Newton-Cotes 和 Gauss 系列求积公式表格推导与数值实验验证

刘 yi 铭

学号:22214502 Email:letusgo126@126.com

1 Introduction

本文首先对 n 为任意大于等于 1 的整数的 Newton-Cotes 积分公式，编写系数推导的代码并在**附录 1** 中列出了 n 从 1 到 10 的系数表。然后本文对 n 为任意大于等于 0 的整数的 Gauss-Legendre、Gauss-Laguerre、Gauss-Hermite 三种高斯积分公式分别编写代码，计算 x_k 和 A_k 的值并在**附录 2、3、4** 中分别列出了 n 从 0 到 9 这些值准确到小数点后 15 位、23 位和 31 位的表格。最后本文对以上四种算法进行了数值实验验证和粗略对比。本文全部工作已经开源在 GitHub：https://github.com/LYMDLUT/NewtonCotes-Guass。

2 制作求积公式节点和系数表格

2.1 Newton-Cotes

设将积分区间 [a ,b] 划分为 n 等份, 步长 $h=\frac{b-a}{n}$, 选取等距节点 $x_k=a+kh$ 构造出的插值型求积公式1称为**牛顿柯特斯 (Newton-Cotes) 公式**。

$$I_n=(b-a)\sum_{k=0}^{n}C_k^{(n)}f(x_k) \tag{1}$$

公式1中的 $C_k^{(n)}$ 称为柯特斯系数，其计算公式为式2[1]:

$$C_k^{(n)}==\frac{(-1)^{n-k}}{nk!(n-k)!}\int_0^n\prod_{j=0}^{n}(t-j)\mathrm{d}t \tag{2}$$

我们按照式2编写代码来计算 n 为任意大于等于 1 的整数的柯特斯系数 [2]:

Listing 1: Cnk 系数的计算

```
function Cnk=Cotes_xishu(n,k)%计算柯特斯系数
%计算Cnk的系数
Ckn_xishu=(-1)^(n-k)/(n * factorial(k) * factorial(n-k));

syms x;%定义变量
fx = 1;
for i=0:n
fx = fx * (x-i);
end
fx = fx / (x-k);
Ckn_jifen=int(fx, x, 0, n);%计算Cnk的积分
```

```
Cnk = Ckn_xishu * Ckn_jifen;%乘在一起获得柯特斯系数
end
```

为了方便整理，计算结果详细列在**附录 1**，由于纸张大小限制，只列举到了 n 从 1 到 10 的 $C_k^{(n)}$。如果需要更大的 n，可以在 matlab 中运行代码获得。同时当 n 8 时，柯特斯系数 $C_k^{(n)}$ 出现负值，初始数据误差将会引起计算结果误差增大，即计算不稳定，一般不会使用。

n=7 的牛顿柯特斯 (Newton-Cotes) 公式为式3，其中 x_i, y_i 为一对采样点，i 的取值从 1 到 n+1:

$$\int_{x_1}^{x_8} f(x)\,dx = \tfrac{7}{17280} h\left(751 f_1 + 3577 f_2 + 1323 f_5 + 2989 f_4 + 2989 f_5 + 1323 f_6 + 3577 f_7 + 751 f_8\right) \tag{3}$$

n=9 的牛顿柯特斯 (Newton-Cotes) 公式为式4:

$$\int_{x_1}^{x_{10}} f(x)dx = \tfrac{9}{89600} h\left[2857(f_1+f_{10}) + 15741(f_2+f_9) + 1080(f_3+f_8) + 19344(f_4+f_7) + 5778(f_5+f_6)\right] \tag{4}$$

n=10 的牛顿柯特斯 (Newton-Cotes) 公式为式5:

$$\int_{x_0}^{x_{11}} f(x)dx = \tfrac{5}{299376} h\left[16067(f_1+f_{11}) + 106300\left(f_2+f_{10}\right) - 48525\left(f_3+f_9\right) + 272400\left(f_4+f_8\right) - 260550\left(f_5+f_7\right) + 427368 f_6\right] \tag{5}$$

2.2 Gauss-Legendre

高斯求积公式为式6:

$$\int_a^b f(x)\rho(x)\mathrm{d}x \approx \sum_{k=0}^{n} A_k f(x_k) \tag{6}$$

其中 A_k(k=0,l,···,n) 为不依赖于 $f(x)$ 的求积系数,x_k,(k=0,1,···,n) 为求积节点。

在高斯求积公式6中，若取权函数 $\rho(x) = 1$，区间为 [-1,1]，则得**高斯-勒让德 (Gauss-Legendre)** 求积公式，如式7:

$$\int_{-1}^{1} f(x)\mathrm{d}x \approx \sum_{k=0}^{n} A_k f(x_k) \tag{7}$$

勒让德多项式是区间 [-1,1] 上的正交多项式，因此 n+1 次勒让德多项式 $P_{n+1}(x)$ 的零点 $x_0, x_1, \cdots$,x_n 就是求积公式7的高斯点。n 阶勒让德多项式 $P_n(x)$ 的表示如式8，借此可以很容易利用 matlab 求出零点。

$$P_n(x) = \frac{1}{2^n n!}\frac{\mathrm{d}^n}{\mathrm{d}x^n}\left[(x^2-1)^n\right] \tag{8}$$

勒让德多项式的递推关系如式9[3]，借此可以很容易利用 matlab 求出零点 [2]。

$$\begin{gathered}
\mathrm{P_0(x) = 1.} \\
P_1(\mathrm{x}) = \mathrm{x} \\
\mathrm{(n+1)P_{n+1}(x) = (2n+1)xP_n(x) - nP_{n-1}(x)} \quad n = 1, 2, \ldots
\end{gathered} \tag{9}$$

勒让德多项式的系数为式10:

$$A_k = \tfrac{2\left(1-x_k^2\right)}{(n+1)^2\left[P_{n+1}(x_k)\right]^2} \tag{10}$$

我们利用式 8 计算勒让德多项式的零点和系数如下:

Listing 2: 勒让德多项式的零点和系数计算

```
function [Result1,Result2]=Guass_Legendre_Z(n)
%-----------------------------------------------------------------------------
```

```matlab
%Guass_Legendre_Z，意思是求n+1次勒让德多项式的零点。
%输入勒让德多项式次数
%输出对应勒让德多项式零点(也可以叫高斯点)Result1和高斯点对应的高斯系数Result2
%输出结果为两个数组
%--------------------------------------------------------------------------------
syms x;
p=sym2poly((x^2-1)^(n+1)); %多项式的n+1次幂转化为矩阵方便计算
for i=1:n+1
p=polyder(p);
end
pn=p./(factorial(n+1)*2^(n+1));%完全构建好的勒让德多项式
Result1=sort(roots(pn));
Result2=[];%预先开好一个空集
N=n+1;%方便循环
a(1:N)=1;%预先开好一个空集
%循环计算每个高斯系数Ak
for i=1:1:N
    b(x)=0*x+1;
    switch i
        case i==1
            for j=2:N
                a(i)=(Result1(i)-Result1(j))*a(i); %a(i)是分母上的，且总体上看是常数
                b(x)=(x-Result1(j))*b(x);
            end%b(x)是分子上的，且总体上看是x的多项式
        case i==N
            for j=1:i-1
                a(i)=(Result1(i)-Result1(j))*a(i);
                b(x)=(x-Result1(j))*b(x);
            end
        otherwise
            for j=[1:(i-1),(i+1):N]
                a(i)=(Result1(i)-Result1(j))*a(i);
                b(x)=(x-Result1(j))*b(x);
            end
    end
    Result2(i)=int(b(x),x,-1,1)/a(i);%高斯系数公式
end
%------------------------------
%输出结果，精确到小数点15位
Result1=vpa(Result1,20);
Result2=vpa(Result2,20);
for i =1:length(Result1)
fprintf('%d:%.15f\n',i,Result1(i))
fprintf('%d:%.15f\n',i,Result2(i))
end
%------------------------------
```

为了方便整理，计算结果详细列在**附录 2**，由于纸张大小限制，只列举到了 n 从 0 到 9 的 x_k，A_k，并且精

确到小数点后 15 位。如果需要更大的 n，可以在 matlab 中运行代码获得。

当积分区间不是 [— 1,1], 而是一般的区间 [a ,b] 时, 只要做如式11变换

$$x=\frac{b-a}{2}t+\frac{a+b}{2} \tag{11}$$

将 [a,b] 化为 [— 1,1], 这时求积公式就如式12:

$$\int_a^b f(x)\mathrm{d}x=\frac{b-a}{2}\int_{-1}^{1} f\left(\frac{b-a}{2}t+\frac{a+b}{2}\right)\mathrm{d}t \tag{12}$$

2.3 Gauss-Laguerre

区间为 $[0,+\infty)$, 权函数 $\rho(x)=\mathrm{e}^{-x}$ 的正交多项式为**拉盖尔多项式 (Laguerre)**, 如式13:

$$\mathrm{L}_n(x)=\mathrm{e}^x\frac{\mathrm{d}^n}{\mathrm{d}x^n}(x^n\mathrm{e}^{-x}) \tag{13}$$

其对应的高斯型求积公式称为**高斯-拉盖尔求积公式 (Gauss-Laguerre)**。

拉盖尔多项式的递推关系如式14[4]:

$$\begin{aligned}\mathrm{L}_0(\mathrm{x})&=1\\ \mathrm{L}_1(\mathrm{x})&=1-\mathrm{x}\\ \mathrm{L}_{n+1}(\mathrm{x})&=(2\mathrm{n}-\mathrm{x}+1)\mathrm{L}_\mathrm{n}(\mathrm{x})-\mathrm{n}^2\mathrm{L}_{\mathrm{n}-1}(\mathrm{x}),\quad \mathrm{n}=1,2,\ldots\end{aligned} \tag{14}$$

借此可以很容易利用 matlab 求出 n+1 阶拉盖尔多项式的零点 $x_k\,(k=0,1,\cdots,n)$。

拉盖尔多项式的系数为式15:

$$A_k=\frac{x_k}{(n+1)^2\left[L_{n+1}\left(x_k\right)\right]^2} \tag{15}$$

按照式14和式15编写代码来计算 n 为任意大于等于 0 的整数的拉盖尔多项式的 x_k, A_k:

Listing 3: 按公式 14 迭代拉盖尔多项式

```
function p=Guass_Laguerre_n_1(x,n)
if n==0
    p = 1;
elseif n==1
    p = 1-x;
else
    p=((2*n - 1-x)*Guass_Laguerre_n_1(x,(n-1)) - (n-1)*Guass_Laguerre_n_1(x,n-2))/(n);
end
end
```

Listing 4: 按公式 15 计算 x_k, A_k

```
function [Result1,Result2]=Guass_Laguerre_Z(n)
%-----------------------------------------------------------------------------------------
%输入Laguerre多项式次数
%输出对应Laguerre多项式零点(也可以叫高斯点)Result1和高斯点对应的高斯系数Result2
%输出结果为两个数组
%-----------------------------------------------------------------------------------------
syms x;
p = Guass_Laguerre_n_1(x,n+1);
```

```
pn = sym2poly(p);
Result1=sort(roots(pn));
Result2=[];
for i=1:n+1
    Result2(i) = Result1(i)/((n+2)^2*Guass_Laguerre_n_1(Result1(i),n+2)^2);
end
Result1=vpa(Result1,30);
Result2=vpa(Result2,30);
for i =1:length(Result1)
fprintf('%d:%.23f\n',i,Result1(i))%输出结果x_k
fprintf('%d:%.23f\n',i,Result2(i))%输出结果A_k
end
```

为了方便整理，计算结果详细列在**附录 3**，由于纸张大小限制，只列举到了 n 从 0 到 9 的 x_k，A_k，并且精确到小数点后 23 位。如果需要更大的 n，可以在 matlab 中运行代码获得。

2.4 Gauss-Hermite

区间为 $(-\infty,+\infty)$, 权函数 $\rho(x)=\mathrm{e}^{-x^2}$ 的正交多项式为**埃尔米特多项式 (Hermite)**, 如式16:

$$\mathrm{H}_n(x)=(-1)^n\mathrm{e}^{x}\frac{\mathrm{d}^n}{\mathrm{d}x^n}\mathrm{e}^{-x^2},\quad n=0,1,\cdots \tag{16}$$

其对应的高斯型求积公式称为**高斯-埃尔米特求积公式 (Gauss-Hermite)**。

埃尔米特多项式的递推关系如式17:

$$\begin{aligned}\mathrm{H}_0(\mathrm{x})&=1\\ \mathrm{H}_1(\mathrm{x})&=2\mathrm{x}\\ \mathrm{H}_{n+1}(\mathrm{x})&=2\mathrm{x}\mathrm{H}_n(\mathrm{x})-2\mathrm{n}\mathrm{H}_{n-1}(\mathrm{x}),\quad \mathrm{n}=1,2,\ldots.\end{aligned} \tag{17}$$

借此可以很容易利用 matlab 求出 n+1 阶埃尔米特多项式的零点 $x_k\ (k=0,1,\cdots,n)$[2]。

埃尔米特多项式的系数为式18:

$$A_k=2^{n+2}\,(n+1)!\frac{\sqrt{\pi}}{[2(n+1)H_n(x_k)]^2} \tag{18}$$

按照式17和式18编写代码来计算 n 为任意大于等于 0 的整数的埃尔米特多项式的 x_k, A_k:

Listing 5: 按公式 17 迭代埃尔米特多项式

```
function p=Guass_Hermite_n_1(x,n)
if n==0
    p = 1;
elseif n==1
    p = 2*x;
else
    p=(2*x)*Guass_Hermite_n_1(x,(n-1)) - (2*(n-1))*Guass_Hermite_n_1(x,(n-2));
end
end
```

Listing 6: 按公式 18 计算 x_k，A_k

```
function [Result1,Result2]=Guass_Hermite_Z(n)
%-----------------------------------------------------------------------------------------
%输入Hermite多项式次数
%输出对应Hermite多项式零点(也可以叫高斯点)Result1和高斯点对应的高斯系数Result2
%输出结果为两个数组
%-----------------------------------------------------------------------------------------
syms x;
p = Guass_Hermite_n_1(x,n+1);
pn = sym2poly(p);
Result1=sort(roots(pn));
Result2=[];
for i=1:n+1
    Result2(i) = factorial(n+1)*sqrt(pi)*2^(n+2)/(2*(n+1)*Guass_Hermite_n_1(Result1(i),n))^2;
end
Result1=vpa(Result1,40);
Result2=vpa(Result2,40);
for i =1:length(Result1)
fprintf('%d:%.31f\n',i,Result1(i)))%输出结果x_k
fprintf('%d:%.31f\n',i,Result2(i))%输出结果A_k
end
```

为了方便整理，计算结果详细列在**附录 4**，由于纸张大小限制，只列举到了 n 从 0 到 9 的 x_k，A_k，并且精确到小数点后 31 位。如果需要更大的 n，可以在 matlab 中运行代码获得。

3 数值实验验证

3.1 Newton-Cotes

$$I = \int_0^{\frac{\pi}{2}} x^2 \cos x \mathrm{d}x \tag{19}$$

$$I = \int_0^2 \frac{1}{x^2+1} \mathrm{d}x \tag{20}$$

设计实验 Listing 7 利用 Newton-Cotes 方法计算不同 n 时式19和式20的积分值，与 matlab 计算得到的高精度参考值比较相对误差，相对误差的定义为式21。

$$\text{相对误差} = \frac{|\text{预测值} - \text{参考值}|}{\text{参考值}} \tag{21}$$

Listing 7: 计算 Newton-Cotes 算法 n 从 1 到 20 的值和相对误差

```
clear
x_1=0;%积分起始点
x_n_1=2;%积分结束点

for n=1:20%n从1到20
```

```
h=(x_n_1-x_1)/n;

inegral = 0;%积分值
for k = 0:n
Cnk = xishu(n,k);
x_k =x_1 + h*k;
inegral = inegral + Cnk*FX(x_k);
end

acc=20;%输出结果的有效位数
cal = vpa(inegral*(x_n_1-x_1),acc);%计算值

syms x1;
fx1 =FX(x1);
ref = vpa(int(fx1,x1,x_1,x_n_1),acc);%利用matlab积分算得高精度参考值
fprintf('n=%d,预测值:%.20f,相对误差:%.20f\n',n,cal,abs((cal - ref)/ref))
end

function fx=FX(x)
%fx = x^2*cos(x);%公式x^2*cos(x)
fx = 1/(x^2+1);%公式1/(x^2+1)
end
```

按以上代码测得如下结果

式19的结果：

n=1, 预测值:0.00000000000000011866, 相对误差:0.99999999999999977796

n=2, 预测值:0.45676559374971492655, 相对误差:0.02275456030468849325

n=3, 预测值:0.46283737834639748776, 相对误差:0.00976403761840321724

n=4, 预测值:0.46756522949499856878, 相对误差:0.00035115283760198026

n=5, 预测值:0.46749304251529161247, 相对误差:0.00019670951330325381

n=6, 预测值:0.46740000697157069176, 相对误差:0.00000233910610894401

n=7, 预测值:0.46740043137035802712, 相对误差:0.00000143110912930812

n=8, 预测值:0.46740110459737677084, 相对误差:0.00000000925337379006

n=9, 预测值:0.46740110303962684180, 相对误差:0.00000000592058338505

n=10, 预测值:0.46740110026089504158, 相对误差:0.00000000002448564785

n=11, 预测值:0.46740110026481540562, 相对误差:0.00000000001609806803

n=12, 预测值:0.46740110027236081436, 相对误差:0.00000000000004525004

n=13, 预测值:0.46740110027235431955, 相对误差:0.00000000000003132334

n=14, 预测值:0.46740110027233960910, 相对误差:0.00000000000000005723

n=15, 预测值:0.46740110027233949808, 相对误差:0.00000000000000028104

n=16, 预测值:0.46740110027233927603, 相对误差:0.00000000000000081966

n=17, 预测值:0.46740110027233977563, 相对误差:0.00000000000000024758

n=18, 预测值:0.46740110027234144097, 相对误差:0.00000000000000385271

n=19, 预测值:0.46740110027233883194, 相对误差:0.00000000000000176637

n=20, 预测值:0.46740110027234404999, 相对误差:0.00000000000000939518
参考值 = 0.46740110027233965471

式20的结果:
n=1, 预测值:1.19999999999999995559, 相对误差:0.08386523031062041722
n=2, 预测值:1.06666666666666665186, 相对误差:0.03656423972389295785
n=3, 预测值:1.08923076923076922462, 相对误差:0.01618386787189838444
n=4, 预测值:1.10769230769230775380, 相对误差:0.00049098182518808381
n=5, 预测值:1.10765414097824654860, 相对误差:0.00045650884658291700
n=6, 预测值:1.10759232923938810700, 相对误差:0.00040067918443822248
n=7, 预测值:1.10740655391790610018, 相对误差:0.00023288300810152875
n=8, 预测值:1.10711128348775411645, 相对误差:0.00003381145254904156
n=9, 预测值:1.10712154595111256228, 相对误差:0.00002454217987273137
n=10, 预测值:1.10713813137524441643, 相对误差:0.00000956187608406614
n=11, 预测值:1.10714234029633140644, 相对误差:0.00000576029006455702
n=12, 预测值:1.10715068801344562743, 相对误差:0.00000177954354600371
n=13, 预测值:1.10715011162042542558, 相对误差:0.00000125893325117409
n=14, 预测值:1.10714899729116122984, 相对误差:0.00000025244763069770
n=15, 预测值:1.10714887967549380221, 相对误差:0.00000014621468698740
n=16, 预测值:1.10714861571109368654, 相对误差:0.00000009220350913154
n=17, 预测值:1.10714864657974443318, 相对误差:0.00000006432229459781
n=18, 预测值:1.10714871240945567088, 相对误差:0.00000000486351535447
n=19, 预测值:1.10714871541871318072, 相对误差:0.00000000214549072966
n=20, 预测值:1.10714872298131772332, 相对误差:0.00000000468521262281
参考值 = 1.107148717794090503

3.2 Gauss-Legendre

设计实验 Listing 8 利用 Gauss-Legendre 方法计算不同 n 时式19和式20的积分值，与 matlab 计算得到的高精度参考值比较相对误差:

Listing 8: 计算 Gauss-Legendre 算法 n 从 0 到 19 的值和相对误差

```
clear
x_1=0;%积分起始点
x_n_1=pi/2;%积分结束点

for n=0:19%n从0到19
[A,B] = Guass_Legendre_Z(n);%求x_k和A_k
I = 0;
for i=1:n+1
    x = (x_n_1 - x_1)*A(i)/2 + (x_n_1 + x_1)/2;
    fx = (x_n_1 - x_1)/2 * FX(x);
    I = I + fx * B(i);
end
```

```

acc = 20;%输出结果的有效位数
cal = vpa(I,acc);%计算值
syms t;
fx2 = FX(t);
ref = vpa(int(fx2,t,x_1,x_n_1),acc);%利用matlab积分算得高精度参考值
fprintf('n=%d,预测值:%.20f,相对误差:%.20f\n',n,cal,abs((cal - ref)/ref))
end

function fx=FX(x)
fx = x^2*cos(x);%公式x^2*cos(x)
%fx = 1/(x^2+1);%公式1/(x^2+1)
end
```

按以上代码测得如下结果

式19的结果:

n=0, 预测值:0.68514839062457233432, 相对误差:0.46586815954296706410

n=1, 预测值:0.47463609897204783739, 相对误差:0.01547920767728741022

n=2, 预测值:0.46724250353022228621, 相对误差:0.00033931615057165946

n=3, 预测值:0.46740206591233340871, 相对误差:0.00000206597715151154

n=4, 预测值:0.46740109737696844405, 相对误差:0.00000000619461787524

n=5, 预测值:0.46740110027756315292, 相对误差:0.00000000001117562241

n=6, 预测值:0.46740110027233344736, 相对误差:0.00000000000001328056

n=7, 预测值:0.46740110027233955359, 相对误差:0.00000000000000021635

n=8, 预测值:0.46740110027233927603, 相对误差:0.00000000000000081018

n=9, 预测值:0.46740110027233972012, 相对误差:0.00000000000000013995

n=10, 预测值:0.46740110027233960910, 相对误差:0.00000000000000009758

n=11, 预测值:0.46740110027233955359, 相对误差:0.00000000000000021635

n=12, 预测值:0.46740110027233960910, 相对误差:0.00000000000000009758

n=13, 预测值:0.46740110027233972012, 相对误差:0.00000000000000013995

n=14, 预测值:0.46740110027233972012, 相对误差:0.00000000000000013995

n=15, 预测值:0.46740110027233960910, 相对误差:0.00000000000000009758

n=16, 预测值:0.46740110027233966461, 相对误差:0.00000000000000002118

n=17, 预测值:0.46740110027233955359, 相对误差:0.00000000000000021635

n=18, 预测值:0.46740110027233960910, 相对误差:0.00000000000000009758

n=19, 预测值:0.46740110027233955359, 相对误差:0.00000000000000021635

参考值 = 0.46740110027233965471

式20的结果:

n=0, 预测值:1.00000000000000000000, 相对误差:0.09677897474114964538

n=1, 预测值:1.13513513513513508713, 相对误差:0.02527792056410040067

n=2, 预测值:1.10703363914373098531, 相对误差:0.00010394145656321729

n=3, 预测值:1.10673998241614146565, 相对误差:0.00036917838713076506

n=4, 预测值:1.10717399816107708865, 相对误差:0.00002283375898854387

n=5, 预测值:1.10715321898259499989, 相对误差:0.00000406556809591501
n=6, 预测值:1.10714809427273275233, 相对误差:0.00000056317760001837
n=7, 预测值:1.10714868825668233399, 相对误差:0.00000002667880808992
n=8, 预测值:1.10714872846950673768, 相对误差:0.00000000964226039654
n=9, 预测值:1.10714871761328015332, 相对误差:0.00000000016331170943
n=10, 预测值:1.10714871765068134657, 相对误差:0.00000000012953016532
n=11, 预测值:1.10714871780482471131, 相对误差:0.00000000000969536262
n=12, 预测值:1.10714871779554369091, 相对误差:0.00000000000131254986
n=13, 预测值:1.10714871779385415351, 相对误差:0.00000000000021347584
n=14, 预测值:1.10714871779408352559, 相对误差:0.00000000000000630216
n=15, 预测值:1.10714871779409418373, 相对误差:0.00000000000000332450
n=16, 预测值:1.10714871779409040897, 相对误差:0.00000000000000008494
n=17, 预测值:1.10714871779409040897, 相对误差:0.00000000000000008494
n=18, 预测值:1.10714871779409085306, 相对误差:0.00000000000000031617
n=19, 预测值:1.10714871779409040897, 相对误差:0.00000000000000008494
参考值 = 1.107148717794090503

可以发现，相同的 n 下，Guass-Legendre 公式的相对误差比 Newton-Cote 低很多，如果被求函数已知的情况下，用 Guass-Legendre 公式效率更高。但如果只有离散的点列，Newton-Cote 积分公式也是良好的选择。

3.3 Gauss-Laguerre

$$\int_0^{+\infty} \mathrm{e}^{-x} \sin x \mathrm{d}x \tag{22}$$

设计实验 Listing 9 利用 Gauss-Laguerre 方法计算不同 n 时式22的积分值，与 matlab 计算得到的高精度参考值比较相对误差:

Listing 9: 计算 Gauss-Laguerre 算法 n 从 0 到 19 的值和相对误差

```
x_1=0;%积分起始点
x_n_1=inf;%积分结束点

for n=0:19%n从0到19
[A,B] = Guass_Laguerre_Z(n);%求x_k和A_k
I = 0;
for i=1:n+1
    x = A(i);
    fx = FX(x);
    I = I + fx * B(i);
end

acc = 20;%输出结果的有效位数
cal = vpa(I,acc);%计算值
syms t;
fx2 =FX(t);
ref = vpa(int(exp(-t)*fx2,t,x_1,x_n_1),acc)%利用matlab积分算得高精度参考值
fprintf('n=%d,预测值:%.20f,相对误差:%.20f\n',n,cal,abs((cal - ref)/ref))
```

```
end

function fx=FX(x)
fx = sin(x);%公式
end
```

按以上代码测得如下结果

式22的结果:

n=0, 预测值:0.84147098480789650488, 相对误差:0.68294196961579300975

n=1, 预测值:0.43245945467984409083, 相对误差:0.13508109064031181834

n=2, 预测值:0.49602982748056151374, 相对误差:0.00794034503887697252

n=3, 预测值:0.50487927946020039194, 相对误差:0.00975855892040078388

n=4, 预测值:0.49890332095605915974, 相对误差:0.00219335808788168052

n=5, 预测值:0.50004947479767292151, 相对误差:0.00009894959534584302

n=6, 预测值:0.50003891199466321549, 相对误差:0.00007782398932643098

n=7, 预测值:0.49998775373532777788, 相对误差:0.00002449252934444424

n=8, 预测值:0.50000135242344856401, 相对误差:0.00000270484689712802

n=9, 预测值:0.50000020496483155164, 相对误差:0.00000040992966310327

n=10, 预测值:0.49999988871497424991, 相对误差:0.00000022257005150017

n=11, 预测值:0.50000001890882339595, 相对误差:0.00000003781764679189

n=12, 预测值:0.50000000011466672056, 相对误差:0.00000000022933344113

n=13, 预测值:0.49999999915773141179, 相对误差:0.00000000168453717642

n=14, 预测值:0.50000000020380364063, 相对误差:0.00000000040760728126

n=15, 预测值:0.49999999998430322279, 相对误差:0.00000000003139355442

n=16, 预测值:0.49999999999687710917, 相对误差:0.00000000000624578167

n=17, 预测值:0.50000000001125288751, 相对误差:0.00000000002250577502

n=18, 预测值:0.50000000002162758861, 相对误差:0.00000000004325517722

n=19, 预测值:0.50000000000294653191, 相对误差:0.00000000000589306381

参考值 = 0.5

3.4 Gauss-Hermite

$$\int_{-\infty}^{+\infty} \mathrm{e}^{-x^2} x^2 \mathrm{d}x \tag{23}$$

Listing 10: 计算 Gauss-Hermite 算法 n 从 0 到 19 的值和相对误差

```
x_1=-inf;%积分起始点
x_n_1=inf;%积分结束点

for n=1:19
[A,B] = Guass_Hermite_Z(n);%求x_k和A_k
I = 0;
for i=1:n+1
```

```
    x = A(i);
    fx = FX(x);
    I = I + fx * B(i);
end

acc = 20;%输出结果的有效位数
cal = vpa(I,acc);%计算值
syms t;
fx2 =FX(t);
ref = vpa(int(exp(-t^2)*fx2,t,x_1,x_n_1),acc);%利用matlab积分算得高精度参考值
fprintf('n=%d,预测值:%.20f,相对误差:%.20f\n',n,cal,abs((cal - ref)/ref))
end

function fx=FX(x)
fx = x^2;%函数e^(-x^2)*x^2
end
```

按以上代码测得如下结果

式23的结果:

n=1, 预测值:0.88622692545275794096, 相对误差:0.00000000000000008202

n=2, 预测值:0.88622692545275805198, 相对误差:0.00000000000000004325

n=3, 预测值:0.88622692545275605358, 相对误差:0.00000000000000221170

n=4, 预测值:0.88622692545275993936, 相对误差:0.00000000000000217293

n=5, 预测值:0.88622692545275938425, 相对误差:0.00000000000000154656

n=6, 预测值:0.88622692545275538745, 相对误差:0.00000000000000296335

n=7, 预测值:0.88622692545275782994, 相对误差:0.00000000000000020730

n=8, 预测值:0.88622692545275705278, 相对误差:0.00000000000000108422

n=9, 预测值:0.88622692545275727483, 相对误差:0.00000000000000083367

n=10, 预测值:0.88622692545276071652, 相对误差:0.00000000000000304986

n=11, 预测值:0.88622692545275472131, 相对误差:0.00000000000000371500

n=12, 预测值:0.88622692545275572051, 相对误差:0.00000000000000258753

n=13, 预测值:0.88622692545276093856, 相对误差:0.00000000000000330041

n=14, 预测值:0.88622692545275949527, 相对误差:0.00000000000000167183

n=15, 预测值:0.88622692545275671971, 相对误差:0.00000000000000146005

n=16, 预测值:0.88622692545275694176, 相对误差:0.00000000000000120950

n=17, 预测值:0.88622692545275394416, 相对误差:0.00000000000000459193

n=18, 预测值:0.88622692545275771892, 相对误差:0.00000000000000033257

n=19, 预测值:0.88622692545275583154, 相对误差:0.00000000000000246225

参考值 = 0.88622692545275801365

4 不足与未来工作

本文的实验发现随着 n 增大相对误差不是单调递减的，这一方面有可能是 n 增大导致算法稳定性下降导致的，另一方面可能是由截断误差导致的，我将继续深入探索研究并解释这个问题。

参考文献

[1] Newton-cotes formulas. `https://mathworld.wolfram.com/Newton-CotesFormulas.html`.

[2] G. R. Lindfield and Jet Penny. Numerical methods using matlab. 2012.

[3] Legendre-gaussquadrature. `https://mathworld.wolfram.com/Legendre-GaussQuadrature.html`.

[4] Laguerre-gaussquadrature. `https://mathworld.wolfram.com/Laguerre-GaussQuadrature.html`.

附录 1

$C_k^{(n)}$	1	2	3	4	5	6	7	8	9	10	11
1	$\frac{1}{2}$	$\frac{1}{2}$									
2	$\frac{1}{6}$	$\frac{2}{3}$	$\frac{2}{3}$								
3	$\frac{1}{8}$	$\frac{3}{8}$	$\frac{3}{8}$	$\frac{1}{8}$							
4	$\frac{7}{90}$	$\frac{16}{45}$	$\frac{2}{15}$	$\frac{16}{45}$	$\frac{7}{90}$						
5	$\frac{19}{288}$	$\frac{25}{96}$	$\frac{25}{144}$	$\frac{25}{144}$	$\frac{25}{96}$	$\frac{19}{288}$					
6	$\frac{41}{840}$	$\frac{9}{35}$	$\frac{9}{280}$	$\frac{34}{105}$	$\frac{9}{280}$	$\frac{9}{35}$	$\frac{41}{840}$				
7	$\frac{751}{17280}$	$\frac{3577}{17280}$	$\frac{1323}{17280}$	$\frac{2989}{17280}$	$\frac{2989}{17280}$	$\frac{1323}{17280}$	$\frac{3577}{17280}$	$\frac{751}{17280}$			
8	$\frac{989}{28350}$	$\frac{5888}{28350}$	$\frac{-928}{28350}$	$\frac{10496}{28350}$	$\frac{-4540}{28350}$	$\frac{10496}{28350}$	$\frac{-928}{28350}$	$\frac{5888}{28350}$	$\frac{989}{28350}$		
9	$\frac{2857}{89600}$	$\frac{15741}{89600}$	$\frac{27}{2240}$	$\frac{1209}{5600}$	$\frac{2889}{44800}$	$\frac{2889}{44800}$	$\frac{1209}{5600}$	$\frac{27}{2240}$	$\frac{15741}{89600}$	$\frac{2857}{89600}$	
10	$\frac{16067}{598752}$	$\frac{26575}{149688}$	$\frac{-16175}{199584}$	$\frac{5675}{12474}$	$\frac{-4825}{11088}$	$\frac{17807}{24948}$	$\frac{-4825}{11088}$	$\frac{5675}{12474}$	$\frac{-16175}{199584}$	$\frac{26575}{149688}$	$\frac{16067}{598752}$

表 1: Cotes 系数，纵轴 (1-10) 代表的是 $C_k^{(n)}$ 中的 n，横轴 (1-11) 代表的是 $C_k^{(n)}$ 中的 k

附录2

高斯-勒让德求积公式的节点和系数(准确到小数点后15位)

	1	2	3	4	5
n=0					
x_k	0.000000000000000				
A_k	2.000000000000000				
n=1					
x_k	+-0.577350269189625				
A_k	1.000000000000000				
n=2					
x_k	+-0.774596669241483	0.000000000000000			
A_k	0.555555555555555	0.888888888888888			
n=3					
x_k	+-0.861136311594053	+-0.339981043584856			
A_k	0.347854845137452	0.652145154862547			
n=4					
x_k	+-0.906179845938664	+-0.538469310105683	0.000000000000000		
A_k	0.236926885056188	0.478628670499366	0.568888888888888		
n=5					
x_k	+-0.932469514203153	+-0.661209386466263	+-0.238619186083196		
A_k	0.171324492379169	0.360761573048141	0.467913934572688		
n=6					
x_k	+-0.949107912342759	+-0.741531185599395	+-0.405845151377396	0.000000000000000	
A_k	0.129484966168868	0.279705391489277	0.381830050505120	0.417959183673469	
n=7					
x_k	+-0.960289856497537	+-0.796666477413629	+-0.525532409916328	+-0.183434642495649	
A_k	0.101228536290373	0.222381034453375	0.313706645877890	0.362683783378359	
n=8					
x_k	+-0.968160239507623	+-0.836031107326636	+-0.613371432700588	+-0.324253423403808	0.000000000000000
A_k	0.081274388361576	0.180648160694853	0.260610696402940	0.312347077039996	0.330239355001259
n=9					
x_k	+-0.973906528517168	+-0.865063366688988	+-0.679409568299025	+-0.433395394129247	+-0.148874338981631
A_k	0.066671344308690	0.149451349150572	0.219086362515990	0.269266719309993	0.295524224714753

附录3

高斯-拉盖尔求积公式的节点和系数(准确到小数点后23位)

	1	2	3	4
n=0				
x_k	1.00000000000000000000000			
A_k	1.00000000000000000000000			
n=1				
x_k	0.58578643762690496554768	3.41421356237309492343001		
A_k	0.85355339059327339779059	0.14644660940672626914249		
n=2				
x_k	0.41577455678347896572688	2.29428036027904491689355	6.28994508293747589533495	
A_k	0.71109300992917512385105	0.27851773356923820168518	0.01038925650158617312868	
n=3				
x_k	0.32254768961939223048673	1.74576110115834626235198	4.53662029692113488721361	9.39507091230111868185304
A_k	0.60315410434163585495781	0.35741869243780022280532	0.03888790851500484313518	0.00053929470556133500113
n=4				
x_k	0.26356031971814092296213	1.41340305910651853338322	3.59642577104073479787871	7.08581000585880893538615
A_k	0.52175561058280905957218	0.39866681108317042481203	0.07594244968170463239154	0.00361175867992220801447
n=5				
x_k	0.22284660417926105413499	1.18893210167262375343000	2.99273632605930517414094	5.77514356910454296212265
A_k	0.45896467394995149602365	0.41700083077211880233647	0.11337338207404944190326	0.01039919745314831812932
n=6				
x_k	0.19304367656036222622439	1.02666489533919658150296	2.56787674495071893687736	4.90035308452654749800103
A_k	0.40931895170128013150545	0.42183127786168900241747	0.14712634865752860502396	0.02063351446871297331653
n=7				
x_k	0.17027963230510118064486	0.90370177679937702119872	2.25108662986612850787082	4.26670017028773962408649
A_k	0.36918858934162551710400	0.41878678081436682134608	0.17579498663717479933765	0.03334349226120489340097
n=8				
x_k	0.15232222773180814634486	0.80722002274225734819168	2.00513515561930999453466	3.78347397333147039688583
A_k	0.33612642179795326757840	0.41121398042397411254356	0.19928752537095517638476	0.04746056276559497411060
n=9				
x_k	0.13779347054049237430994	0.72945454950317412112781	1.80834290174028189390753	3.40143369785533922211584
A_k	0.30844111576502331040217	0.40111992915522998170985	0.21806828761189575582868	0.06208745609850930408102

附录3

高斯-拉盖尔求积公式的节点和系数(准确到小数点后23位)

	5	6	7	8
n=4				
x_k	12.64080084427581063266643			
A_k	0.00002336997238577565852			
n=5				
x_k	9.83746741838254301626420	15.98287398060169905988914		
A_k	0.00026101720281494826493	0.00000089854790642962314		
n=6				
x_k	8.18215344456276838513986	12.73418029179789989768778	19.39572786226251821517507	
A_k	0.00107401014328092816313	0.00001586546434856258639	0.00000003170315478995633	
n=7				
x_k	7.04590540239311380332765	10.75851601018164416245781	15.74067864127750482339251	22.86313173688942868011508
A_k	0.00279453623522804691856	0.00009076508773348906889	0.00000084857467162771063	0.00000000104800117487138
n=8				
x_k	6.20495677787607835540484	9.37298525168795926276743	13.46623691109206966132205	18.83359778899173164745661
A_k	0.00559962661080374950184	0.00030524976709297350994	0.00000659212302607557677	0.00000004110769330349403
n=9				
x_k	5.55249614006128844323484	8.33015274677130967972971	11.84378583788996763814793	16.27925783138683968331861
A_k	0.00950151697527147620636	0.00075300838857460554096	0.00002825923349650162556	0.00000042493139849147913

高斯-拉盖尔求积公式的节点和系数(准确到小数点后23位)

	9	10	11	12
n=8				
x_k	26.37407189092734682844820			
A_k	0.00000000003290874030350			
n=9				
x_k	21.99658581197656914696381	29.92069701227467248827451		
A_k	0.00000000183956482398699	0.00000000000099118272196		

附录4

高斯-埃尔米特求积公式的节点和系数(准确到小数点后31位)

	1	2	3
n=0			
x_k	0.0000000000000000000000000000000		
A_k	1.7724538509055158819194275565678		
n=1			
x_k	+-0.7071067811865475727373109293694		
A_k	0.8862269254527581630043187033152		
n=2			
x_k	+-1.2247448713915889406678161321906	0.0000000000000000000000000000000	
A_k	0.2954089751509191841272183864930	1.1816359006036774026426883210660	
n=3			
x_k	+-1.6506801238857851110708452324615	+-0.5246476232752903534617416880792	
A_k	0.0813128354472449216272522676263	0.8049140900055128389212200090696	
n=4			
x_k	+-2.0201828704560842453474833746440	+-0.9585724646138183979715563509671	0.0000000000000000000000000000000
A_k	0.0199532420590461320730746308527	0.3936193231522412405709587801538	0.9453087204829417888873877018340
n=5			
x_k	+-2.3506049736744882849848181649576	+-1.3358490740136963470519049224094	+-0.4360774119276166205239064765919
A_k	0.0045300099055090104274712281551	0.1570673203228570846690104190201	0.7246295952243922977586976230668
n=6			
x_k	+-2.6519613568352360388757915643509	+-1.6735516287674729873913292976794	+-0.8162878828589642532520542772545
A_k	0.0009717812450994918211516493400	0.0545155828191264471560550930462	0.4256072526101282726962438118789
n=7			
x_k	+-2.9306374202572396114874209160916	+-1.9816567566958467327964399373740	+-1.1571937124467781554670864352374
A_k	0.0001996040722113777822564056885	0.0170779830074129709838182122894	0.2078023258148938012812578790544
n=8			
x_k	+-3.1909932017815370031144084350671	+-2.2665805845318427458323640166781	+-1.4685532892166690555058039535651
A_k	0.0000396069772632596552031092329	0.0049436242755369636664442012374	0.0884745273943759319434931853720
n=9			
x_k	+-3.4361591188377378358609348651953	+-2.5327316742327918852595303178532	+-1.7566836492998829655221015855204
A_k	0.0000076404328552326003352488867	0.0013436457467812033643450586595	0.0338743944554807732694179378540

附录4

高斯-埃尔米特求积公式的节点和系数(准确到小数点后31位)

	4	5	6
n=6			
x_k	0.0000000000000000000000000000000		
A_k	0.8102646175568073427797344265854		
n=7			
x_k	+-0.3811869902073222737826085904089		
A_k	0.6611470125582411538900373670912		
n=8			
x_k	+-0.7235510187528376713217426186020	0.0000000000000000000000000000000	
A_k	0.4326515590025555857423000816197	0.7202352156060508603374614722270	
n=9			
x_k	+-1.0366108297895149092937572277151	+-0.3429013272237045883983341809653	
A_k	0.2401386110823133801517315077944	0.6108626337353257884643653596867	