数 値 解 析 20 － 5
(1 9 8 7. 3. 1 4)

補間法に関する一注意

一松 信

京都大学／数理解析研究所

$f(x)$が、次数の上限 n が既知の多項式ならば、n+1 個の変数値x_iに対する関数値$f(x_i)$ から $f(x)$ を求めることができる。 有限体上のように誤差のない計算ができる場合は、特に有用であり、x_iをうまく選ぶことによって計算の手間を減らすことができる。

A note on the interpolation.

Sin HITOTUMATU

Research Institute for Mathematical Sciences/Kyoto University

(Kitashirakawa-Oiwakecho Sakyo-ku,Kyoto,JAPAN-606)

If $f(x)$ is a polynomial of degree at most n,we can determine it through the functional values $f(x_i)$ at n+1 distinct points x_i. This is particularly useful if we can compute them without errors such as the computations over a finite field.

Taking suitable sequence of the nodes x_i ,we can diminish the computational time.

1. 序

f(x)がn次以下の多項式ならば、n+1 個の点x_iでの値 $f(x_i)$によって、fを求めることができ、それが補間法そのものである。 数式処理において、記憶容量超過を避けるために、変数に数値を代入して計算し、それから多項式であることが既知の関数を復元する方法が、「生活の知恵」として有用である。

このような場合には、x_iをうまく取ることによって、計算の手間を減らすことができる。

以下に述べる方法は、実の所故高橋秀俊先生がずっと以前に注意していた話であり、新しい内容は少ない。

以下では、例えば mod pでの計算のように、誤差のない計算ができる場合を想定する。 普通の浮動小数点演算に機械的に適用すると、ひどい不安定性を生ずることは周知である。
(誤差のない計算などは、「数値解析」の範囲外らしいが、御寛容を乞いたい。)

2. 対称な節点

f(x)が2m次、すなわち偶数次の多項式の場合には、節点x_iを 0,±1,±2,...,±m と原点について対称にとり、次のように2次ずつ上げて行くと便利である。 x_iでの値を $f(x_i)$とする。

y_0= f(0); $f_0(x)=y_0$ (constant);
for k:=1 to m do (${f_{k-1}(x)}$ has been known)
[$y_k:=f(k) - f_{k-1}(k)$; $y_{-k}:=f(-k)-f_{k-1}(-k)$;
$f_k(x):= f_{k-1}(x)+x(x^2-1)(x^2-4)\ldots(x^2-(k-1)^2)(a_k+b_kx)$ where
$a_k:= (y_k -y_{-k})/2\cdot(2k-1)!$; $b_k:=(y_k+y_{-k})/(2k)!$;]
$f(x):= f_m(x)$;

例えば初めの方は(普通の数学のように書く)

$f_1(x)=y_0 +a_1x +b_1x^2$; $a_1=(f(1)-f(-1))/2$; $b_1=(f(1)-2f(0)+f(-1))/2$;

$f_2(x)=f_1(x)+(x^3-x)(a_2 +b_2x)$; $a_2=(y_2-y_{-2})/12$; $b_2=(y_2+y_{-2})/24$;
$y_2=f(2)-f_1(2)$; $y_{-2}=f(-2)-f_1(-2)$;

である。 これらは通常の浮動小数点数でも、それ程困難なく計算できる。

近年記憶装置が安価になったため、補間法が再認識されてきた。 等間隔の表で3個の値による2次補間が簡便である。 計算機は高度の計算能力を有するから、その昔の手計算時代のように、線型補間にこだわるべきではない。

例えば sin,cos は9°(1/10 直角) 刻みの7桁の表から2次補間により、最大 2×10^{-7} の誤差で評価できる。

3.　1の累乗根の利用

これは特別な素数pと次数nの場合にしか有効でないが、適用可能な場合には劇的といってよい程早くできる。

定まった素数pに関する mod pの計算をすると仮定する。　具体的に求めることは困難なことが多いが、p-1 の約数mに対しては、1の原始m乗根wがある。　$x_i=w^i$,i=0,1,...,m-1　と取ると、対応するm-1 次の多項式f(x)は以下のように簡単に計算できる。

$y_k=f(w^k);k=0,1,...,m-1$　　が既知のとき

$f(x)=a_0+a_1x+a_2x^2+...+a_{m-1}x^{m-1}$　　とおくと

$a_j=(y_0+y_1w^{-j}+y_2w^{-2j}+...+y_{m-1}w^{-j(m-1)})/m$　　(計算はmod p の意味で)

$j=0,1,\cdots,m-1.$

mが合成数ならばFFT あるいはその変形算法が適用できる。

例　p=41　に対して，次の行列の固有多項式を求める。

$$\begin{bmatrix} -2 & -2 & -3 & 17 & 2 & -14 & 20 & 20 & 18 & -15 \\ 6 & 8 & 10 & 8 & 8 & -7 & 18 & 9 & 16 & -11 \\ -1 & 8 & -13 & 17 & -10 & -12 & 20 & -16 & 10 & -5 \\ 8 & -9 & -4 & -19 & 10 & 16 & 5 & 14 & -20 & -1 \\ 3 & -15 & -15 & 8 & -7 & -19 & 10 & 8 & 13 & 15 \\ 9 & -8 & -16 & 0 & -19 & -13 & 3 & 4 & 4 & 4 \\ -19 & 13 & 3 & 14 & 12 & 16 & 2 & -18 & -16 & -13 \\ 8 & -13 & 15 & -12 & 15 & -13 & 8 & 1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

これは実は　$x^{10}+x+1$ に対する Berlekamp行列である。　一般にこの固有多項式は

$$\prod_{i=1}^{\ell}(x^{e_i}-1),\quad e_1+...+e_\ell=10$$

の形に因数分解され、そのとき元の多項式がe_1次,...,e_ℓ 次の因子に分解されることが、 mod pでの因数分解の一般論から知られている。

p=41に対しては、 7が一つの原始根であり、m=10 (p-1=40の約数) に対する原始 m乗根は w=-18 (7^4 に当る) である。　自明な因子 (1-x)を除き、9 次の首座小行列A に対して、

$$f(x)=\det(A-xI)$$

にxの数値を代入した値を、標準的なGauss の消去法(mod 41)で計算する。　このとき真に0になったときだけ部分枢軸選びを行い、もしすべてが0になって選択が不可能なら、その段階で行列式を0とする。　これはマイコンで容易にでき、次の表のような結果を得た。

i	0	1	2	3	4	5	6	7	8	9
w^i	1	-18	-4	-10	16	-1	18	4	10	-16
y_i	0	11	0	19	0	2	0	-15	0	-7
$\sum_j y_j w^{-ij}$	10	10	10	10	10	-10	-10	-10	-10	-10
a_i	1	1	1	1	1	-1	-1	-1	-1	-1

$$f(x)=1+x+x^2+x^3+x^4-x^5-x^6-x^7-x^8-x^9=(1+x+x^2+x^3+x^4)(1-x^5)$$

$$\text{固有多項式}=(1-x)f(x)=(x^5-1)^2$$

従って初めの多項式は２個の５次因子の積に分解されるはずである。 実際前にこの研究会で述べた通り

$$x^{10}+x+1=(x^5-2x^4+3x^3+x^2-13x-17)(x^5+2x^4+x^3-5x^2-2x+12) \pmod{41}$$

である。

[参考文献]

[1] E.R.Berlekamp, Factoring polynomials over large finite fields, Math. Comp. 24 (1970), 713-735.

[2] 高橋秀俊, FFTアルゴリズムについて, 数理研講究録, No.172, 1973, 38-57.

[3] 一松 信, 多項式の因数分解に関する注意, 情報処理学会数値解析研究会, 1985, Oct. 5.

[4] 一松 信, 教室に電卓を！ III, 海鳴社, 1986, (第1章 §1.7)