Gauss Laguerre

Prateek Bhardwaj (2020PHY1110) (20068567042)
Monu Chaurasiya (2020PHY1102) (20068567035)

S.G.T.B. Khalsa College, University of Delhi, Delhi-110007, India.

Practical Report Submitted to

Dr. Mamta

"32221401 - Mathematical Physics III"

Table of Contents

1 Theory

(a) Explain Laguerre Gauss Quadrature method for integration. What kind of integrals are evaluated by this method?

- The Gauss–Laguerre quadrature rule is a Gaussian quadrature over the interval $[a,b]$ with the weight function $\psi(x) = e^{-x}$.The general form is,

$$\int_0^{\infty} e^{-x} f(x)dx = \sum_{k=1}^{n} \lambda_k f(x_i)$$

  The nodes x_i are the zeroes of the Laguerre Polynomial,

$$L_n(x) = e^x \frac{d^n}{dx^n}(e^{-x}x^n)$$

  The weights $\lambda_k s$ are given by

$$\lambda_k = \frac{1}{x_k [L_n'(x_i)]^2} = \frac{x_k}{(n+1)^2 [L_{n+1}(x_k)]^2}$$

- Gauss–Laguerre quadrature method is used for approximating the value of integrals of the following kind:

$$\int_0^{+\infty} e^{-x} f(x)dx$$

(b)Write down the Laguerre differential equation and first five Laguerre polynomials.

- The second order Lagurre's Differential Equation is:

$$x\frac{d^2y}{dx^2} + (1-x)\frac{dy}{dx} + ky = 0$$

- The Rodrigues representation for the Laguerre polynomials is

$$L_n(x) = \frac{e^x}{n!}\frac{d^n}{dx^n}(e^{-x}x^n)$$

And first five Laguerre Polynomials are as follows:

$$L_0(x) = 1 \tag{1}$$

$$L_1(x) = -x+1 \tag{2}$$

$$L_2(x) = \frac{1}{2}(x^2-4x+2) \tag{3}$$

$$L_3(x) = \frac{1}{6}(-x^3+9x^2-18x+6) \tag{4}$$

$$L_4(x) = \frac{1}{24}(x^4-16x^3+72x^2-96x+24) \tag{5}$$

(c)Write down the recursion formulae and orthogonality conditions for these polynomials.

- The recurrence relations are:

  1. $L_{n+1}(x) = (2n+1-x)L_n(x) - n^2 L_{n-1}(x)$
  2. $nL_{n-1}(x) = nL'_{n-1}(x) - L'_n(x)$

- The Orthogonal Properties:

  The Laguerre polynomials do not themselves form an orthogonal set. However, we use a related set of functions as $\phi_n(x) = e^{-\frac{x}{2}}L_n(x)$ and $\phi_m(x) = e^{-\frac{x}{2}}L_m(x)$ to form an orthonormal set for the interval $0 \leq x \leq \infty$.

  The orthogonal properties of Laguerre's Polynomials are expressed as

$$\int_0^\infty \phi_n(x)\phi_m(x)dx = 0$$

$$\int_0^\infty [\phi_n(x)]^2 dx = (n!)^2$$

(d)Explicitly derive the 2-point quadrature formula for this method.

We have,

$$L_o(x) = 1$$

$$L_1(x) = -x+1$$

$$L_2(x) = \frac{1}{2}(x^2-4x+2)$$

Gauss-Lagurre Quadrature:

$$\int_0^{\infty} e^{-x} f(x)dx = \sum w_i f(x_i)$$

For 2-point Gauss Lagurre:

$$\int_0^{\infty} e^{-x} f(x)dx = w_1 f(x_1) + w_2 f(x_2)$$

where x_1 and x_2 are the abscissa and w_1 and w_2 are the weights.

The abscissa for n-point rule are the roots of the laguerre function of degree n.

We have,

$$L_2(x) = \frac{1}{2}(x^2-4x+2)$$

The roots of $L_2(x) = 0$ are the abscissa for 2-point Gauss-laguerre rule.

$$\frac{1}{2}(x^2-4x+2) = 0$$

$$\implies x^2-4x+2 = 0$$

$$x = \frac{-b \pm \sqrt{b^2-4ac}}{2a} \implies x = \frac{4 \pm \sqrt{16-8}}{2}$$

$$\implies 2 \pm \sqrt{2}$$

$$x_1 = 2+\sqrt{2}\,,\; x_2 = 2-\sqrt{2}$$

To find the w_1 and w_2 weights, we use $L_0(x)$ and $L_1(x)$ to find relationship eq^n.

Using $L_0(x) = 1$

$$\int_0^{\infty} e^{-x} f(x)dx = w_1 f(x_1) + w_2 f(x_2)$$

$$\int_0^{\infty} e^{-x}(1)dx = w_1 + w_2$$

$$[-e^{-x}]_0^{\infty} = w_1 + w_2$$

$$w_1 + w_2 = 1 \tag{6}$$

Using $L_1(x) = -x+1$

$$\int_0^\infty e^{-x}(-x+1)dx = w_1 f(x_1) + w_2 f(x_2)$$

$$\int_0^\infty e^{-x}(-x)dx + \int_0^\infty e^{-x}dx = w_1 f(2+\sqrt{2}) + w_2 f(2-\sqrt{2})$$

$$[xe^{-x} + e^{-x}]_0^\infty + [-e^{-x}]_0^\infty = W_1(-\sqrt{2}-1) + W_2(\sqrt{2}-1)$$

$$w_1(-\sqrt{2}-1) + w_2(\sqrt{2}-1) = 0 \tag{7}$$

Subtracting $(\sqrt{2}-1)$*eq.6 from eq.7 we get,

$$2w_1\sqrt{2} = \sqrt{2}-1$$

$$w_1 = \frac{\sqrt{2}-1}{2\sqrt{2}}$$

$$w_1 \implies \frac{2-\sqrt{2}}{4}$$

And so,

$$w_2 = \frac{2+\sqrt{2}}{4}$$

2-point Gauss-laguerre quadrature integration rule is given below.

$$\int_0^\infty e^{-x} f(x)dx = \frac{2-\sqrt{2}}{4} f(2+\sqrt{2}) + \frac{2+\sqrt{2}}{4} f(2-\sqrt{2})$$

2 Algorithm

Algorithm 1 Laguerre Polynomials

function MYLAGUQUAD(f, n)

▷ *MyLaguQuad function takes the parameter f and n*

▷ *f: function , n: no.of points*

▷ *Inbuilt function np.polynomial.laguerre.laggauss(n) takes n as a parameter and it returns two arrays of weights and points.*

Integral=0

xi,wi=np.polynomial.laguerre.laggauss(n)

▷ *n-point gauss-laguerre quadrature Integration formula is the sum of the product of weight and points*

for (**do**Xi,Wi) in zip (xi,wi):

Integral+=Wi*f(Xi)

return Integral

3 Programming

```
'''
Name-Monu Chaurasiya
Roll No. - 2020PHY1102


Partner -
Name-Prateek Bhardwaj
Roll No. - 2020PHY1110
'''


import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import math
from scipy import integrate
from sympy import *
from sympy import simplify
import scipy
from MyIntegration import MySimp
from MyIntegration import MyLaguQuad

#(c)
print("Name-Prateek Bhardwaj \n Roll No. - 2020PHY1110")
def new_simp(f,a,R0,R_max,tol):
    #j=MySimp(f,a,R0,2,key1=True,N_max=10**8,key2=True,tol=0.1e-5)
    lis=[]
    R_a=[]
    w=0
    a_a=[]
    while R0<=R_max:
        j=MySimp(f,a,R0,2,key1=True,N_max=10**8,key2=True,tol=0.1e-5)
        #j=MySimp(f,a,R0,2,key1=False)
        lis.append(j[0])
        R_a.append(R0)
        a_a.append(a)
        if len(lis)>=2:
            if lis[-1]<=0.1e-5:
                err=abs(lis[-1]-lis[-2])
            else:
                err=abs((lis[-1]-lis[-2])/lis[-1])
            if err<=tol:
                w=1
                break
            else:
                pass
        R0=10*R0
    if w==0:
            s=("R_max reached without achieving required tolerance")
    elif w==1:
             s="Given tolerance achieved with R=",R_a[-1]
    return lis[-1],R_a[-1],s,lis,R_a,a_a      #returning integral,
```

```
 number of intervals and message


#Q3(b)
#(i)
n=2
f_x=["1","x","x**2","x**3","x**4","x**5"]
Calc=[]
Exact=[1,1,2,6,24,120]
for i in range(0,6):
    print(i+1,"th function")
    f=eval("lambda x:"+input("Enter the value of the FUNCTION F(x): ")
   )
    Calc.append(MyLaguQuad(f, n))
data={"f(x)":f_x,"Calculated":Calc,"Exact":Exact}
print()
print("*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*")
print()
print("METHOD USED : Gauss Laguerre quadrature (TWO POINT)")
print(pd.DataFrame(data))
print()
print("*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*")
print()


n=4
f_x=["1","x","x**2","x**3","x**4","x**5","x**6","x**7","x**8","x**9"]
Calc=[]
Exact=[1,1,2,6,24,120,720,5040,40320,362880]
for i in range(0,10):
    print(i+1,"th function")
    f=eval("lambda x:"+input("Enter the value of the FUNCTION F(x): ")
   )
    Calc.append(MyLaguQuad(f, n))

data={"f(x)":f_x,"Calculated":Calc,"Exact":Exact}
print()
print("*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*")
print()
print("METHOD USED : Gauss Laguerre quadrature (FOUR POINT)")
print(pd.DataFrame(data))
print()
print("*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*")
print()


#(ii)
I1_a=[]
I2_a=[]
n_a=[2,4,8,16,32,64,128]
f1=lambda x : 1/(1+x**2)
f2=lambda x : np.exp(x)/(1+x**2)
for n in n_a:
    I1_a.append(MyLaguQuad(f1, n))
    I2_a.append(MyLaguQuad(f2, n))
```

```

DataOut = np.column_stack((n_a,I1_a,I2_a))
np.savetxt("quad-lag-1092", DataOut,delimiter=',')

print("#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#")
print()
data={"n":n_a,"I1":I1_a,"I2":I2_a}
print(pd.DataFrame(data))


q1=np.array([0.621449624235]*len(I1_a))
q2=np.array([np.pi/2]*len(I2_a))
fig, (ax1, ax2) = plt.subplots(1, 2)
fig.suptitle('GAUSS LAGURRE QUADRATURE')
ax1.plot(n_a,I1_a,marker="*",label="Calculated",linestyle='dashed')
ax1.plot(n_a,q1,label="EXACT",c="red",linewidth=1)
ax2.plot(n_a,I2_a,marker="*",label="Calculated",linestyle='dashed')
ax2.plot(n_a,q2,label="EXACT",c="red",linewidth=1)
ax1.grid()
ax1.legend()
ax1.set(xlabel="N",ylabel="Integral",title="Integral I1 ")
ax2.set(xlabel="N",ylabel="Integral",title="Integral I2 ")
ax2.grid()
ax2.legend()
plt.show()


a=0
R0=10
R_max=10**6
tol=0.1e-7
F1=lambda x : np.exp(-1*x)/(1+x**2)
F2=lambda x : 1/(1+x**2)
s1=new_simp(F1,a,R0,R_max,tol)
s2=new_simp(F2,a,R0,R_max,tol)

#d
print()
print("#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#")

print()
print("RESULTS USING SIMPSON METHOD")
print("Tolerance for MYSimp defined in MyIntegration Module = 0.1e-5")
print("Tolerance for the value of Integral with respect to value of b(upper limit) = 0.1e-7")
print()
data={"a(lower limit)":s1[5],"b(upper limit)":s1[4],"Integral I1":s1[3]}
print(pd.DataFrame(data))
print()
data={"a(lower limit)":s2[5],"b(upper limit)":s2[4],"Integral I2":s2[3]}
print(pd.DataFrame(data))
```

```
print()
print("#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#
-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#")
print()

q1=np.array([0.621449624235]*len(s1[3]))
q2=np.array([np.pi/2]*len(s2[3]))
fig, (ax1, ax2) = plt.subplots(1, 2)
fig.suptitle('SIMPSON METHOD (TOLERANCE = 0.1e-7)')
ax1.plot(s1[4],s1[3],marker="*",label="I1 using SIMPSON",linestyle='dashed')
ax1.plot(s1[4],q1,label="EXACT",c="red",linewidth=1)
ax2.plot(s2[4],s2[3],marker="*",label="I2 using SIMPSON",linestyle='dashed')
ax2.plot(s2[4],q2,label="EXACT",c="red",linewidth=1)
ax1.grid()
ax1.legend()
ax1.set(xlabel="b (upper limit)",ylabel="Integral",title="Integral I1 calculated using SIMPSON")
ax2.set(xlabel="b (upper limit)",ylabel="Integral",title="Integral I2 calculated using SIMPSON")
ax2.grid()
ax2.legend()
plt.show()
```

4 Result and Discussion

```
*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*

METHOD USED : Gauss Laguerre quadrature (TWO POINT)
   f(x)   Calculated  Exact
0     1          1.0      1
1     x          1.0      1
2  x**2          2.0      2
3  x**3          6.0      6
4  x**4         20.0     24
5  x**5         68.0    120

*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
```

```
*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*

METHOD USED : Gauss Laguerre quadrature (FOUR POINT)
   f(x)  Calculated    Exact
0     1         1.0        1
1     x         1.0        1
2  x**2         2.0        2
3  x**3         6.0        6
4  x**4        24.0       24
5  x**5       120.0      120
6  x**6       720.0      720
7  x**7      5040.0     5040
8  x**8     39744.0    40320
9  x**9    339264.0   362880
```

```
#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#

     n        I1        I2
0    2  0.647059  1.493257
1    4  0.636427  1.501190
2    8  0.620075  1.533760
3   16  0.621507  1.553738
4   32  0.621449  1.562483
5   64  0.621450  1.566725
6  128  0.621450  1.568789

#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#-#
```

For gauss-laguerre quadrature we verify the results for n=2 and n=4 and we can se it shows correct value upto 2n-1 degree and that we can see clearly in output table.

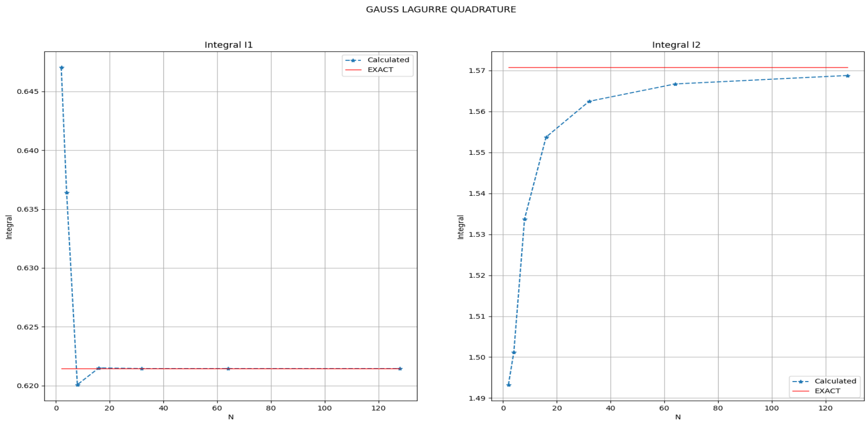

Figure 1: Integral value vs No. of intervals

From the graph we can see that the integral value I_1 approaches to true value very rapidly i.e it converges for increasing value of n. The graph overlap over the exact value line for n=20.

For integral value I_2 graph we can see that the graph approaching towards true value as n increases but it doesn't overlap over the exact value line and the integral I_2 graph line shows exponential growth due to presence of exponential term in the integrand.

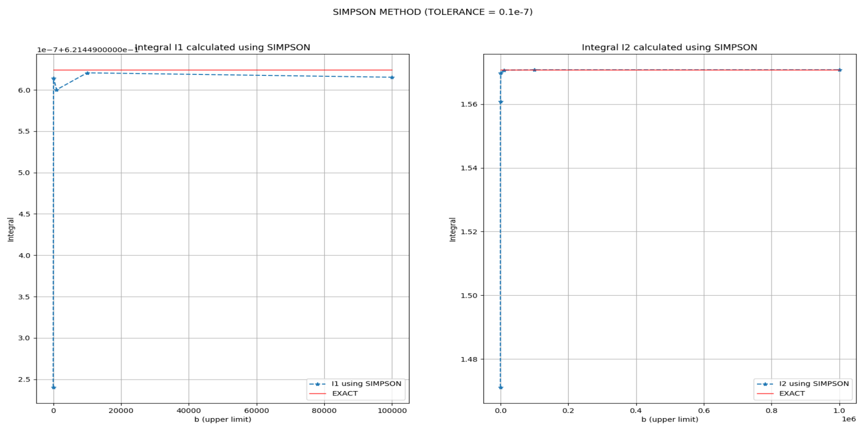

Figure 2: Integral Value vs The limit

It is the graph between integral value vs the limits that we are using instead of infinity to check for a Given tolerance what is the limit that gives us a exact or true value or we can say that we are finding a value of limit from there the integral start converging.

For I1 we can see that the integral approaching towards exact value for b=1000 but after that it start diverging . And I2 converges for the value of b=0.1*10