Homework 2 – A Report on Interpolation and Polynomial Approximation

Xiaolong Liu, 201900170025

24th.Oct.2021

Shandong University, Taishan College

Abstract

We use three interpolation polynomials to compute the function $\log(x^2 + e^x)$.

Contents

1 Introductions

We use three interpolation polynomials to compute the function $\log(x^2 + e^x)$.

2 Algorithm Description & Mathematical Derivation

2.1 Newton's Interpolation Polynomial

First we define the k-th divided-difference of f with respect to x_i. The 0-th divided-difference of f are just $f[x_i] = f(x_i)$. We write inductively, if we have $(k-1)$-st divided differences $f[x_i, ..., x_{i+k-1}]$ and $f[x_{i+1}, ..., x_{i+k}]$, then the k-th divided difference is

$$f[x_1, ..., x_{i+k}] = \frac{f[x_{i+1}, ..., x_{i+k}] - f[x_i, ..., x_{i+k-1}]}{x_{i+k} - x_i}.$$

Let $a_k = f[x_0, ..., x_k]$, then $P_n(x) = f[x_0] + \sum_{k=1}^{n} f[x_0, ..., x_k](x - x_0) \cdots (x - x_{k-1})$. This is what we want. When the nodes are arranged consecutively with equal spacing, e.g. $x =$

$x_0 + sh, h = x_{i+1} - x_i$, then $P_n(x) = f(x_0) + \sum_{k=1}^{n} \binom{s}{k} k! h^k f[x_0, x_i, ..., x_k]$. Use the forward-difference notation Δ,we have $P_n(x) = f(x_0) + \sum_{k=1}^{n} \binom{s}{k} \Delta^k f(x_0)$. Use the backward-difference notation ∇,we have $P_n(x) = f(x_n) + \sum_{k=1}^{n} (-1)^k \binom{-s}{k} \nabla^k f(x_n)$.

2.2 Piecewise Hermite Interpolation Polynomial

Theorem 2.2.1. *If $f \in C^1[a, b]$ and $x_0, ..., x_n \in [a, b]$ are distinct, the unique polynomial of least degree agreeing with f and f' at $x_0, ..., x_n$ is the Hermite polynomial of degree at most $2n + 1$ given by*

$$H_{2n+1}(x) = \sum_{j=0}^{n} f(x_j) H_{n,j}(x) + \sum_{j=0}^{n} f'(x_j) \widehat{H}_{n,j}(x),$$

where, for $L_{n,j}(x)$ denoting the j-th Lagrange coefficient polynomial of degree n, we have

$$H_{n,j}(x) = (1 - 2(x - x_j) L'_{n,j}(x_j)) L^2_{n,j}(x), \widehat{H}_{n,j}(x) = (x - x_j) L^2_{n,j}(x).$$

2.3 Natural cubic spline Interpolation Polynomial

Given a function f defined on $[a, b]$ and a set of nodes $a = x_0 < x_1 < ... < x_n = b$, a cubic spline interpolant S for f is a function that satisfies the following conditions:

(a) $f(x)$ is a cubic polynomial, denoted $S_j(x)$, on the subinterval $[x_j, x_{j+1}]$ for each $j = 0, 1, ..., n - 1$;

(b) $S_j(x_j) = f(x_j)$ and $S_j(x_{j+1}) = f(x_{j+1})$ for each $j = 0, 1, ..., n - 1$;

(c) $S_{j+1}(x_{j+1}) = S_j(x_{j+1})$ for each $j = 0, 1, ..., n - 2$;

(d) $S'_{j+1}(x_{j+1}) = S'_j(x_{j+1})$ for each $j = 0, 1, ..., n - 2$;

(e) $S''_{j+1}(x_{j+1}) = S''_j(x_{j+1})$ for each $j = 0, 1, ..., n - 2$;

(f) One of the following sets of boundary conditions is satisfied:

(i) $S''(x_0) = S''(x_n)$ (natural boundary);

(ii) $S'(x_0) = f'(x_0)$ and $S'(x_n) = f'(x_n)$ (clamped boundary).

3 Codes and Numerical Experiments

3.1 Newton's Interpolation Polynomial

Problem 3.1.1. *Suppose that* $x_0 = -1.5, x_1 = -1, x_2 = -0.5, x_3 = 0, x_4 = 0.5$ *and* $f(x) = \log(x^2 + e^x), x \in [-2, 1]$. *Determine the Newton's interpolating polynomial.*

Solution. Consider the following code:

```
format long
x0=-1.5;x1=-1;x2=-0.5;x3=0;x4=0.5;
f0=log(1.5*1.5+exp(-1.5));f1=log(1*1+exp(-1));f2=log(0.5*0.5+exp
    (-0.5));
f3=log(exp(0));f4=log(0.5*0.5+exp(0.5));
f01=(f1-f0)/(x1-x0);f12=(f2-f1)/(x2-x1);f23=(f3-f2)/(x3-x2);
f34=(f4-f3)/(x4-x3);
f012=(f12-f01)/(x2-x0);f123=(f23-f12)/(x3-x1);f234=(f34-f23)/(x4-x2);
f0123=(f123-f012)/(x3-x0);f1234=(f234-f123)/(x4-x1);
f01234=(f1234-f0123)/(x4-x0);
```

Then we have

i	x_i	$f[x_i]$	$f[x_{i-1}, x_i]$	$f[x_{i-2}, x_{i-1}, x_i]$	$f[x_{i-3}, ..., x_i]$	$f[x_{i-4}, ..., x_i]$
0	-1.5	0.905484				
			-1.184445			
1	-1	0.313261		0.248192		
			-0.936253		0.665194	
2	-0.5	-0.154865		1.245984		-0.423715
			0.309730		-0.182235	
3	0	0		0.972631		
			1.282361			
4	0.5	0.641181				

Then we have

$$\begin{aligned} P_4(x) = & 0.905484 - 1.184445(x + 1.5) + 0.248192(x + 1.5)(x + 1) \\ & + 0.665194(x + 1.5)(x + 1)(x + 0.5) \\ & - 0.423715(x + 1.5)(x + 1)(x + 0.5)x. \end{aligned}$$

Well done. □

3.2 Piecewise Hermite Interpolation Polynomial

Problem 3.2.1. *Suppose that* $x_0 = -1.5, x_1 = -0.5, x_2 = 0.5$ *and* $f(x) = \log(x^2 + e^x), x \in [-2, 1]$. *Determine the piecewise Hermite interpolating polynomial.*

Solution. Consider the following code:

```
format long
x0=-1.5;x1=-0.5;x2=0.5;
z0=x0;z1=x0;z2=x1;z3=x1;z4=x2;z5=x2;
f0=log(1.5*1.5+exp(-1.5));f1=log(1.5*1.5+exp(-1.5));f2=log(0.5*0.5+
    exp(-0.5));
f3=log(0.5*0.5+exp(-0.5));f4=log(0.5*0.5+exp(0.5));f5=log(0.5*0.5+exp
    (0.5));
f01=(2*(-1.5)+exp(-1.5))/(1.5*1.5+exp(-1.5));f12=(f2-f1)/(z2-z1);
f23=(2*(-0.5)+exp(-0.5))/(0.5*0.5+exp(-0.5));f34=(f4-f3)/(z4-z3);
f45=(2*(0.5)+exp(0.5))/(0.5*0.5+exp(0.5));
f012=(f12-f01)/(z2-z0);f123=(f23-f12)/(z3-z1);f234=(f34-f23)/(z4-z2);
    f345=(f45-f34)/(z5-z3);
f0123=(f123-f012)/(z3-z0);f1234=(f234-f123)/(z4-z1);f2345=(f345-f234)
    /(z5-z2);
f01234=(f1234-f0123)/(z4-z0);f12345=(f2345-f1234)/(z5-z1);
f012345=(f12345-f01234)/(z5-z0);
```

Then we have

z	$f(z)$	1st	2ed	3rd	4th	5th
-1.5	0.905484					
		-1.122815				
-1.5	0.905484		0.062466			
		-1.060349		0.538508		
-0.5	-0.154865		0.600974		-0.105642	
		-0.459375		0.327223		-0.193100
-0.5	-0.154865		1.255421		-0.491844	
		0.796045		-0.656464		
0.5	0.641180		0.598956			
		1.395002				
0.5	0.641180					

Then we have

$$H_5(x) = 0.905484 - 1.122815(x+1.5) + 0.062466(x+1.5)^2 + 0.538508(x+1.5)^2(x+0.5) - 0.105642(x+1.5)^2(x+0.5)^2 - 0.193100(x+1.5)^2(x+0.5)^2(x-0.5).$$

Well done. □

3.3 Natural cubic spline Interpolation Polynomial

Problem 3.3.1. *Suppose that* $x_0 = -2, x_1 = -1, x_2 = 0, x_3 = 1$ *and* $f(x) = \log(x^2 + e^x), x \in [-2, 1]$. *Determine the natural cubic spline interpolating polynomial.*

Solution. Consider the following code:

```
format long
fb=3*(c-b)-3*(b-a)
   fb =1.1478
fc=3*(d-c)-3*(c-b);fd=3*(e-d)-3*(d-c);
l=1;u=0;z=0;l=4-u;l =4;l=4-u;u=1/lz=(fb-z)/l;
l=4-u;u=1/l;z=(fc-z)/l;l=4-u;u=1/l;z=(fd-z)/l;
c=0.3804;c=0.6516-0.2667*0.3084;c=0.287-0.25*0.5693;
c =0.1447;b3=(d-c)-(0+2*0.3084)/3;b3 =-1.0039;
b2=(c-b)-(0.3804+2*0.5693)/3;b1=(b-a)-(0.5694+2*0.1447)/3;
d3=0-0.3804;d2=0.3804-0.5693;d1=0.5693-0.1447;d3=d3/3;
d2=d2/3;d1=d1/3;
```

Then we have

$$S(x) = \begin{cases} 0.06844x^3 + 0.1483x^2 - 1.141x - 0.9093 & x \in [-2, -1] \\ 0.5568x^3 + 2.1x^2 + 1.234x + 0.01059 & x \in [-1, 0] \\ -0.332x^3 + 0.5506x^2 + 1.096x - 0.004392 & x \in [0, 1] \end{cases}$$

Well done! □

4 Discussions and Conclusions

The Newton's interpolation polynomial is rough but we can easy to compute. The piecewise Hermite interpolation polynomial has more computation but one can get the target function more precisely and we can even make the first derivative of approximated polynomial tangent to the given function. Natural cubic spline we can use the cubic polynomial to compute it piecewisely and get more precise one.

References

[1] Richard L. Burden, J. Douglas Faires, Annette M. Burden, *Numerical Analysis (Tenth Edition)*, Cengage Learning, 2014.