第六次作业

李阳 515072910019 黄弘毅 515072910038 王潇卫 515072910032

1

Before we solve this question, let's define to functions:

TrapComp.m

```
function s=TrapComp(f,a,b,M)

%Input      - f is the integrand
%               - a and b are upper and lower limits of integration
%               - M is the number of subintervals
%Output   - s is the trapezoidal rule sum

h=(b-a)/M;
s=0;

for k=1:(M-1)
   x=a+h*k;
   s=s+f(x);
end

s=h*(f(a)+f(b))/2+h*s;
```

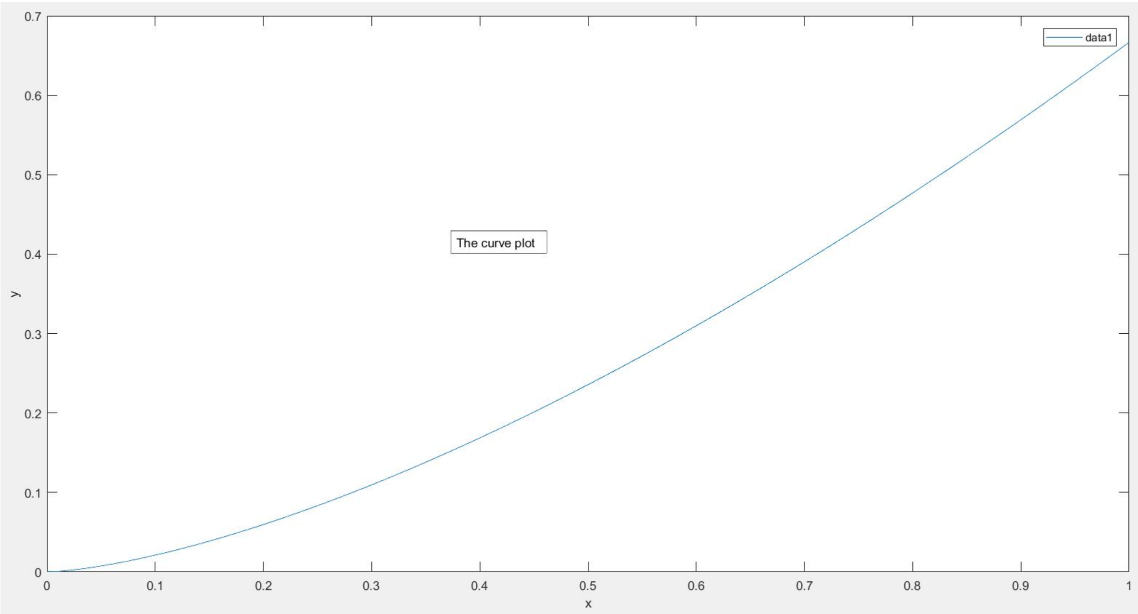
Simpson1_3.m

```
function I = Simpson1_3(f,a,b,n)
%
% Simpson estimates the value of the integral of f(x)
% from a to b by using the composite Simpson’s 1/3 rule
% applied to n equal-length subintervals.
%
% I = Simpson1_3(f,a,b,n) where
%
% f is an inline function representing the integrand,
% a, b are the limits of integration,
% n is the (even) number of subintervals,
%
% I is the integral estimate.
```

```
for i = 1:2:n
    I = I + f(x(i)) + 4*f(x(i+1)) + f(x(i+2));
end
I = I*h/3;
```

（a）

We choose curve: $f(x) = \frac{2x^{2/3}}{3}, 0 \leq x \leq 1$



the curve length $L = \int_0^1 \sqrt{1+x}dx$ Using trapezoid rule with three points: $h = 1/2$
$L = \frac{h}{2}[f(0) + 2f(1/2) + f(1)] = 1.215925826289068$ Using Simpson rule:
$L = \frac{h}{3}[f(0) + 4f(1/2) + f(1)] = 1.218865507989908$

Code：

```
%% Question 1 (a):
clear all; clc;
format long;
f = inline('sqrt(1+x)');
L1 = (1/4)*(f(0)+2*f(1/2)+f(1))
L2 = (1/6)*(f(0)+4*f(1/2)+f(1))
```

(b)

The theoretical length of arc is $L = \int_0^1 \sqrt{1+x}dx = \frac{2}{3}(2^{3/2} - 1) = 1.218951416497460$

Code：

```
%% Question 1 (b):
L=1.218951416497460;
R1 = (L1-L)/L;
```

$R1 = -0.002482125347609$ $R2 = 7.047738440501107e-05$ So using Simpson rule is more accurate to evaluate the curve length than trapezoid rule.

(c)

Code:

```
%% Question 1 (c):
format long;
clear all;clc;
f = inline('sqrt(1+x)');
L=1.218951416497460;
n = [2 10 20 40 80 160];
epsit = zeros(1,6); epsis = zeros(1,6);
for i=1:6
    epsit(i) = abs((TrapComp(f,0,1,n(i))-L))/L;
    epsis(i) = abs((Simpson1_3(f,0,1,n(i))-L))/L;
end
% We make the result diagram
result = zeros(6,3);
for i=1:6
    result(i,1) = n(i);
    result(i,2) = epsit(i);
    result(i,3) = epsis(i);
end
```

Result:

1	2	3
2	0.0025	7.0477e-05
10	1.0008e-04	1.3904e-07
20	2.5027e-05	8.7673e-09
40	6.2572e-06	5.4919e-10
80	1.5643e-06	3.4343e-11
160	3.9108e-07	2.1462e-12

Column 1： n Column 2: ε_T Column 3: ε_S

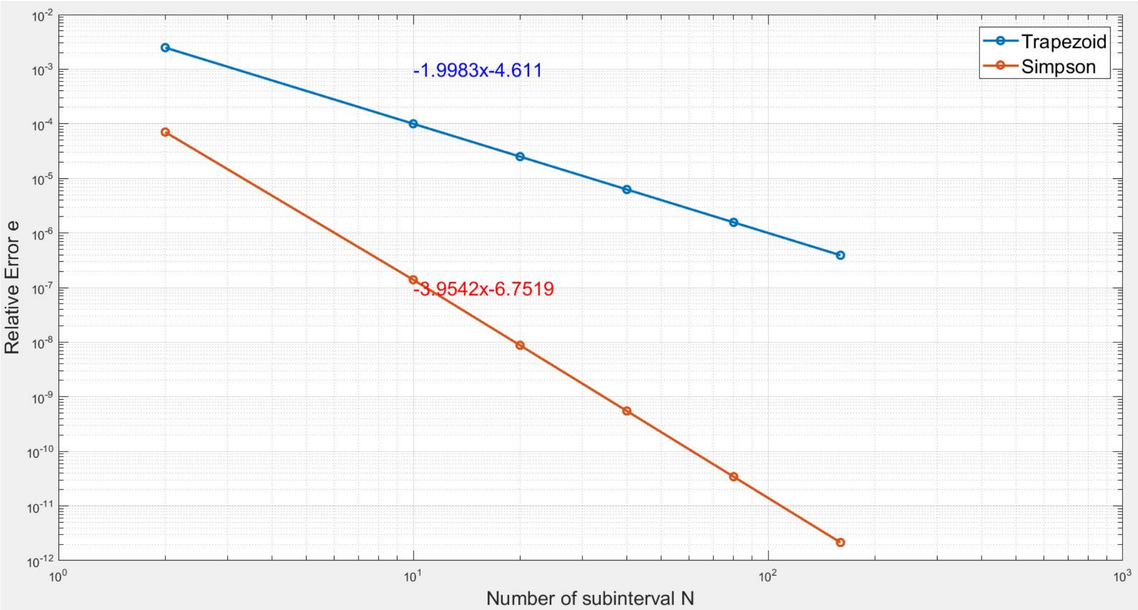
(d)

Code:

```
%% Question 1 (d):
result(:,1)=[];
loglog(n, result,'-o','LineWidth',2);
p1=polyfit(log(n),log(epsit),1);
```

```
axis([1 1e+3 1e-12 1e-2]);
text(10,1e-3,[num2str(p1(1)) 'x' num2str(p1(2))],'FontSize',16,'Color',
[0 0 1]);
text(10,1e-7,[num2str(p2(1)) 'x' num2str(p2(2))],'FontSize',16,'Color',
[1 0 0]);
legend1=legend('Trapezoid','Simpson');
set(legend1,'FontSize',16)
grid on;
```

Result:



（e）

Frow the plot, we know that the the power-law dependence is for Traoezoid: $\varepsilon_T = -1.9983N^{-4.611}$ for Simpson: $\varepsilon_S = -3.9542N^{-6.7519}$ Because the ordinate on log-log plot will be the negative of the number of decimal places of precision in calculation, So the Calculation precision is as follows:

Precision means the number of decimals in calculation		
N	**Precision for Trap**	**Precision for Simpson**
2	3	5
10	4	7
20	5	9
40	6	10
80	6	11
160	7	12

2

$$f(x)=\frac{(x-(a+h))(x-(a+2h))(x-(a+3h))}{(-h)(-2h)(-3h)}f_0+\frac{(x-a)(x-(a+2h))(x-(a+3h))}{(h)(-h)(-2h)}f_1+\frac{(x-a)(x-(a+h))(x-(a+3h))}{(2h)(h)(-h)}f_2$$
$$+\frac{(x-a)(x-(a+h))(x-(a+2h))}{(3h)(2h)(3h)}f_3+\frac{(x-a)(x-(a+h))(x-(a+2h))(x-(a+3h))}{4!}f^{(4)}(\xi)$$

Where ξ is some value between a and b

Integrate both sides

$\mathbf{Example:}$ The way we integrate $(x-(a+h))(x-(a+2h))(x-(a+3h))$ Let $x=\frac{3h}{2}(t+1)+a$,then integral becomes $\int_{-1}^{1}\frac{3}{2}h(\frac{h}{2})^3(3t+1)(3t-1)(3t-3)\mathrm{d}t=-\frac{9}{4}h^4$ divide by (-h)(-2h)(-3h),we get coefficient $\frac{1}{8}$

integrate all 4 terms, we derive $\int_a^b f(x)\mathrm{d}x=\frac{3h}{8}(f_0+3f_1+3f_2+f_3)-\frac{3}{80}h^5f^{(4)}(c)$

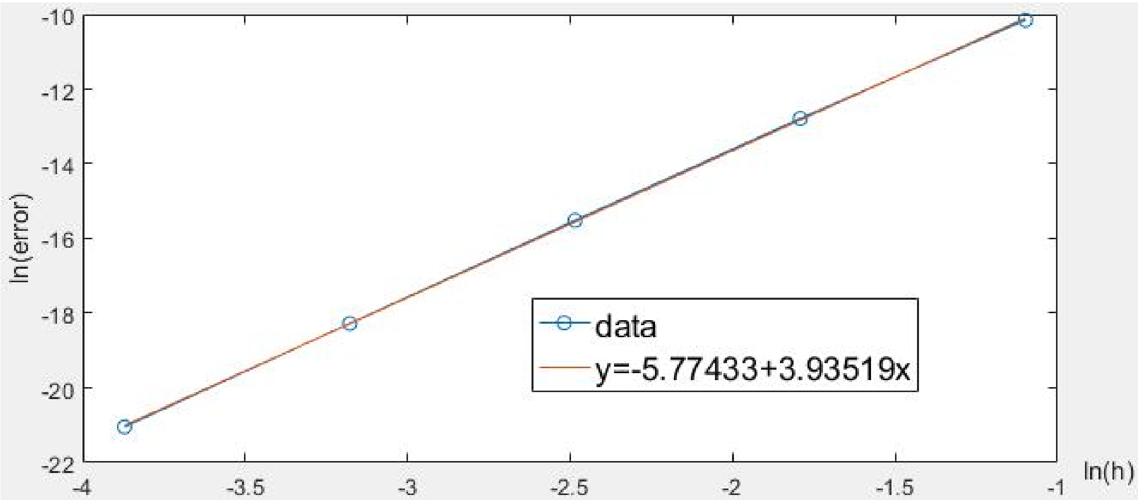
Note: in the first equation,$f^{(4)}(\xi)$should depend on x you choose, but in the integral we just treat it as a constant

(b)

We use integral $\int_0^1\sqrt{1+x}\mathrm{d}x$ The exact value of the integral is: $\frac{2}{3}(2^{1.5}-1)=1.218951416497460$ Then we apply Simpson's $\frac{3}{8}$ rule, where N = 1,2,4,8,16, we get

Exact value	1.218951416497460		
N		error	Relative error (e-04)
1	1.218912315464783	3.910103267634746e-05	0.320775973079391
2	1.218948613626460	2.802870999962792e-06	0.022994115778761
4	1.218951233411024	1.830864355678585e-07	0.001501999448788
8	1.218951404914901	1.158255891198223e-08	0.000095020677241
16	1.218951415771289	7.261711232331436e-10	0.000005957342626

We plot ln(error) against ln(h)



Which shows error is proportional to h^4

Simpson3_8.m:

```
function I=Simpson3_8(f,N,a,b)
% ***here 3N intervals totally***
```

```
%a,b:integral range
%n: how many segments [must be even]
x=linspace(a,b,3*N+1);
h=x(2)-x(1);
y=f(x);
w=9*h/8*ones(1,3*N+1);
w(1:3:end)=6*h/8;
w(1)=3*h/8;
w(end)=3*h/8;
I=sum(y.*w);
```

Script:

```
ee=[3.910103267634746e-05,2.802870999962792e-06,1.830864355678585e-
07,1.158255891198223e-08,7.261711232331436e-10];
N=[1,2,4,8,16];
h=1./N/3;
ee=log(ee);
h=log(h);
[a0,a1]=Linearregression(h,ee,ones(1,5));
a0
a1
x=linspace(h(1),h(end),100);
y=a0+a1*x;
plot(h,ee,'-o');
hold on
plot(x,y);
```

Linear regression:

```
function [a0,a1]=Linearregression(X,Y,U)
%Linear regression, round-off error susceptible
S=sum(1./U./U);
Sx=sum(X./U./U);
Sy=sum(Y./U./U);
t=(X-Sx/S)./U;
Stt=sum(t.*t);
a1=sum(t.*Y./U)/Stt;
a0=(Sy-Sx*a1)/S;
u2a0=(1+Sx*Sx/S/Stt)/S;
u2a1=1/Stt;
Cov=-Sx/S/Stt;
r=Cov/sqrt(u2a0)/sqrt(u2a1);
Y1=a0+a1*X;
Chi2=sum((Y-Y1).*(Y-Y1)./U./U);
[a0,sqrt(u2a0),a1,sqrt(u2a1),Chi2,Cov,r];
```

From the results of (b), we can see the required number N (when relative error is less than 5e-09) is somewhere between 8 and 16 And we just calculate all relative errors from N=8 to N=16, we find when N>=10, relative error begins to be lower than 5e-09

N	Relative error
8	9.502067724129323e-09
9	5.937251615421968e-09
10	3.897859676847526e-09

Script:

```
exact=2/3*(power(2,3/2)-1);
f=@(x) sqrt(1+x);
for N=8:16
I=Simpson3_8(f,N,0,1);
(I-exact)/exact
end
```

3

(a)

We get the four roots are

$$x_3 = \sqrt{\frac{15-2\sqrt{30}}{35}}, x_2 = -\sqrt{\frac{15-2\sqrt{30}}{35}}$$

$$x_4 = \sqrt{\frac{15+2\sqrt{30}}{35}}, x_1 = -\sqrt{\frac{15+2\sqrt{30}}{35}}$$

And four weights are

$$w_1 = w_4 = \frac{3\sqrt{30}-5}{6\sqrt{30}}$$

$$w_2 = w_3 = \frac{3\sqrt{30}+5}{6\sqrt{30}}$$

(b)

Using the substitution $x = \frac{b-a}{2}(t+1)+a$, we could convert the integral to $\int_a^b f(x)\mathrm{d}x = \int_{-1}^{1} f(\frac{b-a}{2}(t+1)+a)\frac{b-a}{2}\mathrm{d}t$ And just sum up the function values of 4 abscissas multiplied by respected weights

Gauss_quad_4.m

```
% exact when polynomial's order <=7
% f: handle
% [a,b] interval
x=zeros(1,4);
w=zeros(1,4);
x(1)=-sqrt((15+2*sqrt(30))/35);
x(2)=-sqrt((15-2*sqrt(30))/35);
x(3)=-x(2);
x(4)=-x(1);
w(1)=(3*sqrt(30)-5)/6/sqrt(30);
w(4)=w(1);
w(2)=(3*sqrt(30)+5)/6/sqrt(30);
w(3)=w(2);
I=sum(f((b-a)/2*(x+1)+a)*(b-a)/2.*w);
```

(c)

```
>> format long
>> I=Gauss_quad_4(f,0,1)

I =

   1.218951433509519
```

Exact value: 1.218951416497460 It has attained the precision of N=8 in (b)'s 3/8 Simpson's rule, which use 25 points, here only 4 point's is required.

4

(a)

Using Simpson's $\frac{1}{3}$rule for N=4, the code has given in question 1 and we get the integral is $\mathbf{0.062168726647009}$.

Simpon1_3.m

```
function I = Simpson1_3(f,a,b,N)
%
% Simpson estimates the value of the integral of f(x)
% from a to b by using the composite Simpson's 1/3 rule
% applied to N equal-length subintervals.
%
% I = Simpson1_3(f,a,b,N) where
%
% f is a handle of function   eg:f=@sin, f=@(x) sin(x)
% a, b are the limits of integration,
% n is the (even) number of subintervals.
```

```
assert(mod(N,2)==0, 'N(subintervals) must be even.');

h = (b-a)/N;
x = a:h:b;

I = 0;
for i = 1:2:(N-1)
    I = I + f(x(i)) + 4*f(x(i+1)) + f(x(i+2));
end

I = I*h/3;
```

(b)

(b)

Using Gauss_Quad_4 we get the integral is $\mathbf{0.062204707404342}$. The code of Gauss_Quad_4 has given in question 3(b).

(c)

Using Romberg with N = 2 and $N_{levels} = 3$we get the integral is $\mathbf{0.062204326376580}$. When using Romberg method, we get the first line of matrix T through Composite Trapezoid Rule and the code has given in question 1. Action: The following code Romberg.m is not the one given by teacher, so when testing this problem, the code should be copy.

TrapComp.m

```
function s=TrapComp(f,a,b,M)

%Input      - f is the integrand
%           - a and b are upper and lower limits of integration
%           - M is the number of subintervals
%Output   - s is the trapezoidal rule sum

h=(b-a)/M;
s=0;

for k=1:(M-1)
   x=a+h*k;
   s=s+f(x);
end

s=h*(f(a)+f(b))/2+h*s;
```

```
% Romberg method
% [a,b]: interval
% f: handle of integrand
% N_subintervals: subintervals(even)
% N_levels: Levels of accuracy
% T: the matrix records estimations of each step
% I: the integral of appointed levels of accuracy

assert(mod(N_subintervals,2)==0, 'N_subintervals must be even.');
T = zeros(N_levels);
for j = 1:N_levels
    T(j,1) = TrapComp(f,a,b,N_subintervals*2^(j-1)); % Using Trapezoid
Rule gets the first line
end

% Romberg method
for k = 2:N_levels
    for j = 1:(N_levels - k + 1)
        T(j,k) = (4^(k-1)*T(j+1,k-1) - T(j,k-1))/(4^(k-1)-1);
    end
end

I = T(1,N_levels);
```

(d)

(d)

actual value : $\mathbf{0.062204682443299}$ and the relative errors(%) for (a),(b),(c) separately are:

```
re_errors =
  -0.057802395057962   0.000040127272705  -0.000572411440343
```

$\textbf{Comment :}$ We can see that Gauss_Quad_4 has the smallest relative error, Simpson1/3 Rule has the biggest one. And only Gauss_Quad_4 is positive error.

Script for the whole problem:

```
%第六次作业 第4题
clear all
clc

f = @(x) exp(-x).*sin(x)./(x.^3+1);
a = 1;
b = 2;
```

```
I_Simpson1_3 = Simpson1_3(f,a,b,4)

%% (b)
I_Gauss_quad_4 = Gauss_quad_4(f,a,b)

%% (c)
[I_Romberg_N2_levels3,T] = Romberg(a,b,f,2,3)

%% (d)
acc_val = 0.062204682443299
re_errors = zeros(1,3);
re_errors(1,1) = 100*(I_Simpson1_3 - acc_val)/acc_val;
re_errors(1,2) = 100*(I_Gauss_quad_4 - acc_val)/acc_val;
re_errors(1,3) = 100*(I_Romberg_N2_levels3 - acc_val)/acc_val;

re_errors
```