Exercise-3 for Numerical Methods

Student: M202370666 高超

Notices: All my codes written in this homework have been uploaded to GitHub. Thus, you can check my codes in Ermisgc/Numerical-Method---W3: 数值方法第三次作业的源代码 (github.com)

1. The accompanying table gives the hourly temperature readings (Fahrenheit) during a 12-hour period in a suburb of Los Angeles. Find the natural cubic spline for the data. Graph the natural cubic spline and the data on the same coordinate system. Use the natural cubic spline and the results of part(a) of Exercise 12 to approximate the average temperature during the 12-hour period.

Time, a.m.	Degrees	Time, a.m.	Degrees
1	58	7	57
2	58	8	58
3	58	9	60
4	58	10	64
5	57	11	67
6	57	Noon	68

Solution:

Because of the natural cubic spline, We set the boundary conditions to

$$m_0 = m_1 = 0.$$

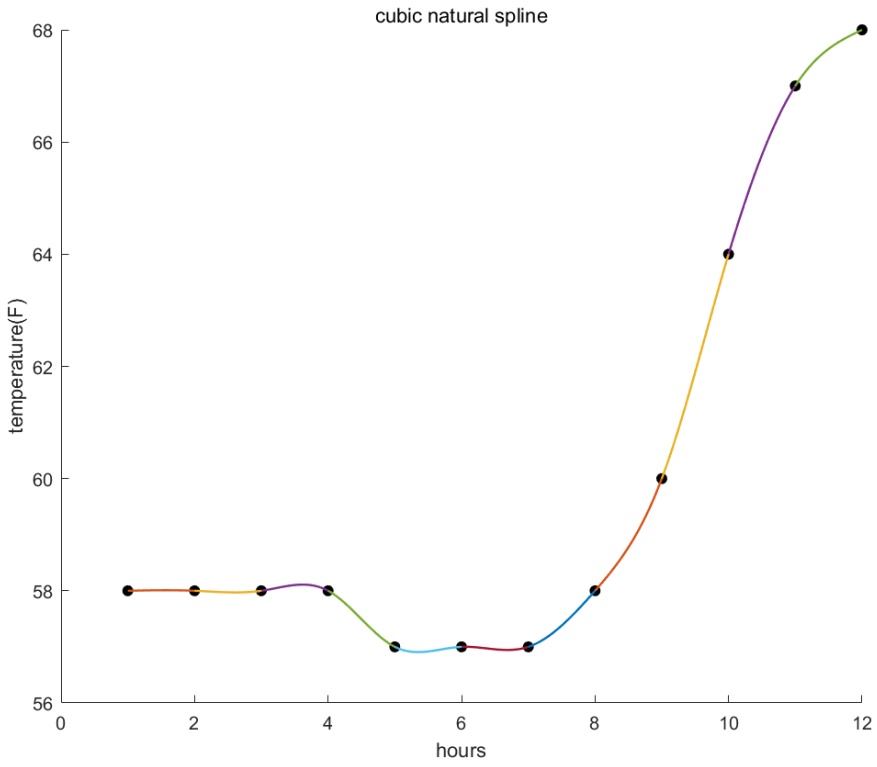
Then recursing equation at the header is

$$2(h_0 + h_1)m_1 + h_1 m_2 = u_1.$$

Recursing equation at the tail is

$$h_{n-2}m_{n-2} + 2(h_{n-2} + h_{n-1})m_{n-1} = u_{n-1}$$

Then it's easy to compute and plot the scatter of the spline. The result is shown below.



In Exercise 12 we are assigned to give a formula for $\int_{x_k}^{x_{k+1}} S_k(x)dx$. The formula of $S_k(x)$ is:

$$S_k(x) = S_{k,0} + S_{k,1}(x - x_k) + S_{k,2}(x - x_k)^2 + S_{k,3}(x - x_k)^3.$$

Then it's easy to get:

$$\begin{aligned}\int_{x_k}^{x_{k+1}} S_k(x)dx &= \int_{x_k}^{x_{k+1}} S_{k,0} + S_{k,1}(x - x_k) + S_{k,2}(x - x_k)^2 + S_{k,3}(x - x_k)^3 dx \\ &= S_{k,0}x + \frac{1}{2}S_{k,1}(x - x_k)^2 + \frac{1}{3}S_{k,2}(x - x_k)^3 + \frac{1}{4}S_{k,3}(x - x_k)^4 \Big|_{x_k}^{x_{k+1}} \\ &= S_{k,0}h_k + \frac{1}{2}S_{k,1}h_k^2 + \frac{1}{3}S_{k,2}h_k^3 + \frac{1}{4}S_{k,3}h_k^4.\end{aligned}$$

We use the formula to get the average temperature:

$$mean = \frac{\sum_{k=1}^{11} \int_{x_k}^{x_{k+1}} S_k(x)dx}{11}.$$

The result is

$$mean = 59.7231.$$

Actually we can get the average directly by adding them together and dividing them by the count of the numbers, which is

$$average = \frac{58 + 58 + 58 + \cdots + 64 + 67 + 68}{12} = 60.$$

The values of *mean* and *average* are almost the same, which can prove that the formula we derive is correct.

2. Let $f(x)$ be given by the table following. The inherent round-off error has the bound $|e_k| \leq 5 \times 10^{-6}$. Use the rounded values in your calculations.

x	$f(x) = \ln(x)$
2.900	1.06471
2.990	1.09527
2.999	1.09828
3.000	1.09861
3.001	1.09895
3.010	1.10194
3.100	1.13140

(a) Find approximation for $f'(3.0)$ using formula $f'(x_0) \approx \frac{y_1 - y_{-1}}{2h}$ with $h = 0.1$, $h = 0.01$ and $h = 0.001$.

(b) Compare with $f'(3.0) = \frac{1}{3} \approx 0.33333$.

(c) Find the total error bound $E(f, h) = \frac{e_1 - e_{-1}}{2h} - \frac{h^2 f^{(3)}(c)}{6}$ for the three cases in part(a).

Solution:

(a) When $h = 0.1$, $f'(3.0) \approx \frac{f(3.1) - f(2.9)}{2h} = \frac{1.13140 - 1.06471}{2*0.1} = 0.33345$.

When $h = 0.01$, $f'(3.0) \approx \frac{f(3.01) - f(2.99)}{2h} = \frac{1.10194 - 1.09527}{2*0.01} = 0.33350$.

When $h = 0.001$, $f'(3.0) \approx \frac{f(3.001) - f(2.999)}{2h} = \frac{1.09895 - 1.09828}{2*0.001} = 0.33500$.

(b) When $h = 0.1$, $error = |0.33345 - 0.33333| = 0.00012$.

When $h = 0.01, error = |0.33350 - 0.33333| = 0.00017$.

When $h = 0.001, error = |0.33500 - 0.33333| = 0.00167$.

It is an interesting phenomenon that at this situation, along with the decrease of h,

the error strangely gets bigger! That is why we have to choose an appropriate value of h instead of a value as small as possible.

(c) The third-derivative of $f(x)$ is :

$$f^{(3)}(x) = \frac{2}{x^3}.$$

Let $M = \max\{|f^{(3)}(x)|\} = \frac{2}{(3-h)^3}$.

The formula of total error bound can be transformed to:

$$|E(f,h)| \le \frac{|e_1| + |e_{-1}|}{2h} + \frac{Mh^2}{6} = \frac{|e_1| + |e_{-1}|}{2h} + \frac{h^2}{3(3-h)^3} = \frac{|e_k|}{h} + \frac{h^2}{3(3-h)^3}.$$

For $h = 0.1$, $|E(f,h)| \le \frac{5\times10^{-6}}{0.1} + \frac{0.1^3}{3(3-0.1)^3} = 0.000187.$

For $h = 0.01$, $|E(f,h)| \le \frac{5\times10^{-6}}{0.01} + \frac{0.01^3}{3(3-0.01)^3} = 0.000501.$

For $h = 0.001$, $|E(f,h)| \le \frac{5\times10^{-6}}{0.001} + \frac{0.001^3}{3(3-0.001)^3} = 0.005000.$

It's obvious that along the decrease of h, the error bound gets bigger.

3. Use Program 7.6 to approximate the value of the definite integral $\int_0^3 \frac{\sin(2x)}{1+x^5}dx$. Use the starting tolerance $\epsilon_0 = 0.00001$.

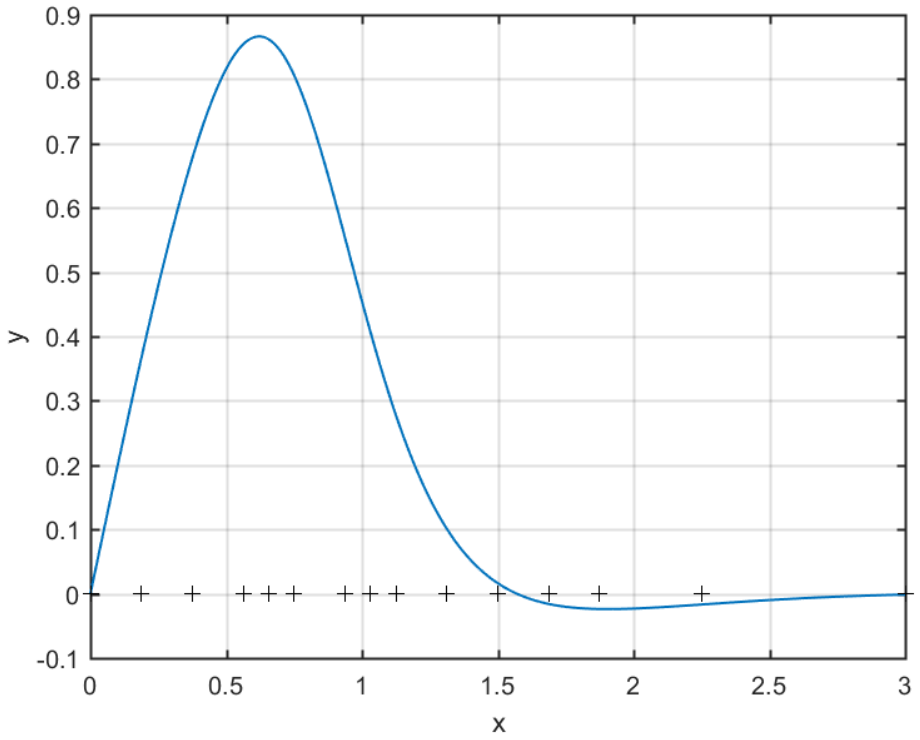
Solution:

Because the program 7.5 and 7.6 the book offers are kind of hard to intuitively understand, I code a new program that is more intelligible using recursion to solve this integral.

```
function [S, interval] = adapt(f,a,b,to1)
%   这里考虑用递归的写法来实现我自己的自适应函数
%   S 是最终的返回值
%   interval 是最终返回的区间
S_old = simpson(f,a,b);
S_new1 = simpson(f,a,(a + b)/2);
S_new2 = simpson(f,(a+b)/2,b);
%   递归的基
if(abs(S_new1 + S_new2 - S_old) < 16*to1)
    S = S_new1 + S_new2;
    interval = [a b];
else
```

```
    [S1,interval1] = adapt(f,a,(a+b)/2,to1/2);
    [S2,interval2] = adapt(f,(a+b)/2,b,to1/2);
    % 拼接区间：
    interval_length = length(interval1) + length(interval2) - 1;
    interval = zeros(1,interval_length);
    interval(1:length(interval1)) = interval1;
    interval(length(interval1) + 1 : interval_length) = interval2(2:length(interval
2));
    % 得到结果
    S = S1 + S2;
end
```

The more detail can be found in my GitHub page as written in the beginning of this document. The result plot are shown below.



The little crosses here are the corresponding bound of intervals.

4. By solving an appropriate initial value problem, make a table of values of the function $f(t)$ given by the following integral:

$$f(x) = \frac{1}{2} + \frac{1}{\sqrt{2\pi}} \int_0^x e^{-t^2/2} dt \quad \text{for } 0 \le x \le 3.$$

Use the Runge-Kutta method of order $N = 4$ with $h = 0.1$ for your computations.

Your solution should agree with the values in the following table. *Remark.* This is a good way to generate the table of areas for a standard normal distribution.

x	$f(x)$
0.0	0.5
0.5	0.6914625
1.0	0.8413448
1.5	0.9331928
2.0	0.9972499
2.5	0.9937903
3.0	0.9986501

Solution:

(1) Firstly we have to transform the function to a differential equation. Taking the derivative of both sides yields:

$$f'(x) = \frac{\boldsymbol{e^{-x^2/2}}}{\boldsymbol{\sqrt{2\pi}}}$$

(2) Use RK4 method to generate the sequence. The result sequence is shown below.

x	*RK4 Result*	x	*RK4 Result*
0	**0.5000000**	1.6	0.9452007
0.1	0.5398278	1.7	0.9554345
0.2	0.5792597	1.8	0.9640697
0.3	0.6179114	1.9	0.9712834
0.4	0.6554218	**2.0**	**0.9772499**
0.5	**0.6914625**	2.1	0.9821356
0.6	0.7257469	2.2	0.9860965
0.7	0.7580364	2.3	0.9892759
0.8	0.7881446	2.4	0.9918025
0.9	0.8159399	**2.5**	**0.9937903**
1.0	**0.8413448**	2.6	0.9953388

1.1	0.8643340	2.7	0.9965330
1.2	0.8849303	2.8	0.9974449
1.3	0.9031995	2.9	0.9981342
1.4	0.9192433	**3.0**	**0.9986501**
1.5	**0.9331928**		

Comparing my result with the values of standard normal distribution can make sure that my result is correct.

5. (a) Modify Programs 10.2 and 10.3 to accept the boundary conditions $u(0,t) = g_1(t) \neq 0$ and $u(a,t) = g_2(t) \neq 0$.

(b) Use your modified Program 10.3 to solve the heat equations in Problems 1 and 2, but use the boundary conditions

$$u(0,t) = g_1(t) = t^2 \quad \text{for } x = 0 \text{ and } 0 \leq t < 0.1,$$

$$u(1,t) = g_2(t) = t^2 \quad \text{for } x = 1 \text{ and } 0 \leq t < 0.1,$$

in place of $c_1 = c_2 = 0$.

Note: The heat equation is:

$$u_t(x,t) = c^2 u_{xx}(x,t), \text{ for } 0 < x < 1 \text{ and } 0 < t < 0.1.$$

Initial conditions are:

$$u(x,0) = f(x), \text{ for } t = 0 \text{ and } 0 \leq x \leq 1.$$

In Problem 1: $f(x) = sin(\pi x) + sin(2\pi x)$.

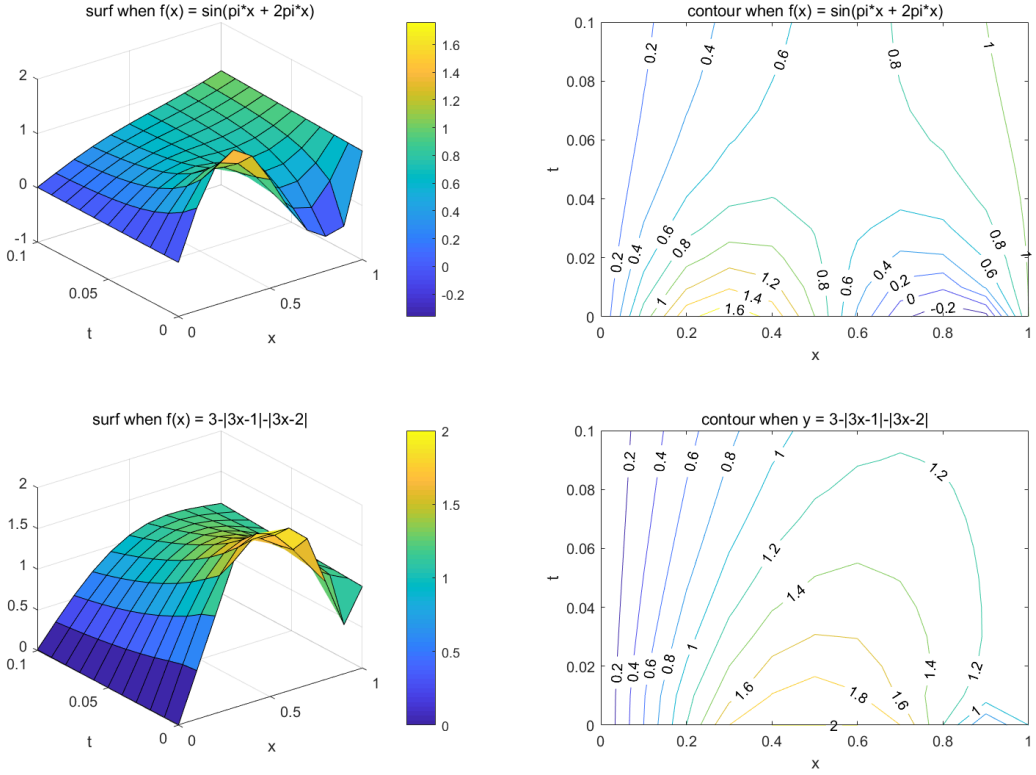
In Problem 2: $f(x) = 3 - |3x - 1| - |3x - 2|$.

(c) **Use the *surf* and *contour* commands to plot your approximate solutions.**

Solution:

My codes after modifying are put in my GitHub repository and I don't put them here.

The result after approximation is plotted as surfs and contours that are shown below together.



The top two figures use the $f(x) = sin(\pi x) + sin(2\pi x)$ of Problem 1 while the bottom two figures use the $f(x) = 3 - |3x - 1| - |3x - 2|$ of Problem 2.

What's worth mentioning is that the four figures are all results of Crank-Nicholson method, and if we use Forward-Difference methods here, the result is unstable because $r = 1 > \frac{1}{2}$, which means instability of the method.

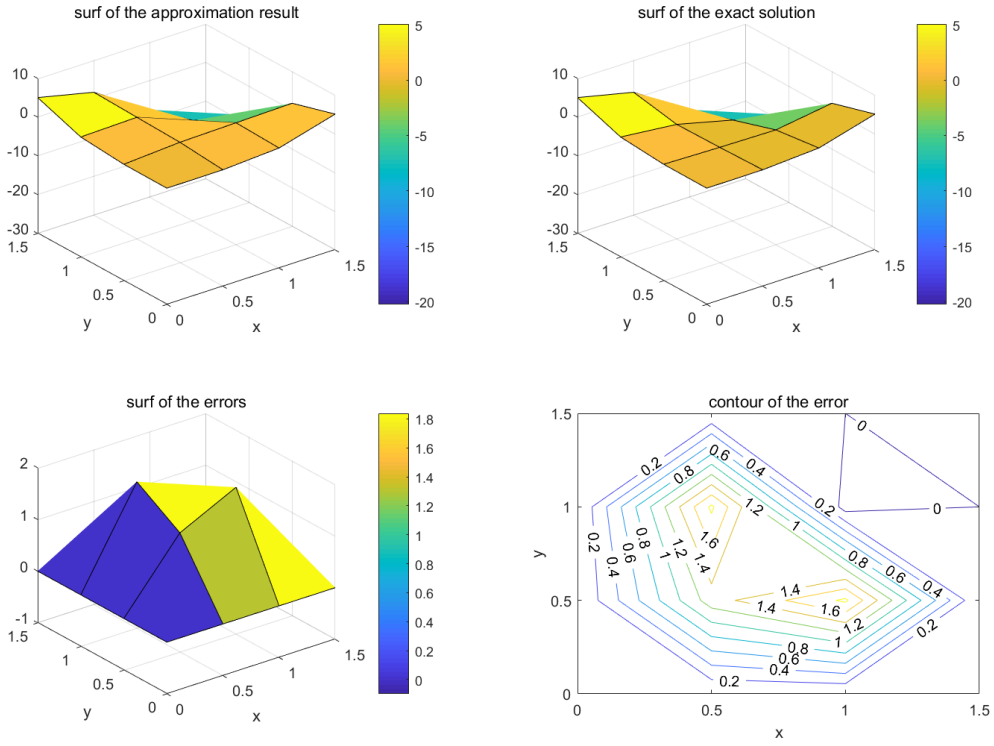
6. Use Program 10.4 to compute approximations for the harmonic function $u(x, y)$ in the rectangle $R = \{(x, y): 0 \le x \le 1.5, 0 \le y \le 1.5\}$; use h = 0.5. The boundary values are

$$u(x, 0) = x^4 \text{ and } u(x, 1.5) = x^4 - 13.5x^2 + 5.0625 \quad \text{for } 0 \le x \le 1.5,$$

$$u(0, y) = y^4 \text{ and } u(1.5, y) = y^4 - 13.5y^2 + 5.0625 \quad \text{for } 0 \le y \le 1.5.$$

Solution:

This question is relatively easy to solve. The results are shown below.



The figure at left-top corner is the surf of the approximation result and the figure at right-top corner is the surf of the exact solution. The two figures are almost the same about the shapes, which means our computation may be very correct.

The figure at left-bottom corner is the surf of the approximation errors and the right one is its corresponding contour. We can see from the figures that our error is still huge and the maximum error is about 1.6.