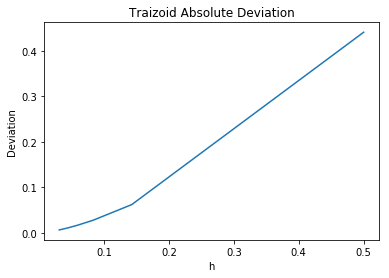
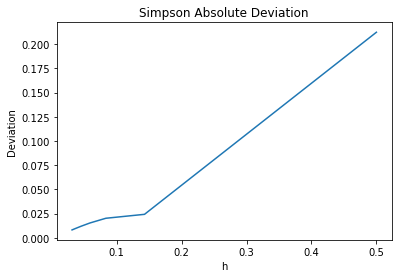
**1、用不同积分方法计算如下定积分值**

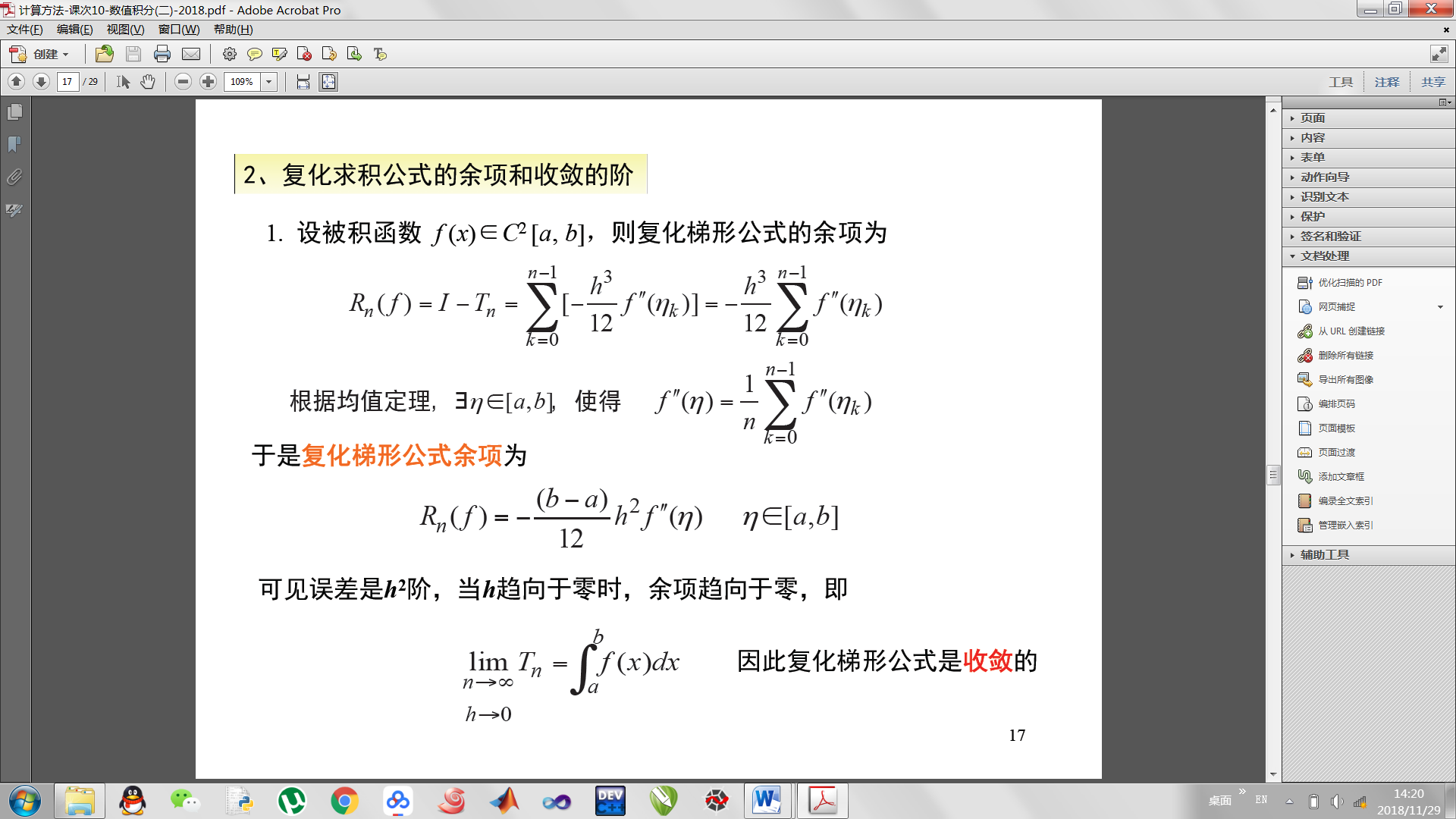
算法1误差由图例展示



横坐标h为步长，纵坐标为误差的绝对值。

可见，随着步长的减小，误差越来越小，逐渐趋近于0.

我觉得不存在一个最小的步长，使得精度不再改善，根据余项的公式



随着h的减小，Rn越来越小，直到收敛为0.

算法2运行结果如下图第一行，精度达到 **10－4**时龙贝格求积积分值为0.-444394

**2、计算Gamma函数值**

使用复化积分公式和高斯拉盖尔迭代法分别对x（1~10）进行计算

结果输出形式为

True value （真值，即x-1的阶乘）

Fuhua Jifen end = ?(截断终止点)

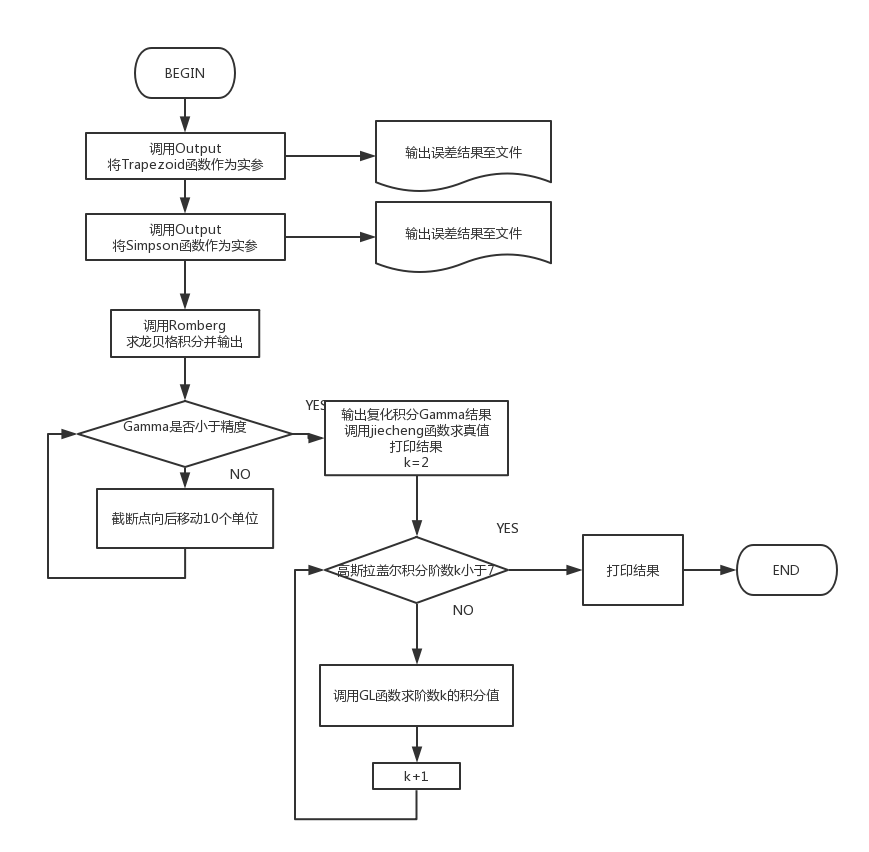
Gauss-Laguerre 不同阶的值

积分节点及求积公式系通过查阅数学手册得知

复化积分精度较好，但是其算法复杂，需要循环计算。

高斯拉盖尔积分精度稍差，但是算法只是几个乘方的加权求和，计算较快。

流程图如下



C语言实现代码如下：

#include<stdio.h>

#include<math.h>

#include<stdlib.h>

#define eps 1e-4

typedef double(\*pfun)(double,double,double);

double f(double x)

{

return sqrt(x) \* log(x);

}

double Trapezoid(double s,double e,double n)

{

double sum = f(s) + f(e);

double h = (e - s) / n;

double x = s + h;

int i = 0;

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

{

sum += 2 \* f(x);

x += h;

}

return sum \* h / 2;

}

double Simpson(double s,double e,double n)

{

double sum = f(s) + f(e);

double h = (e - s) / n;

double x = s + h;

double mid =s + h / 2;

int i = 0;

sum += 4 \* f(mid);

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

{

sum += 2 \* f(x);

sum += 4 \* f(mid);

x += h;

mid += h;

}

return h / 6 \* sum;

}

void Output(pfun fun,double s,double e,char name)

{

double n = 2;

char cBuffer[50] = {0};

FILE \*fp;

sprintf(cBuffer,"D:\\numerical\\6\\Output%c.txt",name);

if((fp = fopen(cBuffer,"w+")) == NULL)

{

printf("The file can not be opened.\n");

exit(1);

}

for(;n <= pow((float)2,5);n += 5)

{

fprintf(fp,"%d\t",(int)n);

fprintf(fp,"%lf\n",fun(s,e,n) + 0.444444);

}

fclose(fp);

}

double Romberg(double s,double e)

{

double n = 1;

double T1[4] = {0},T2[3] = {0},T3[2] = {0};

int i = 0;

while(n <=8)

{

T1[i++] = Trapezoid(s,e,n);

n \*= 2;

}

i = 0;

while(i < 3)

{

T2[i] = (4 \* T1[i+1] - T1[i]) / 3;

i++;

}

i = 0;

while(i<2)

{

T3[i] = (16 \* T2[i+1] - T2[i]) / 15;

i++;

}

double T1n = 0,T1o = T1[3];

double T2n = 0,T2o = T2[2];

double T3n = 0,T3o = T3[1];

double T4n = 0,T4o = (64 \* T3[1] - T3[0]) / 63;

while(fabs(T4n - T4o) > eps)

{

if(T4n != 0)

{

T4o = T4n;

}

T1n = Trapezoid(s,e,n);

T2n = (4 \* T1n - T1o) / 3;

T3n = (16 \* T2n - T2o) / 15;

T4n = (64 \* T3n - T3o) / 63;

T1o = T1n;

T2o = T2n;

T3o = T3n;

n \*= 2;

}

return T4n;

}

double G(double x,double t)

{

double e = 2.718281;

return pow(t,x-1) \* pow(e,-t);

}

double Gamma(double a,double s,double e)

{

double n = 1024;

double sum = G(a,s) + G(a,e);

double h = (e - s) / n;

double x = s + h;

double mid =s + h / 2;

int i = 0;

sum += 4 \* G(a,mid);

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

{

sum += 2 \* G(a,x);

sum += 4 \* G(a,mid);

x += h;

mid += h;

}

return h / 6 \* sum;

}

double GL(double x,int n)

{

switch(n)

{

case 2: return 0.853553 \* pow(0.585786,x-1) + 0.146446 \* pow(3.414214,x-1);

case 3: return 0.711093 \* pow(0.415775,x-1) + 0.278518 \* pow(2.294280,x-1) + 0.103893 \* pow(6.289945,x-1);

case 4: return 0.603154 \* pow(0.322548,x-1) + 0.357419 \* pow(1.745761,x-1) + 0.038888 \* pow(4.53662,x-1)\

+ 0.000522 \* pow(9.395071,x-1);

case 5: return 0.521756 \* pow(0.26356,x-1) + 0.398667 \* pow(1.413403,x-1) + 0.075942 \* pow(3.596426,x-1)\

+ 0.003612 \* pow(7.08581,x-1) + 0.000023 \* pow(12.6408,x-1);

case 6: return 0.458964 \* pow(0.222847,x-1) + 0.417 \* pow(1.188932,x-1) + 0.113373 \* pow(2.992736,x-1)\

+ 0.010399 \* pow(5.775144,x-1) + 0.000261 \* pow(9.837467,x-1) + 0.0000009 \* pow(15.982874,x-1);

default : return -1;

}

}

int jiecheng(int n)

{

if(n == 1 || n == 0)

{

return 1;

}

return n \* jiecheng(n-1);

}

int main()

{

double s = 0.000001,e = 1,n = 1000;;

Output(Trapezoid,s,e,'T');

Output(Simpson,s,e,'S');

printf("Romberg = %lf\n\n",Romberg(s,e));

double i = 10;

for(int j = 1;j <= 10;j++)

{

while(fabs(Gamma(j,0,i) - Gamma(j,0,10+i)) > eps)

{

i += 10;

}

printf("x = %d\tTrue value = %d\n",j,jiecheng(j - 1));

printf("Fuhua Jifen end = %lf\tvalue = %lf\n",i,Gamma(j,0,i));

printf("Gauss-Laguerre\n");

for(int k = 2;k <= 6;k++)

{

printf("%d\t%lf\n",k,GL(j,k));

}

i = 10;

printf("\n");

}

return 0;

}

绘图由python实现：

# -\*- coding: utf-8 -\*-

"""

Created on Sat Nov 24 10:38:14 2018

@author: Administrator

"""

import matplotlib.pyplot as plt

from math import log10

fps = open('D:\\numerical\\6\\OutputS.txt', 'r')

line = fps.readlines()

c1 = []

c2 = []

for i in line:

i = i.split()

c1.append(1/float(i[0]))

c2.append(abs(float(i[1])))

fig1 = plt.figure()

plt.plot(c1,c2)

plt.title("Simpson Absolute Deviation")

plt.xlabel("h")

plt.ylabel("Deviation")

fps = open('D:\\numerical\\6\\OutputT.txt', 'r')

line = fps.readlines()

c1 = []

c2 = []

for i in line:

i = i.split()

c1.append(1/float(i[0]))

c2.append(abs(float(i[1])))

fig1 = plt.figure()

plt.plot(c1,c2)

plt.title("Traizoid Absolute Deviation")

plt.xlabel("h")

plt.ylabel("Deviation")

运行结果屏幕截图在下一页：

