2021 Spring MAS 365: Homework 7

posted on May 13; due by May 20

1. [10+10 points]

   (a) Let $f \in C^2[a,b]$, and let the nodes $a = x_0 < x_1 < \cdots < x_n = b$ be given. Derive an error estimate similar to that in Theorem 3.13 in the textbook for the piecewise linear interpolating function F.

   (b) A clamped cubic spline s for a function f is defined by

$$s(x) = \begin{cases} s_0(x) = 1 + Bx + 2x^2 - 2x^3, & \text{if } 0 \le x < 1, \\ s_1(x) = 1 + b(x-1) - 4(x-1)^2 + 7(x-1)^3, & \text{if } 1 \le x \le 2. \end{cases}$$

   Find $f'(0)$ and $f'(2)$.

Solution:

(a) By Theorem 3.3 in the textbook, there exists ξ_j between x_j and x_{j+1} with

$$f(x) = F(x) + \frac{f''(\xi_j)}{2}(x - x_j)(x - x_{j+1}).$$

With $M = \max_{a \le x \le b} |f''(x)|$, we have

$$\begin{aligned} |f(x) - F(x)| &= \max_{0 \le j \le n-1} \max_{x_j \le x \le x_{j+1}} \left| \frac{f''(\xi_j)}{2}(x - x_j)(x - x_{j+1}) \right| \\ &\le \max_{0 \le j \le n-1} \frac{M}{2} \left| \left(\frac{x_j + x_{j+1}}{2} - x_j \right) \left(\frac{x_j + x_{j+1}}{2} - x_{j+1} \right) \right| \\ &= \frac{M}{8} \max_{0 \le j \le n-1} (x_{j+1} - x_j)^2 \end{aligned}$$

for all x in $[a,b]$.

(b) Note that

$$s'(x) = \begin{cases} s_0'(x) = B + 4x - 6x^2, & \text{if } 0 \le x < 1, \\ s_1'(x) = b - 8(x-1) + 21(x-1)^2, & \text{if } 1 \le x \le 2. \end{cases}$$

$$s''(x) = \begin{cases} s_0''(x) = 4 - 12x, & \text{if } 0 \le x < 1, \\ s_1''(x) = -8 + 42(x-1), & \text{if } 1 \le x \le 2. \end{cases}$$

We have

$$s_0(1) = 1 + B = s_1(1) = 1, \quad s_0'(1) = B - 2 = s_1'(1) = b, \quad s_0''(1) = -8 = s_1''(1) = -8.$$

Then solving this system of equation gives

$$B = 0, \quad b = -2,$$

and thus

$$f'(0) = s'(0) = B = 0, \quad f'(2) = S'(2) = b + 13 = 11.$$

2. [10+10 points]

   (a) Use the most accurate three-point formula to determine each missing entry in the following table.

x	$f(x)$	$f'(x)$
-0.3	-0.27652	
-0.2	-0.25074	
-0.1	-0.16134	
0	0	

   (b) The data were taken from the function $f(x) = e^{2x} - \cos 2x$. Compute the actual error, and find error bounds using the error formulas. Note that the $f^{(3)}(x)$ is nonincreasing function on $[-0.3, 0]$.

Solution:

(a) We use the three-point midpoint formula for the points $x = -0.2$ and -0.1

$$f'(x_0) = \frac{1}{2h}[f(x_0 + h) - f(x_0 - h)],$$

the three-point endpoint formula for the point $x = -0.3$

$$f'(x_0) = \frac{1}{2h}[-3f(x_0) + 4f(x_0 + h) - f(x_0 + 2h)],$$

and another three-point endpoint formula for the point $x = 0$

$$f'(x_0) = \frac{1}{2h}[f(x_0 - 2h) - 4f(x_0 - h) + 3f(x_0)].$$

x	$f(x)$	$f'(x)$
-0.3	-0.27652	-0.06030
-0.2	-0.25074	0.57590
-0.1	-0.16134	1.25370
0	0	1.97310

(b) The error bound for the midpoint formula is

$$\frac{h^2}{6}|f^{(3)}(\xi)| \le \frac{0.1^2}{6} \max_{x_0-h\le\xi\le x_0+h} |f^{(3)}(\xi)|$$

and the error bound for the endpoint formula is either

$$\frac{h^2}{3}|f^{(3)}(\xi)| \le \frac{0.1^2}{3} \max_{x_0\le\xi\le x_0+2h} |f^{(3)}(\xi)|, \quad \text{or} \quad \frac{h^2}{3}|f^{(5)}(\xi)| \le \frac{0.1^2}{3} \max_{x_0-2h\le\xi\le x_0} |f^{(3)}(\xi)|.$$

Since $f^{(3)}(x) = 8e^{2x} - 8\sin 2x$, we have the following table.

x	Actual error	Error bound
-0.3	0.028638	0.029692
-0.2	0.014097	0.014846
-0.1	0.013577	0.014130
0	0.026900	0.028260

3. [10 points] Suppose that $N(h)$ is an approximation to M for every $h > 0$ and that

$$M = N(h) + K_1 h^2 + K_2 h^4 + K_3 h^6 + \cdots ,$$

for some constants $K_1, K_2, K_3, \ldots$. Use the values $N(h)$, $N\left(\frac{h}{3}\right)$, and $N\left(\frac{h}{9}\right)$ to produce and $O(h^6)$ approximation to M.

Solution: We have

$$M = N\left(\frac{h}{3}\right) + K_1 \frac{h^2}{9} + K_2 \frac{h^4}{81} + K_3 \frac{h^4}{3^6} + \cdots ,$$

and subtracting the equation in the problem from nine-times the above equation yields

$$8M = 8N\left(\frac{h}{3}\right) + \left[N\left(\frac{h}{3}\right) - N(h)\right] + K_2\left(\frac{h^4}{9} - h^4\right) + K_3\left(\frac{h^6}{81} - h^6\right) + \cdots .$$

Define

$$N_2(h) = N\left(\frac{h}{3}\right) + \frac{1}{8}\left[N\left(\frac{h}{3}\right) - N(h)\right],$$

and then we have an $O(h^4)$ approximation to M

$$M = N_2(h) - K_2 \frac{h^4}{9} - K_3 \frac{10h^6}{81} - \cdots . \tag{1}$$

Replacing h by $h/3$ yields

$$M = N_2\left(\frac{h}{3}\right) - K_2 \frac{h^4}{3^6} - K_3 \frac{10h^6}{3^{10}} - \cdots , \tag{2}$$

and subtracting the equation (**??**) from 81-times the equation (**??**) gives

$$80M = 80N_2\left(\frac{h}{3}\right) + \left[N_2\left(\frac{h}{3}\right) - N_2(h)\right] - K_3\left(\frac{10h^6}{3^6} - \frac{10h^6}{81}\right) - \cdots .$$

Define

$$N_3(h) = N_2\left(\frac{h}{3}\right) + \frac{1}{80}\left(N_2\left(\frac{h}{3}\right) - N_2(h)\right),$$

and we have an $O(h^6)$ approximation to M

$$M = N_3(h) + K_3 \frac{h^6}{3^6} + \cdots .$$

4. [5+5 points]

   (a) The quadrature formula $\int_0^2 f(x)dx = c_0 f(0) + c_1 f(1) + c_2 f(2)$ is exact for all polynomials of degree less than or equal to two. Determine c_0, c_1 and c_2.

   (b) Find the constants c_0, c_1, and x_1 so that the quadrature formula

   $$\int_0^1 f(x)dx = c_0 f(0) + c_1 f(x_1)$$

   has degree of precision 2.

Solution:

(a) We have that

$$\begin{aligned} f(x) = 1: &\quad 2 = c_0 + c_1 + c_2, \\ f(x) = x: &\quad 2 = c_1 + 2c_2, \\ f(x) = x^2: &\quad \frac{8}{3} = c_1 + 4c_2 \end{aligned}$$

so solving them gives $c_0 = \frac{1}{3}$, $c_1 = \frac{4}{3}$, $c_2 = \frac{1}{3}$.

(b) We have that

$$\begin{aligned} f(x) = 1: &\quad 1 = c_0 + c_1, \\ f(x) = x: &\quad \frac{1}{2} = c_1 x_1, \\ f(x) = x^2: &\quad \frac{1}{3} = c_1 x_1^2 \end{aligned}$$

so solving them gives $c_0 = \frac{1}{4}$, $c_1 = \frac{3}{4}$, $x_1 = \frac{2}{3}$.

5. [10+10 points]

(a) Implement Newton-Cotes formulas via MATLAB grader.

(b) Impement composite numerical intergration methods via MATLAB grader.

Solution:

(a)
```
function approx = newton_cotes(f, a, b, n, Closed)
    if Closed
        % Closed Newton-Cotes
        if n == 1
            h = b-a; approx = h/2*(f(a) + f(b));
        elseif n == 2
            h = (b-a)/2; approx = h/3*(f(a) + 4*f(a+h) + f(b));
        elseif n == 3
            h = (b-a)/3; approx = 3*h/8*(f(a) + 3*f(a+h) + 3*f(a+2*h) + f(b));
        elseif n == 4
            h = (b-a)/4; approx = 2*h/45*(7*f(a) + 32*f(a+h) + 12*f(a+2*h) + 32*f(a+3*h)
        end
    else
        % Open Newton-Cotes
        if n==0
            h = (b-a)/2; approx = 2*h*(f(a+h));
        elseif n==1
            h = (b-a)/3; approx = 3*h/2*(f(a+h) + f(a+2*h));
        elseif n==2
            h = (b-a)/4; approx = 4*h/3*(2*f(a+h) - f(a+2*h) + 2*f(a+3*h));
        elseif n==3
            h = (b-a)/5; approx = 5*h/24*(11*f(a+h) + f(a+2*h) + f(a+3*h) + 11*f(a+4*h))
        end
    end
end
```

(b)
```
function [s1 s2 s3] = composite_integration(f, a, b, n)
    % Composite Trapezoidal rule
    h = (b-a)/n;
    tmp = 0;
    for i=1:n-1
        tmp = tmp + f(a+i*h);
    end
    s1 = h/2*(f(a) + 2*tmp + f(b));
    % Composite Simpson's rule
    h = (b-a)/n;
    tmp1 = 0; tmp2 = 0;
    for i=1:n-1
        if mod(i,2) == 0
           tmp1 = tmp1 + f(a+i*h);
        else
           tmp2 = tmp2 + f(a+i*h);
        end
    end
    s2 = h/3*(f(a) + 2*tmp1 + 4*tmp2 + f(b));
    % Composite Midpoint rule
    h = (b-a)/(n+2);
    tmp = 0;
    for i=0:2:n
       tmp = tmp + f(a+(i+1)*h);
    end
    s3 = 2*h*tmp;
end
```