delta x_{i-1}}{3}+\frac{\delta x_i}{3}\right)z_i + \frac{\delta x_i}{6}z_{i+1} = \frac{\delta y_i}{\delta x_i} - \frac{\delta y_{i-1}}{\delta x_{i-1}}. \tag{3}$$
   (f) En utilisant le fait que $z_1 = z_n = 0$ écrire deux relations similaires faisant intervenir respectivement z_2, z_3 et z_{n-1}, z_n).
   (g) En déduire que le vecteur $(z_k)_{2\leqslant k\leqslant n-1}$ est solution d'un système linéaire que l'on précisera.
   (h) Que réalise le code matlab suivant ?

```
function S=Spline3(a,b,x,y,N)
n=length(x);
dx=diff(x);
dy=diff(y);
M=zeros(n-2,n-2);
M(1,1)=2*dx(1)+2*dx(2);M(1,2)=dx(2);
M(n-2,n-3)=dx(n-2);M(n-2,n-2)=2*dx(n-2)+2*dx(n-1);
k=2;
if n>4
   for k=2:n-3
       M(k,k-1)=dx(k);M(k,k)=2*(dx(k)+dx(k+1));M(k,k+1)=dx(k+1);
   end
end
u=dy./dx;
du=diff(u)';
z=6*inv(M)*du;
z=[0;z;0];
t=a+(b-a)*[0:1/(N-1):1];
k=1;
S=0*t;
while t(k)<x(1)
    coef=dy(1)/dx(1)-z(2)*dx(1)/6;
   S(k)=coef*(t(k)-x(1))+y(1);
   k=k+1;
end
while t(k)<x(n)
   I=sum(t(k)>x);
   A=z(I+1)*(t(k)-x(I)).^3/(6*dx(I));
   B=z(I)*(x(I+1)-t(k)).^3/(6*dx(I));
   C=(t(k)-x(I))*(dy(I)/dx(I)-(z(I+1)-z(I))*dx(I)/6);
   D=y(I)-z(I)*dx(I)^2/6;
   S(k)=A+B+C+D;
   k=k+1;
```

```
end
while k<(N+1)
    coef=dy(n-1)/dx(n-1)+z(n-1)*dx(n-1)/6;
    S(k)=coef*(t(k)-x(n))+y(n);
    k=k+1;
end
plot(t,S);
hold on;plot(x,y,'or');
hold off;
```

10. Tester le code sur des exemples (polynôme dans un premier temps, sur un sinus ...).
11. Comparer sur une même fonction, l'interpolation par splines et par polynômes de Lagrange.