实验一

1. 实验名称

利用递推的梯形公式求积分

1. 实验内容

自己找一个积分并给定误差，利用递推的梯形公式求出积分

1. 实验代码

#include<stdio.h>

#include<math.h>

void main()

{

int k;

double a,b,n,h,x;

double f,f1,f2,T,F=0.0;

printf("input a,b,n:");

scanf("%1f%1f%1f",&a,&b,&n);

h=(b-a)/n;

for(k=1;k<n;k++)

{

x=a+k\*h;

f=sin(x)/x;

printf("\nf=%1f",f);

F+=f;

printf("\nF=%1f",F);

}

if(a!=0.0)

f1=sin(a)/a;

else f1=1.0;

printf("\nf1=%1f",f1);

f2=sin(b)/b;

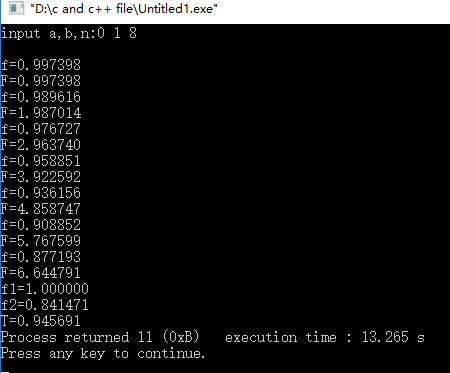
printf("\nf2=%1f",f2);

T=0.5\*h\*(f1+2\*F+f2);

printf("\nT=%1f0",T);

}

1. 实验结果截图



实验二

1. 实验名称

列主元消去法解线性方程

1. 实验内容

利用列主元消去法解线性方程组

C:\Users\Administrator\Documents\Tencent Files\1456067340\Image\C2C\OBG%UJ$M2AWP7(L42`)EI30.jpg

并求出系数矩阵A的行列式的值

1. 实验代码

#include<stdio.h>

#include<math.h>

void main()

{

void zhu(float \*,int,float[]);

int i;

float x[3];

float c[3][5]={12,-3,3,15,

-18,3,-1,-15,

1,1,1,6};

zhu(c[0],3,x);

for(i=0;i<=2;i++)

printf("x[%d]=%f\n",i+1,x[i]);

}

void zhu(float \*c,int n,float x[])

{

int i,j,t,k;

float p;

for(i=0;i<=n-2;i++)

{

k=i;

for(j=i+1;j<=n-1;j++)

if(fabs(\*(c+j\*(n+1)+i))>(fabs(\*(c+k\*(n+1)+i))))

k=j;

if(k!=i)

for(j=i;j<=n;j++)

{

p=\*(c+i\*(n+1)+j);

\*(c+i\*(n+1)+j)=\*(c+k\*(n+1)+j);

\*(c+k\*(n+1)+j)=p;

}

for(j=i+1;j<=n-1;j++)

{

p=(\*(c+j\*(n+1)+i))/(\*(c+i\*(n+1)+i));

for(t=i;t<=n;t++)

\*(c+j\*(n+1)+t)-=p\*(\*(c+i\*(n+1)+t));

}

}

for(i=n-1;i>=0;i--)

{

for(j=n-1;j>=i+1;j--)

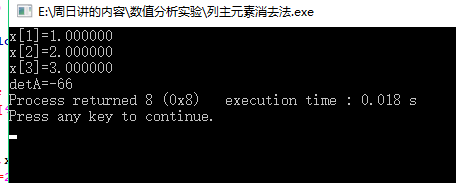
(\*(c+i\*(n+1)+n))-=x[j]\*(\*(c+i\*(n+1)+j));

x[i]=(\*(c+i\*(n+1)+n))/(\*(c+i\*(n+1)+i));

}

}

1. 实验结果截图



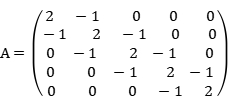
实验三

1. 实验名称

追赶法

1. 实验内容

利用追赶法解三对角线性方程



1. 实验代码

#include<stdio.h>

#include<math.h>

void main()

{

int i;

int n=4;

float x[5];

float c[5][6]={2,-1,0,0,0,1,

-1,2,-1,0,0,0,

0,-1,2,-1,0,0,

0,0,-1,2,-1,0,

0,0,0,-1,2,0};

void zhui(float \*,int,float[]);

zhui(c[0],5,x);

for(i=0;i<=4;i++)printf("x[%d]=%f\n",i+1,x[i]);

}

void zhui(float \*u,int n,float x[])

{

int i,j;

float p;

for(i=1;i<=n-1;i++)

{

p=\*(u+i\*(n+1)+i-1)/(\*(u+(i-1)\*(n+1)+i-1));

\*(u+i\*(n+1)+i-1)=0;

\*(u+i\*(n+1)+i)-=p\*(\*(u+(i-1)\*(n+1)+i));

\*(u+i\*(n+1)+n)-=p\*(\*(u+(i-1)\*(n+1)+n));

}

x[n-1]=(\*(u+(n-1)\*(n+1)+n))/(\*(u+(n-1)\*(n+1)+n-1));

for(j=n-2;j>=0;j--)

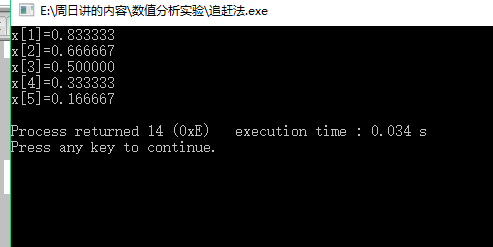
{

x[j]=((\*(u+j\*(n+1)+n))-x[j+1]\*(\*(u+j\*(n+1)+j+1)))/(\*(u+j\*(n+1)+j));

}

}

1. 实验结果截图



实验四

1. 实验名称

雅克比迭代法

1. 实验内容

利用雅克比迭代法或者高斯-赛德尔迭代法解方程组  
C:\Users\Administrator\Documents\Tencent Files\1456067340\Image\C2C\M]]TBTF2JQ@{709{CBX2G)0.jpg

（1）考察雅可比迭代法或高斯-赛德尔迭代法解此方程组的收敛性；（2）用雅可比迭代法或高斯-赛德尔迭代法解次方程组，要求当||C:\Users\Administrator\Documents\Tencent Files\1456067340\Image\C2C\XN@YAPI]343[FRQD]6}MFQ9.jpg-C:\Users\Administrator\Documents\Tencent Files\1456067340\Image\C2C\%}KHGA~4%@ZO`W}G1P`B[VN.jpg||<C:\Users\Administrator\Documents\Tencent Files\1456067340\Image\C2C\OPT(2JV13I0X]7ZHH%F88QM.jpg时迭代终止。

1. 实验代码

import numpy as np

A = np.array([[5.0, 2, 1], [-1, 4, 2], [2, -3, 10]])

B = np.array([-12.0, 20, 3])

x0 = np.array([0.0, 0, 0])

x = np.array([0.0, 0, 0])

times = 0

while True:

for i in range(3):

temp = 0

for j in range(3):

if i != j:

temp += x0[j] \* A[i][j]

x[i] = (B[i] - temp) / A[i][i]

calTemp = max(abs(x - x0))

times += 1

if calTemp < 1e-4:

break

else:

x0 = x.copy()

print(times)

print(x)

1. 实验结果截图

