

**矩阵与数值分析**

数值实验

|  |  |
| --- | --- |
| 院系名称： | 运载工程与力学学部 |
| 专业班级： | Good |
| 学号： | Something |
| 姓名： | Max |

大连理工大学

Dalian University of Technology

目录

[第一题 2](#_Toc28426)

[第二题： 7](#_Toc9388)

[第三题： 9](#_Toc1509)

[第四题： 15](#_Toc7439)

[第5题： 18](#_Toc9090)

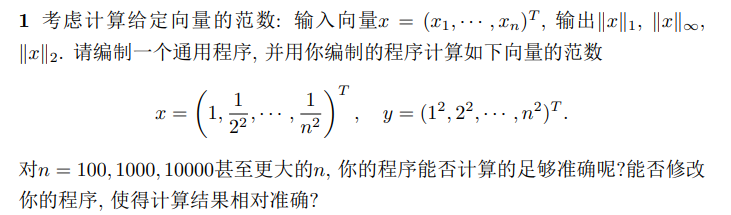
[第六题 23](#_Toc30465)

[第七题： 31](#_Toc10224)

[第八题 33](#_Toc13075)

[第九题： 36](#_Toc5957)

## 第一题



**计算范数的函数：**

function [x1, x8, x2] = fanshu(Y)

% 输入：Y 是输入的向量

% 输出：x1 是一范数，x2 是二范数，x8 无穷范数

x1 = 0;

x2 = 0;

x8 = 0;

% 算一范数

for i = 1:1:numel(Y)

x1 = x1 + abs(Y(i));

end

% 算二范数

for i = 1:1:numel(Y)

x2 = x2 + Y(i)^2;

end

x2 = sqrt(x2);

% 算无穷范数

for i = 1:1:numel(Y)

if(abs(Y(i)) > x8)

x8 = Y(i);

end

end

end

**在主函数中运行：**

clc

clear

% for i = 1:1:100

% Y(i) = i^2;

% end

for i = 1:1:100

Y(i) = 1/i^2;

end

[x1, x8, x2] = fanshu(Y);

fprintf('一范数x1 = %f\n',x1);

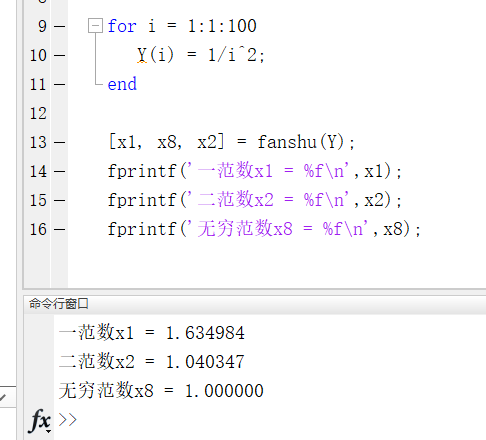
fprintf('二范数x2 = %f\n',x2);

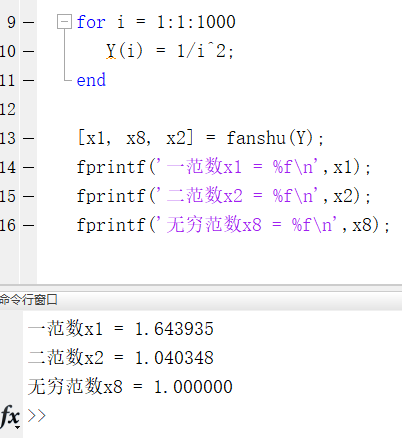
fprintf('无穷范数x8 = %f\n',x8);

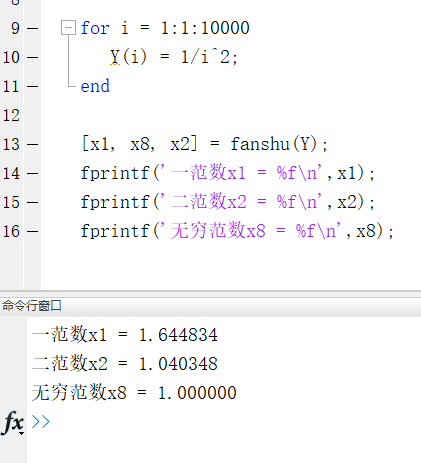
**运行结果：**

**x向量：**

**N = 100，1000，10000时的计算结果：**

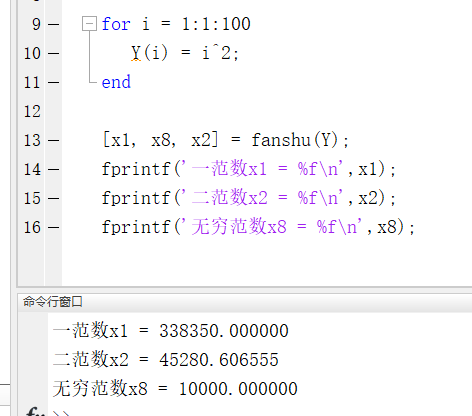


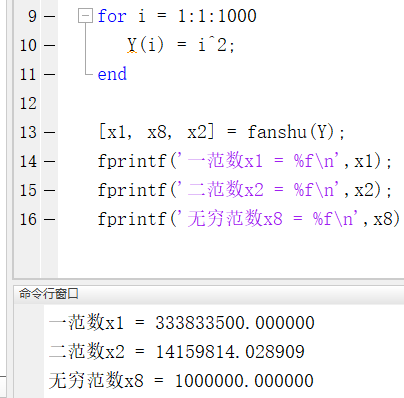


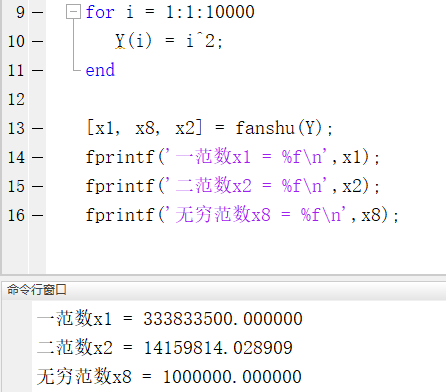


**y向量：**

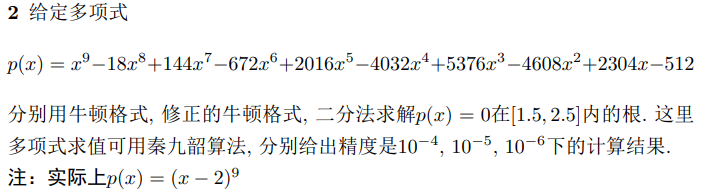
**N = 100，1000，10000：**







## 第二题：



MATLAB代码：

function [] = qiugen( kesai )

flag=1;

x0=2.1;

y0=2.1;

while(flag)

x1=x0-(x0-2)^9/(9\*(x0-2)^8);

if (abs(x1-x0)<kesai)

fprintf('牛顿法的结果为%f\n',x1);

break;

end

x0=x1;

end

while(flag)

y1=y0-8\*(y0-2)^9/(9\*(y0-2)^8);

if (abs(y1-y0)<kesai)

fprintf('修改牛顿法的结果为%f\n',y1);

break;

end

y0=y1;

end

end

%二分法

function []=erfen(kesai)

a=1.5;

b=2.5;

z0=(a+b)/2;

c=a;

d=b;

fun=@(x)(x-2)^9;

while(abs(c-d)>kesai)

if feval(fun,z0)\*feval(fun,c)>0

c=z0;

else

d=z0;

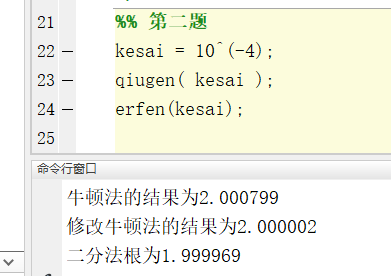
end

z0=(c+d)/2;

end

fprintf('二分法根为%f\n',z0);

end



**精度为1e-5**：

牛顿法的结果为2.000076

修改牛顿法的结果为2.000000

二分法根为1.999996

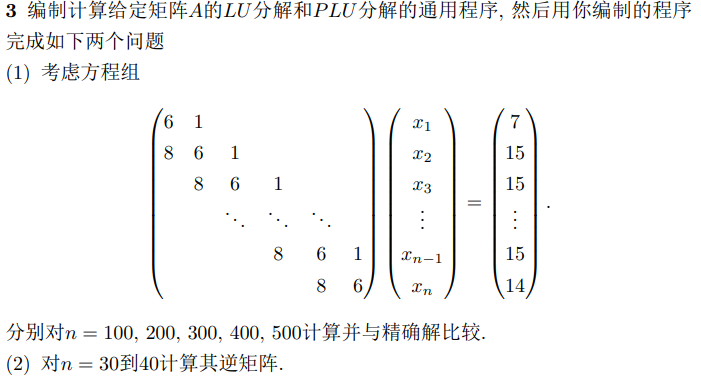
**精度为1e-6时**：

牛顿法的结果为2.000007

修改牛顿法的结果为2.000000

二分法根为2.000000

## 第三题：



Matlab代码：

LU分解：

function x=lufenjie(a,b)

n=size(a,1);

m = eye(n,n);

% initial the n\*n matrix m zeros

x=zeros(n,1);

y=zeros(n,1);

for j = 1 : n-1

if abs(a(j,j))<eps;

error('zero pivot encountered');

end

for i = j+1 : n

mult = a(i,j)/a(j,j);

m(i,j) = mult;

for k = j:n

a(i,k) = a(i,k) - mult\*a(j,k);

end

end

y(1)=b(1);

for k=2:n

y(k)=b(k)-m(k,1:k-1)\*y(1:k-1);

end

x(n)=y(n)/a(n,n);

for k=n-1:-1:1

x(k)=(y(k)-a(k,k+1:n)\*x(k+1:n))/a(k,k);

end

end

PLU分解：

function [x] = zhuyuanfenjie(A,b)

n=size(A,1);

x=zeros(n,1);

y=zeros(n,1);

temprow=zeros(n,1);

tempconstant=0;

%p=zeros(n,1);

for i=1:n-1

[max\_value,index]=max(abs(A(i:n,i)));

temprow=A(i,:);

A(i,:)=A(i+index-1,:);

A(i+index-1,:)=temprow;

tempconstant=b(i);

b(i)=b(index+i-1);

b(index+i-1)=tempconstant;

if A(i,i)==0

disp('矩阵A为非奇异');

end

for j=i+1;n

miu=A(j,i)/A(i,i);

A(j,i)=miu;

A(j,i+1:n)=A(j,i+1:n)-miu\*A(i,i+1:n);

end

end

y(1)=b(1);

for k=2:n

y(k)=b(k)-A(k,1:k-1)\*y(1:k-1);

end

x(n)=y(n)/A(n,n);

for k=n-1:-1:1

x(k)=(y(k)-A(k,k+1:n)\*x(k+1:n))/A(k,k);

end

主函数运行程序：

%% 第三题

n = 100;

A = zeros(n,n);

for i = 1:1:n

A(i,i) = 6;

try

A(i,i-1) = 8;

catch

end

end

for i = 1:1:n-1

A(i,i+1) = 1;

end

B = zeros(n,1);

B(1) = 7;

B(100) = 14;

for i = 2:1:n-1

B(i) = 15;

end

x = lufenjie(A,B);

x1 = zhuyuanfenjie(A,B);

[c] = nijuzhen(40);

**运行结果：**

**N = 20时**

LU、PLU分解结果为：

[1,1,1,1……………………….1]T

然而当N取值增大时，求解结果会趋于发散。

**逆矩阵：**

function [c] = nijuzhen(n)

temp=eye(n);

c=zeros(n,n);

a=zeros(n,n);

a(1,1)=6;

a(1,2)=1;

a(n,n)=6;

a(n,n-1)=8;

for i=2:n-1

a(i,i-1)=8;

a(i,i)=6;

a(i,i+1)=1;

end

for i=1:n

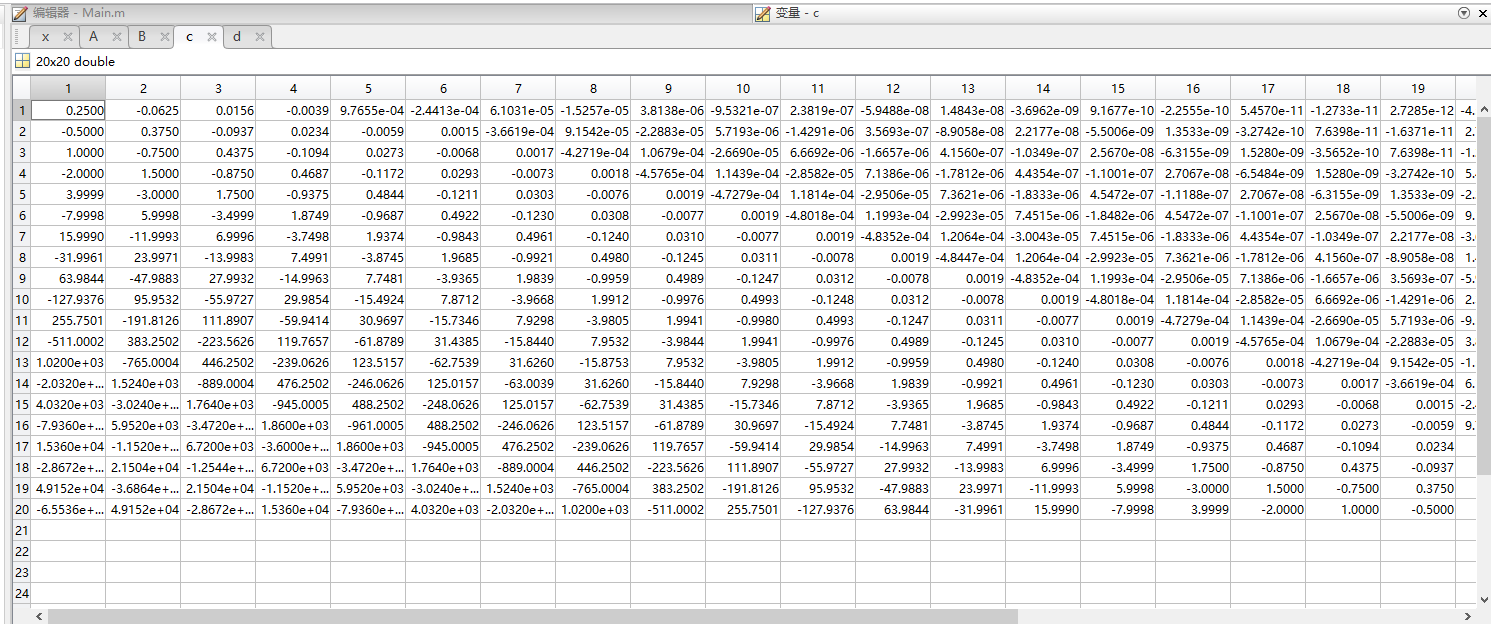
c(:,i)=lufenjie(a,temp(:,i));

end

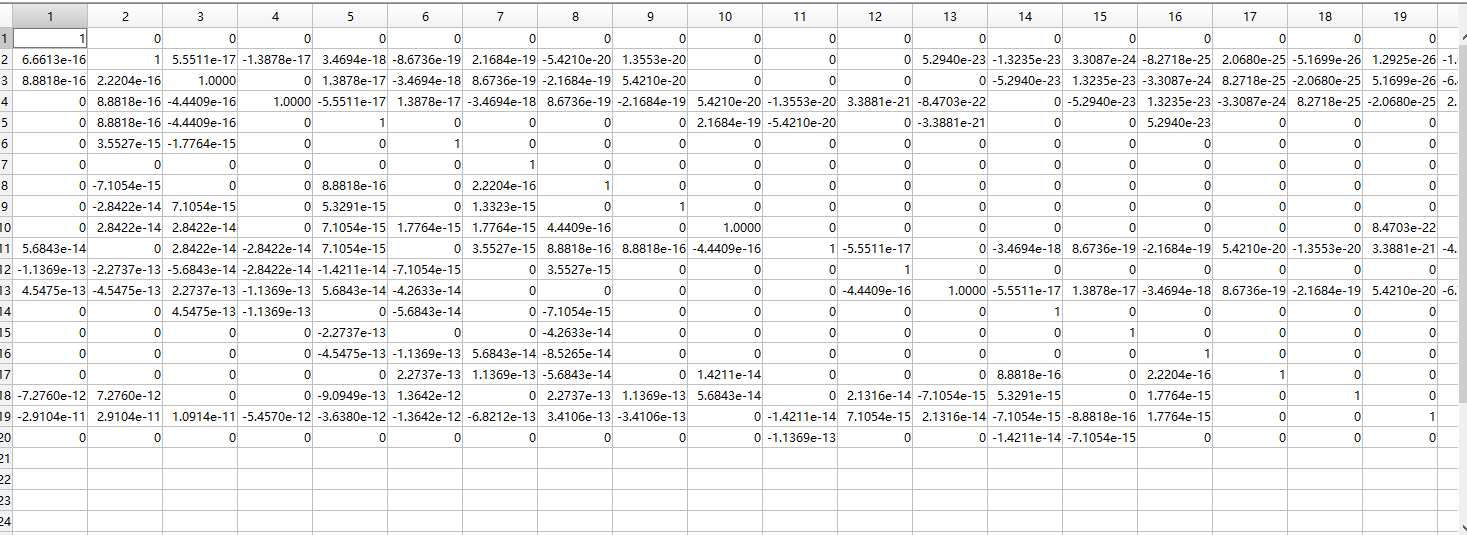
end

N = 20时：

逆矩阵为：

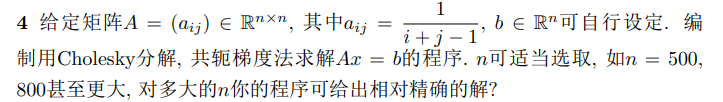


**验证得：**



结果基本正确。

## 第四题：



**Matlab代码：**

function [x] = choleskyfenjie(a,b)

n=size(a,1);

y=zeros(n,1);

x=zeros(n,1);

for k=1:n

a(k,k)=sqrt(a(k,k));

a(k+1:n,k)=a(k+1:n,k)/a(k,k);

for j=k+1:n

a(j:n,j)=a(j:n,j)-a(j:n,k)\*a(j,k);

end

end

y(1)=b(1)/a(1,1);

for i=2:n

y(i)=(b(i)-a(i,1:i-1)\*y(1:i-1))/a(i,i);

end

x(n)=y(n)/a(n,n);

for i=n-1:-1:1

x(i)=(y(i)-a(i+1:n,i)'\*x(i+1:n))/a(i,i);

end

end

function [x] = gongetidu(a,b)

n=size(a,1);

x=zeros(n,1);

r=b-a\*x;

p=r;

for k=0:n-1

alpha=(r'\*r)/(p'\*a\*p);

x=x+alpha\*p;

r2=b-a\*x;

if((norm(r2)<=1e-3)||(k==n-1))

break;

end

beta=norm(r2)^2/norm(r)^2;

p=r2+beta\*p;

r=r2;

end

end

function [x1,x2]=matrixa(n)

a=zeros(n);

for i=1:n

for j=1:i

a(i,j)=1/(i+j-1);

a(j,i)=a(i,j);

end

end

b=ones(n,1);

x1=gongetidu(a,b);

fprintf('chol分解的误差为：%f\n',norm(a\*x1-b));

x2=choleskyfenjie(a,b);

fprintf('共轭梯度的误差为：%f\n',norm(a\*x2-b));

end

运行结果：

**N=500,**

chol分解的误差为：0.000705

共轭梯度的误差为：2543604465798403000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000.000000

**N=800,**

chol分解的误差为：0.000845

共轭梯度的误差为：194940788397572700000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000

## 第五题：



**Matlab 代码：**

拉格朗日插值：

function yh = lagrange(x,y,xh)

% xh 是要估计的点

n = length(x);

m = length(xh);

x = x(:);

y = y(:);

xh = xh(:);

yh = zeros(m,1);

c1 = ones(1,n-1);

c2 = ones(m,1);

for i = 1:n

xp = x([1:i-1 i+1:n]);

yh = yh + y(i) \* prod((xh\*c1-c2\*xp')./(c2\*(x(i)\*c1-xp')),2);

end

**牛顿插值代码：**

function yi = NewtonInterp(x,y,xi)

% Newton 基本插值公式，调用格式为

% yi = NewtonInterp(x,y,xi)

% 其中

% x 为插值节点，y为节点处函数值，

% xi 为为估计函数自变量，yi 为xi处函数估计值

%

%计算差商表

n = length(x); Y= zeros(n); Y(:,1) = y';

for k = 1 : n-1

for i = 1 : n - k

if abs(x(i+k) - x(i))<eps

error('% 输入的插值节点必须互异！');

end

Y(i,k+1) = (Y(i+1,k) - Y(i,k))/(x(i+k) - x(i));

end

end

m = length(xi); yi = zeros(1,m);

for i = 1 : n % 计算Newton插值公式

z = ones(1,m);

for k = 1 : i - 1

z = z .\*(xi - x(k));

end

yi = yi + Y(1,i) \* z;

End

运行主函数：

%% 第五题

% sin(pi\*x)曲线

x = 0:0.0000001:1;

y = sin(pi \* x);

plot(x,y,'r-');

hold on

% 拉格朗日插值的曲线

x = 0:0.01:1;

y = sin(pi \* x);

yh = lagrange(x,y,x); % 拉格朗日插值函数

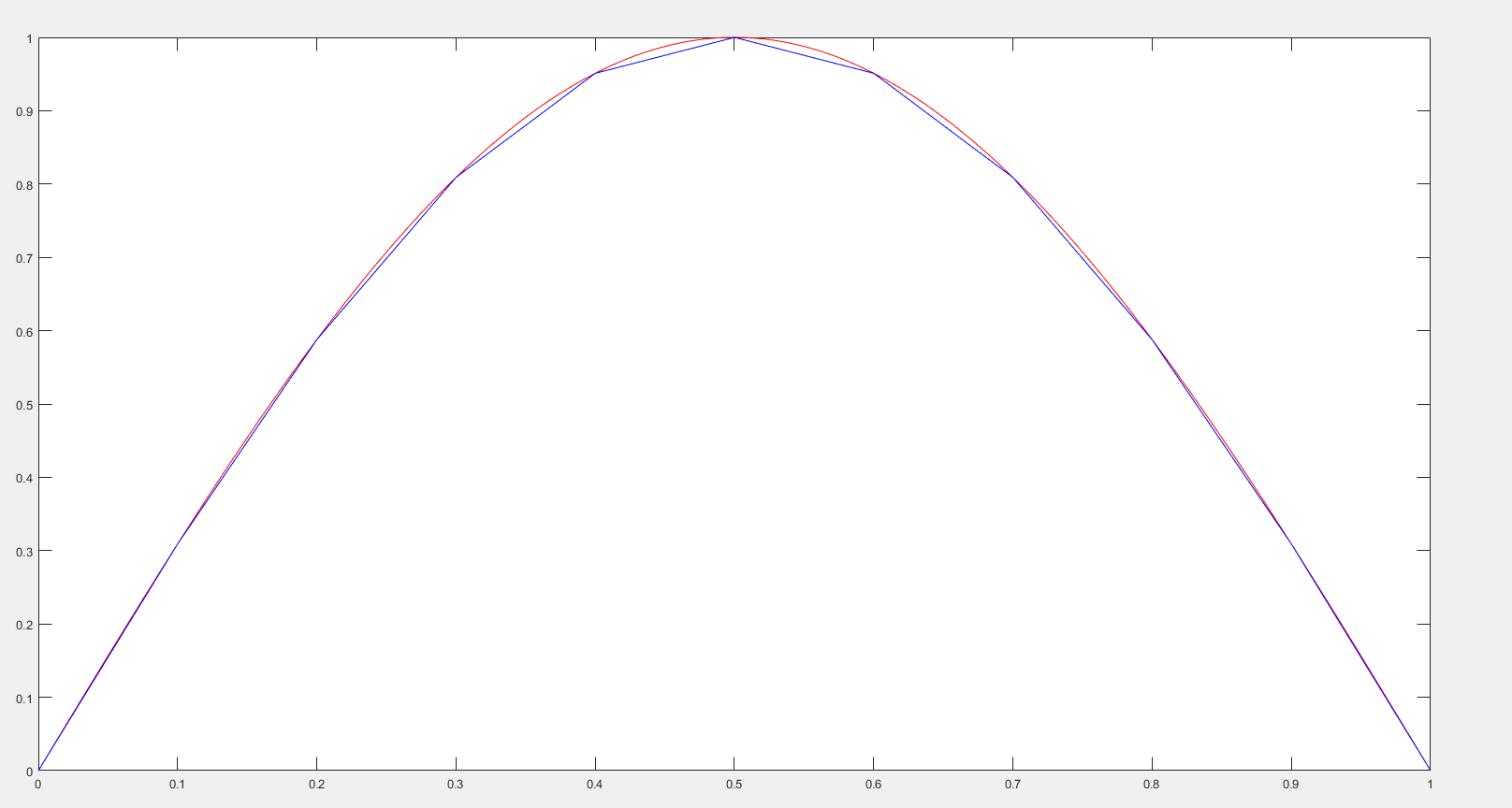
% plot(x,yh,'b-');

% hold on

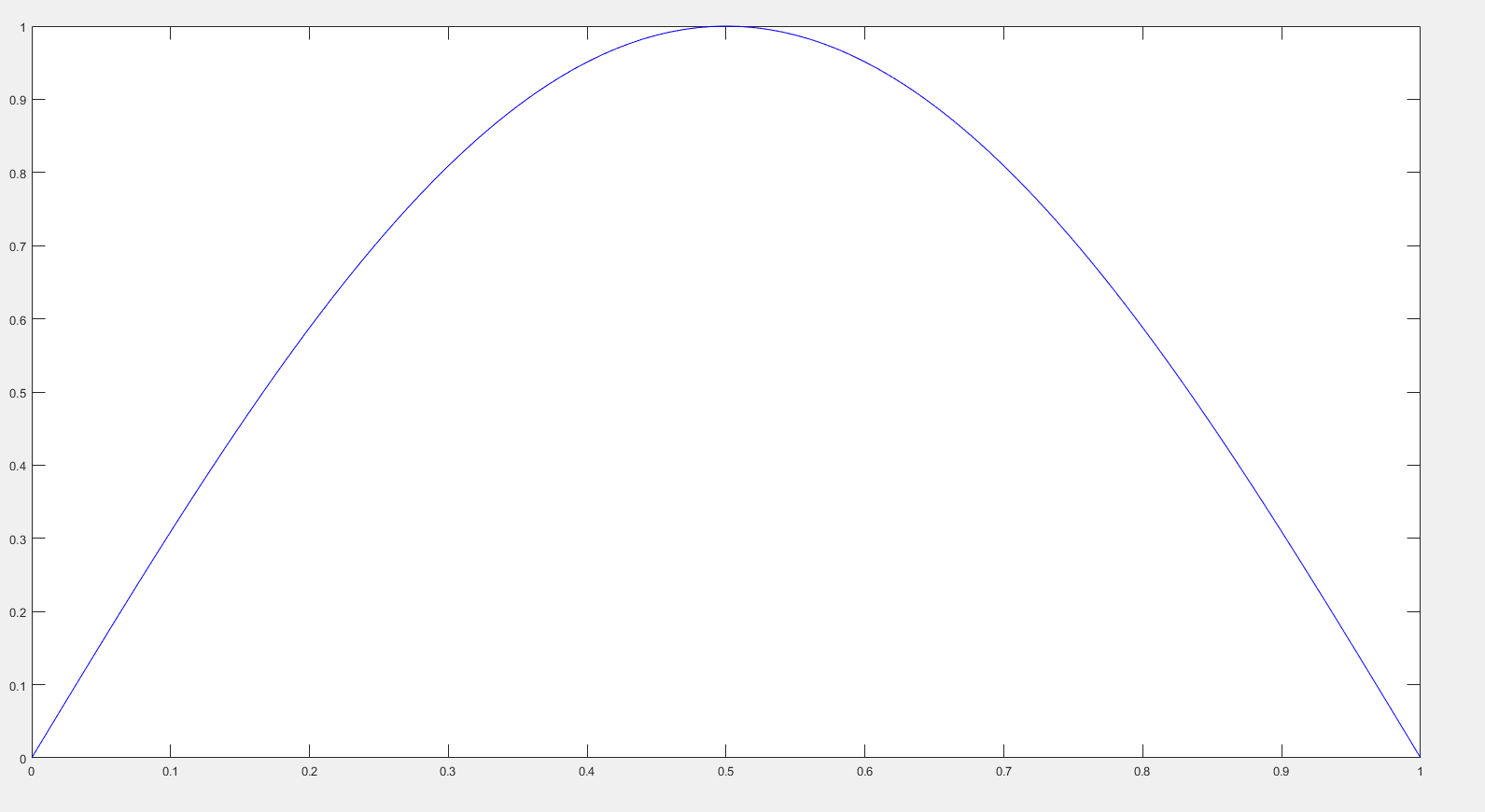
yi = NewtonInterp(x,y,x);

plot(x,yh,'g-');

拉格朗日插值结果，间隔0.1：



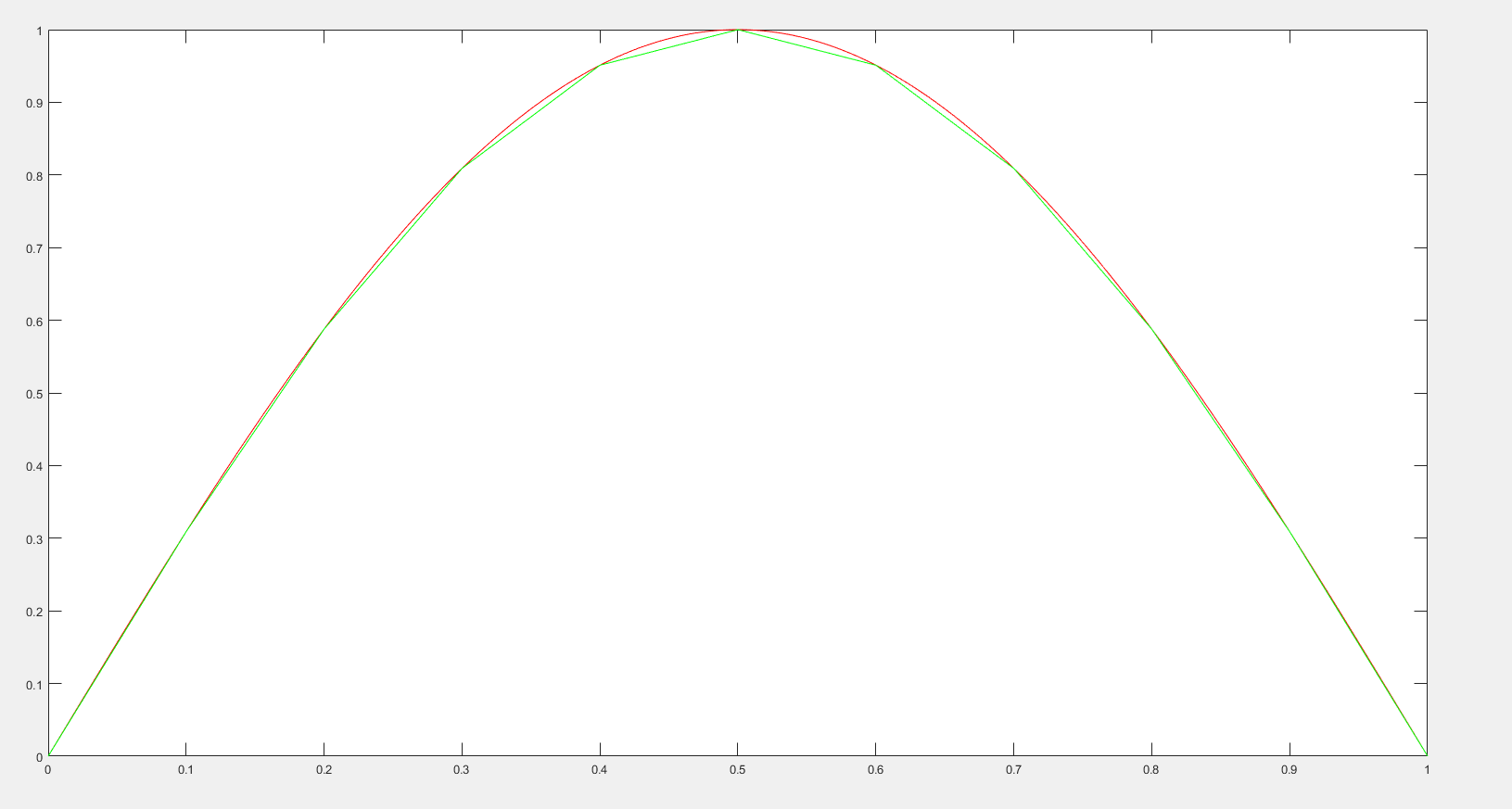
间隔0.01：



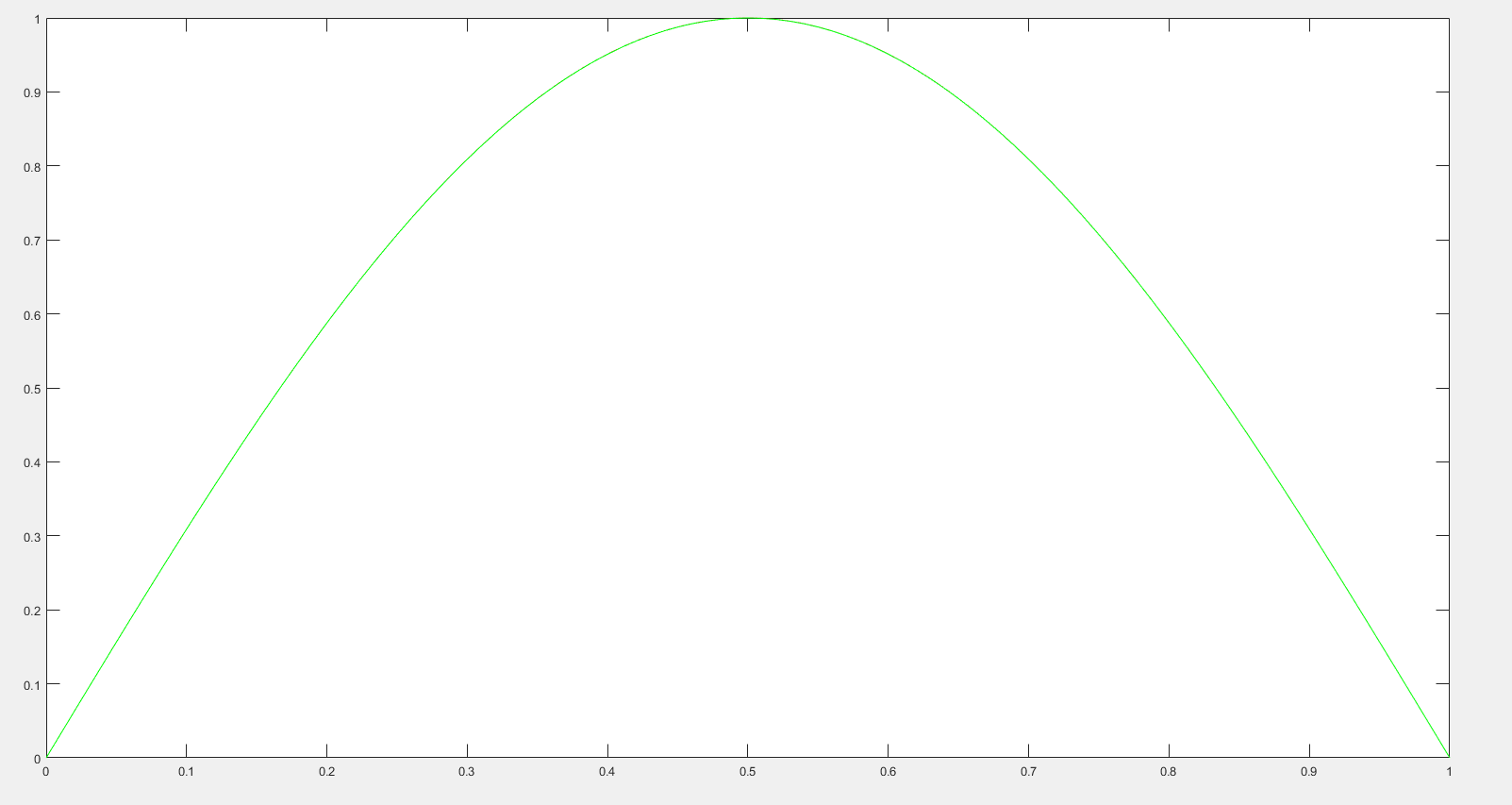
结果很好。

牛顿插值结果：

间隔0.1：

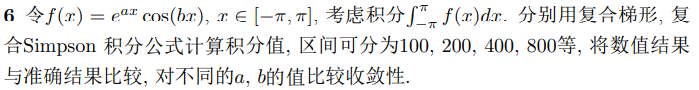


间隔0.01：



结果很好。

## 第六题



**复合梯形：**

Matlab代码：

function res = ComTrapezium(f,n,a,b)

format long;

if b < a

c = b;

b = a;

a = c;

end

h = (b-a)/n;

d = f(a);

for i = a+h:h:b-h

d = d + (2 \* f(i));

end

d = d + f(b);

res = (d \* h / 2);

end

**运行主函数代码：**

%% 第六题

% 求函数精确积分

syms x;

jingque = int(exp(1 \* x) \* cos(1 \* x), -pi, pi);

disp('精确解为：');

fprintf('%f\n',jingque);

a = 1;

b = 1;

n = 100;

fx = inline('exp(1 \* x) \* cos(1 \* x)','x');

res = ComTrapezium(fx,n,-pi,pi); % 复化的梯形公式

m = abs(jingque - res)/abs(jingque);

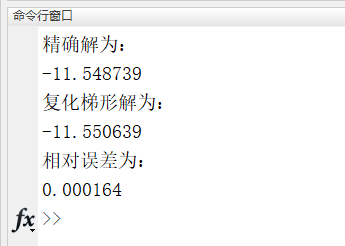
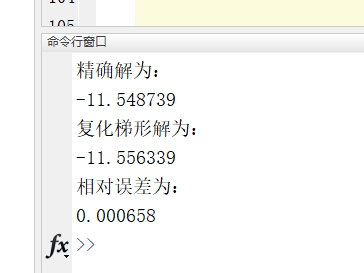
disp('复化梯形解为：');

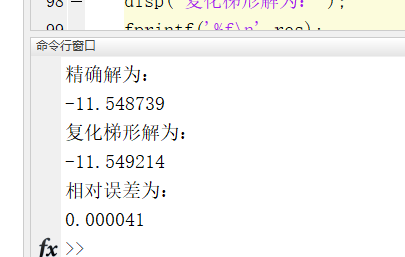
fprintf('%f\n',res);

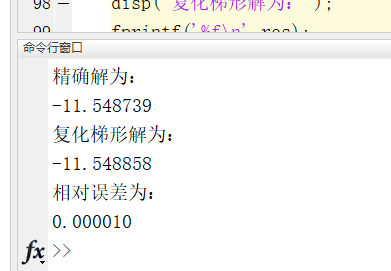
disp('相对误差为：');

fprintf('%f\n',m);

以下结果是N = 100，200，400，800的结果以及相对误差：







**复化Simpson公式：**

**Matlab 代码：**

%计算x对应的函数值f(x)

function [y\_x] = CalcuFunctionValue(x)

% inputs:

% x:待求值

% outputs:

% y\_x:x对应的函数值

y\_x = exp(1 \* x) \* cos(1 \* x);

end

function [result] = ComplexSimpson(x\_LowBound, x\_Up\_Bound,n)

% simpson求积公式

% Inputs:

% x\_LowBound:积分区间下界

% x\_UpBound :积分区间上界

% n ：等分数量,需要为2n等分，即节点个数必须满足2n+1

% Outputs:

% result : 复化Simpson积分结果

% 判断积分区间个数是否是2的倍数，满足则进行计算，否则打印提示

if mod(n,2) == 0

% 获取步长h

step\_length = (x\_Up\_Bound - x\_LowBound)/n;

%累积计算

result = 0;

for i = 1:2:n-1

result = result + CalcuFunctionValue(x\_LowBound+step\_length\*(i-1))...

+4\*CalcuFunctionValue(x\_LowBound+step\_length\*i)...

+CalcuFunctionValue(x\_LowBound+step\_length\*(i+1));

end % 循环结束

result = result \* step\_length / 3;

else

print('等分区间数错误！');

end % if判断结束

end % 函数结束

**运行主函数：**

% 求函数精确积分

syms x;

jingque = int(exp(1 \* x) \* cos(1 \* x), -pi, pi);

disp('精确解为：');

fprintf('%f\n',jingque);

a = 1;

b = 1;

n = 100;

fx = inline('exp(1 \* x) \* cos(1 \* x)','x');

res = ComTrapezium(fx,n,-pi,pi); % 复化的梯形公式

m = abs(jingque - res)/abs(jingque);

% disp('复化梯形解为：');

% fprintf('%f\n',res);

% disp('相对误差为：');

% fprintf('%f\n',m);

[result] = ComplexSimpson(-pi, pi,n);

mc = abs(jingque - result)/abs(jingque);

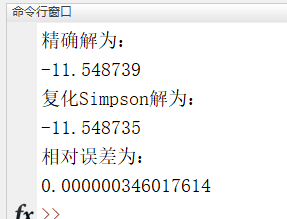
disp('复化Simpson解为：');

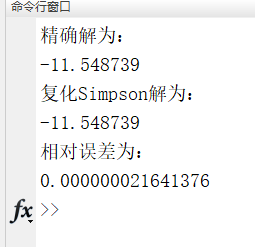
fprintf('%f\n',result);

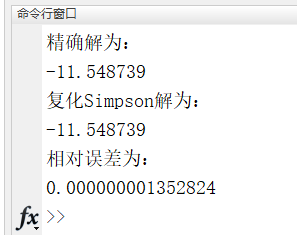
disp('相对误差为：');

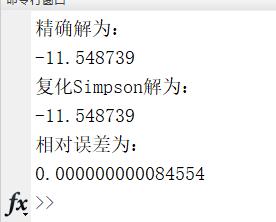
fprintf('%.15f\n',mc);

N = 100，200，400，800时的结果以及相对误差：

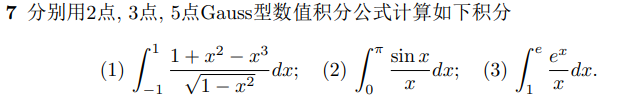








## 第七题：



MATLAB代码(以第三个为例，其余代码类似)：

function[quad1,quad2,quad3] = gaosishuzhijifen(f,a,b)

x1=[0.7745966692,0,-0.774596692];

w1=[0.5555555556,0.8888888889,0.5555555556];

T1=(a+b)/2+(b-a)/2\*x1;

quad1=(b-a)/2\*sum(w1.\*subs (f,T1));

%三点

x2=[0.8611363116,-0.8611363116,0.3399810436,-0.3399810436];

w2=[0.3478548451,0.3478548451,0.6521451549,0.6521451549];

T2=(a+b)/2+(b-a)/2\*x2;

quad2=(b-a)/2\*sum(w2.\*subs (f,T2));

%五点

x3=[0.9324695142,-0.9324695142,0.6612093865,-0.6612093865,0.2366191861,-0.2366191861];

w3=[0.2369268851,0.2369268851,0.4786286705,0.4786286705,0.5688888889,0.5688888889];

T3=(a+b)/2+(b-a)/2\*x3;

quad3=(b-a)/2\*sum(w3.\*subs (f,T3));

end

**运行主函数：**

syms x

f=exp(x)/x;

[quad1,quad2,quad3]=gaosishuzhijifen(f,1,exp(1));

fprintf('两点的结果为%f\n',quad1);

fprintf('三点的结果为%f\n',quad2);

fprintf('五点的结果为%f\n',quad3);

**运行结果**：

（1）

两点的结果为3.699803

三点的结果为3.930531

五点的结果为5.522704

（2）

两点的结果为1.851976

三点的结果为1.851937

五点的结果为2.370443

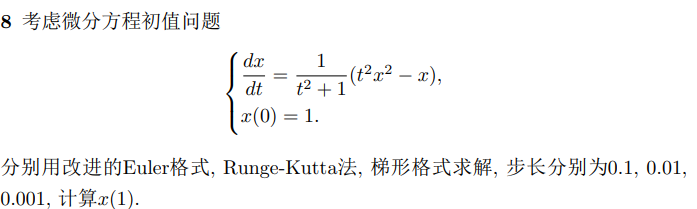
（3）

两点的结果为6.316241

三点的结果为6.316545

五点的结果为8.135540

## 第八题



**Matlab 代码：**

function R\_K(h)

%欧拉法

y = 1;

% fprintf('欧拉法：x=%f, y=%f\n',0,1);

for i=1:1/h

x = (i-1) \* h;

K = f(x,y);

y = y + h \* K;

end

fprintf('欧拉法：x = %.10f, y = %.10f\n',x+h,y);

fprintf('\n');

%改进的欧拉法

y = 1;

% fprintf('改进的欧拉法：x=%f, y=%f\n',0,1);

for i=1:1/h

x=(i-1)\*h;

K1=f(x,y);

K2=f(x+h,y+h\*K1);

y=y+(h/2)\*(K1+K2);

end

fprintf('改进的欧拉法：x = %.10f, y = %.10f\n',x+h,y);

fprintf('\n');

%龙格库塔方法

y=1;

% fprintf('龙格库塔法：x=%f, y=%f\n',0,1);

for i=1:1/h

x=(i-1)\*h;

K1=f(x,y);

K2=f(x+h/2,y+(h/2)\*K1);

K3=f(x+h/2,y+(h/2)\*K2);

K4=f(x+h,y+h\*K3);

y=y+(h/6)\*(K1+2\*K2+2\*K3+K4);

end

fprintf('龙格库塔法：x=%.10f, y = %.10f\n',x+h,y);

end

function z=f(x,y)

z = 1/(x^2 + 1) \*(x^2 \* y^2 - y);

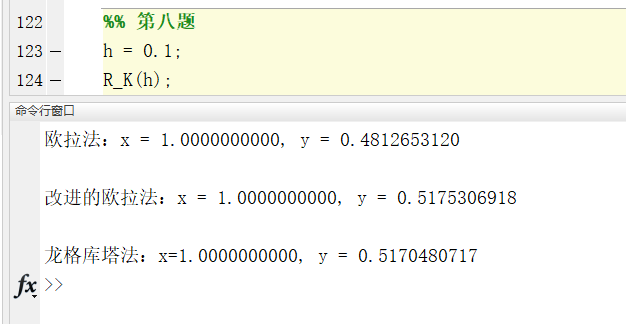
End

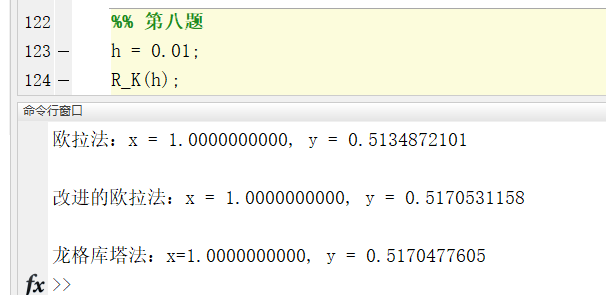
**运行主函数：**

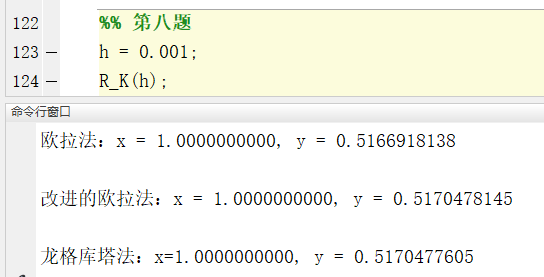
h = 0.001;

R\_K(h);

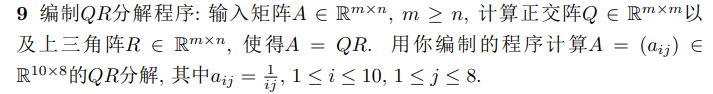
运行结果：h = 0.1，0.01，0.001的时候







## 第九题：



**Matlab 代码：**

function [Q,R] = qrfenjie()

A=zeros(10,8);

for i=1:10

for j=1:8

A(i,j)=1/(i\*j);

end

end

[M,N]=size(A);

A1=A;

H1=zeros(M,M);

for j=1:M

H1(j,j)=1;

end

for k=1:N

H0=zeros(M,M);

for i=1:M

H0(i,i)=1;

end

s=0;

for i=k:M

s=s+A1(i,k)\*A1(i,k);

end

s=sqrt(s);

u=zeros(N,1);

if (A1(k,k)>=0)

u(k)=A1(k,k)+s;

else

u(k)=A1(k,k)-s;

end

%-------------------------------

for i=k+1:M

u(i)=A1(i,k);

end

du=0;

for i=k:M

du=du+u(i)\*u(i);

end

for i=k:M

for j=k:M

H0(i,j)=-2\*u(i)\*u(j)/du;

if i==j

H0(i,j)=1+H0(i,j);

end

end

end

A2=H0\*A1;

H1=H1\*H0;

end

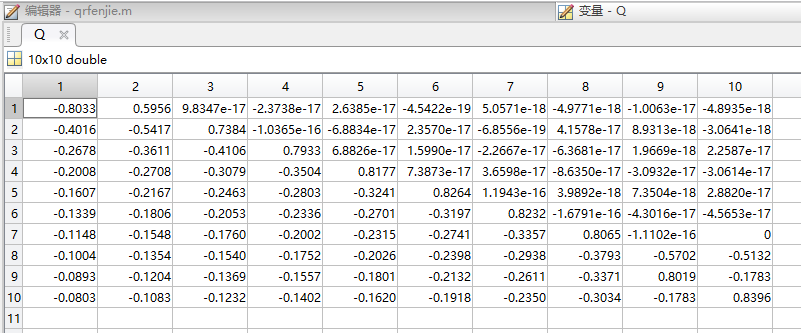
Q=H1;

R=A1;

end

**运行结果：**

**Q：**



**R：**

