Mathematical Physics III

Lab Assignment #4

Ishmeet Singh

College Roll No	**:**	2020PHY1221
University Roll No	**:**	20068567051
Unique Paper Code	**:**	32221401
Paper Title	**:**	Mathematical physics III Lab
Course and Semester	**:**	B.Sc.(Hons.) Physics, Sem IV
Due Date	**:**	10 March 2022
Date of Submission	**:**	10 March 2022
Lab Report File Name	**:**	2020PHY1221_A4.pdf
Submitted to	**:**	Dr. Mamta Dahiya

Team Member :

Name	**:**	Sarthak Jain
Roll Number	**:**	2020PHY1201

Shri Guru Tegh Bahadur Khalsa College, University of Delhi
New Delhi-110007, India.

Contents

1 Theory

1.1 Laguerre Gauss Quadrature method for Integration

Laguerre Gauss quadrature is a Gaussian quadrature over the interval $[0, \infty)$ with weighting function $W(x) = e^{(-x)}$. It fits all polynomials of degree $2m - 1$ exactly.
The method is used for evaluating the integrals of the following kind:

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

1.2 Laguerre differential equation

The Laguerre differential equation is given by:

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

The first five Laguerre Polynomials are:

- $L_0 = 1$
- $L_1 = -x + 1$
- $L_2 = \frac{1}{2}(x^2 - 4x + 2)$
- $L_3 = \frac{1}{6}(-x^3 + 9x^2 - 18x + 6)$
- $L_4 = \frac{1}{24}(x^4 - 16x^3 + 72x^2 - 96x + 24)$

1.3 Recurrence Relations

Recurrence Relation I:

$$(n+1)L_{n+1}(x) = (2n + 1 - x)L_n(x) - nL_{n-1}(x)$$

Recurrence Relation II:

$$xL_n^{'}(x) = nL_n(x) - nL_{n-1}(x)$$

1.4 Orthogonal Properties

The orthogonal properties of Laguerre Polynomials are expressed as,

$$\int_0^{\infty} e^{-x} L_m(x) L_n(x)\, dx = \begin{cases} 0, & m \neq n \\ (n!)^2, & m = n \end{cases}$$

2 Algorithm

Algorithm 1 Algorithm for n-point Gauss Laguerre Quadrature Method

```
function MyLaguQuad(expression, n)    ▷ Function to perform numerical integration
                                        using n-point Gauss Laguerre quadrature method
    x = var('x')                                                                   ▷
                                                        expr = sympify(expression)
    func = lambdify(x,expr)
    xvals,weights = np.polynomial.laguerre.laggauss(n)                             ▷
    w_fx = [weights[i]*func(xvals[i])
    for i in range(len(weights)) do:                                               ▷
        result = sum(w_fx)
    end for
    return result
end function
```

3 Programming

3.1 2020PHY1221_A4.py

```

import numpy as np
import matplotlib.pyplot as plt
import MyIntegration as mi
import pandas as pd

"Name: Ishmeet Singh, 2020PHY1221"
"Partner Name: Sarthak Jain, 2020PHY1201"

print("My Roll No.: 2020PHY1221")

def integral_simp(I1_exact,I2_exact):
    i = 2
    I1_simp = []
    I2_simp = []
    count1_simp = []
    count2_simp = []
    I1_exact_S = []
    I2_exact_S = []

    while True:
        if abs(I1_exact - mi.MySimp("exp(-x)/(1+x**2)",1000,0,i)) <=
    10**(-2):
            I1_simp.append(mi.MySimp("exp(-x)/(1+x**2)",1000,0,i))
            I1_exact_S.append(I1_exact)
            count1_simp.append(i)
            break
        else:
            I1_simp.append(mi.MySimp("exp(-x)/(1+x**2)",1000,0,i))
            I1_exact_S.append(I1_exact)
            count1_simp.append(i)
            i += 2

    i = 2

    while True:
        if abs(I2_exact - mi.MySimp("1/(1+x**2)",1000,0,i)) <= 10**(-2):
            I2_simp.append(mi.MySimp("1/(1+x**2)",1000,0,i))
```

```python
                I2_exact_S.append(I2_exact)
                count2_simp.append(i)
                break
            else:
                I2_simp.append(mi.MySimp("1/(1+x**2)",1000,0,i))
                I2_exact_S.append(I2_exact)
                count2_simp.append(i)
                i += 2

    return I1_simp,I2_simp,count1_simp,count2_simp,I1_exact_S,I2_exact_S

def graph(I1,I2,n,I1_simp,I2_simp,count1_simp,count2_simp,I1_exact_S,
    I2_exact_S):
    fig1,ax1 = plt.subplots(1, 2)
    fig2,ax2 = plt.subplots(1, 2)
    ax1[0].plot(n,I1,label = "MyLagQuad")
    ax1[0].plot(n,I1_exact_LL,label = "Analytic Value")
    ax1[1].plot(count1_simp,I1_simp,label = "MySimp")
    ax1[1].plot(count1_simp,I1_exact_S,label = "Analytic Value")
    ax2[0].plot(n,I2,label = "MyLagQuad")
    ax2[0].plot(n,I2_exact_LL,label = "Analytic Value")
    ax2[1].plot(count2_simp,I2_simp,label = "MySimp")
    ax2[1].plot(count2_simp,I2_exact_S,label = "Analytic Value")
    for i in range(2):
        if i == 0:
            ax1[i].set(xlabel = "Nodal Points (n)",ylabel = "Value of
    Integration (I)",title = "Gauss Laguerre Quadrature")
            ax2[i].set(xlabel = "Nodal Points (n)",ylabel = "Value of
    Integration (I)",title = "Gauss Laguerre Quadrature")
        elif i == 1:
            ax1[i].set(xlabel = "Nodal Points (n)",ylabel = "Value of
    Integration (I)",title = "Simpson 1/3 Method")
            ax2[i].set(xlabel = "Nodal Points (n)",ylabel = "Value of
    Integration (I)",title = "Simpson 1/3 Method")
        ax1[i].grid(ls = "--")
        ax2[i].grid(ls = "--")
        ax1[i].legend()
        ax2[i].legend()
    fig1.suptitle("INTEGRAL 1")
    fig2.suptitle("INTEGRAL 2")
    plt.show()

```

```python
if __name__ == "__main__":

    # PART B I

    count = 0
    func = []
    Exact=[1,1,2,6,24,120,720,5040,40320,362880]

    for count in range(len(Exact)):
        f = input("\nEnter Function: ")
        func.append(f)
        if count < (len(Exact) - 1):
            ans = input("Do you want to enter more function (Y/N) ?\t")
            if ans == "N" or ans == "n":
                break

    for j,m in zip(func,Exact):
        for k in range(2,6,2):
            print("\nValue of integration of",j,"for n =",k,"is: ",mi.MyLaguQuad(j,k))
        print("\nExact Value of integration of",j,"is: ",m)
        print("--------------------------------------------------------------")


    # PART B II

    I1 = []
    I2 = []
    I1_exact = 0.621449624235813
    I2_exact = 1.570796326794897
    I1_exact_LL = []
    I2_exact_LL = []
    n = []

    for i in range(2,130,2):
        n.append(i)
        I1_exact_LL.append(I1_exact)
        I2_exact_LL.append(I2_exact)
        i1 = mi.MyLaguQuad("1/(1+x**2)",i)
        I1.append(i1)
        i2 = mi.MyLaguQuad("exp(x)/(1+x**2)",i)
```

```python
        I2.append(i2)

    data1 =  np.column_stack([n,I1,I2])
    file1 = np.savetxt("quad-lag-1221.txt",data1,header = ("n,I1,I2"))

    df1 =  pd.DataFrame({"n": n, "I1": I1, "I2": I2})
    print("\nGAUSS LAGUERRE QUADRATURE:\n",df1)

    # PART B III & IV

    I1_simp,I2_simp,count1_simp,count2_simp,I1_exact_S,I2_exact_S = 
   integral_simp(I1_exact,I2_exact)

    df2 =  pd.DataFrame({"n": count1_simp, "I1": I1_simp})
    print("\nTOLERNACE LIMIT = 10**(-2)")
    print("\nSIMPSON FOR INTEGRAL 1:\n",df2)
    data2 =  np.column_stack([count1_simp,I1_simp])
    file2 = np.savetxt("Simpson-Integral_1-1221.txt",data2,header = ("n,I1
   "))

    print("\n
   -----------------------------------------------------------------")

    df3 =  pd.DataFrame({"n": count2_simp, "I1": I2_simp})
    print("\nTOLERNACE LIMIT = 10**(-2)")
    print("\nSIMPSON FOR INTEGRAL 1:\n",df3)
    data3 =  np.column_stack([count2_simp,I2_simp])
    file3 = np.savetxt("Simpson-Integral_2-1221.txt",data3,header = ("n,I2
   "))

    graph(I1,I2,n,I1_simp,I2_simp,count1_simp,count2_simp,I1_exact_S,
   I2_exact_S)
```

4 Results and Discussion

According to Part b) i. of the assignment, we have verified in our code that n-point quadrature formula gives exact result when $f(x)$ is a polynomial of order $2n-1$ taking $n=2$ and $n=4$

Also, according to Part b) ii. of the assignment, we were asked to compute numerically the integration values for two different functions, first, by using n - point Gauss Laguerre Quadrature method of integration and second, by using Simpson $\frac{1}{3}$ method of integration.

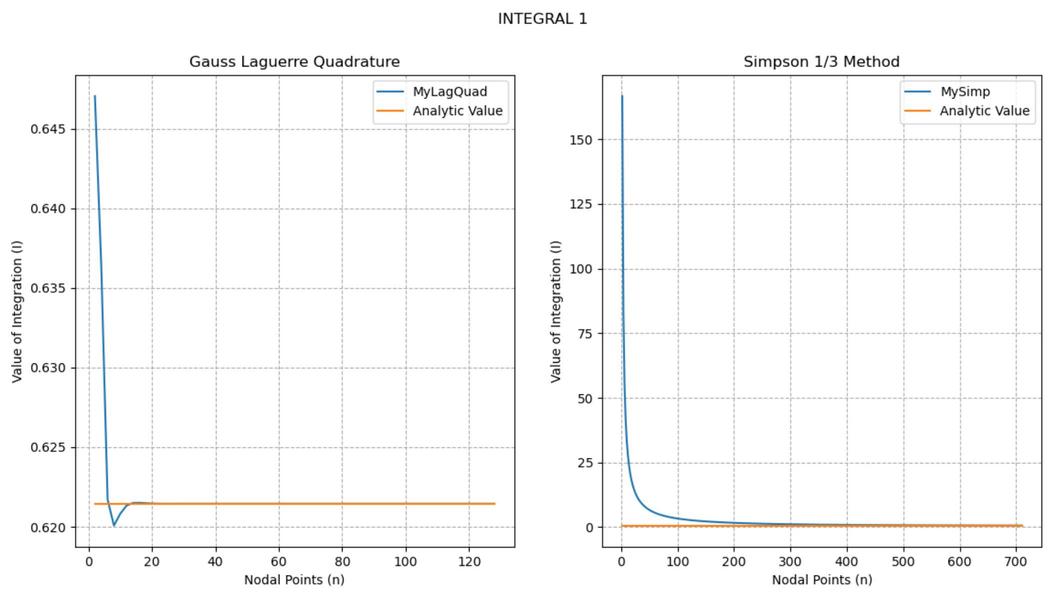

Figure 1: Comparing Gauss Laguerre Quadrature with Simpson $\frac{1}{3}$ for First Integral

In the above graph (1), we have shown the comparison between the two numerical methods while simultaneously comparing each of them with the analytical values. It can be inferred from the above graph (1) that the n - point Gauss Laguerre quadrature method starts approaching the analytical value for fewer nodal points (≈ 14) while for the Simpson $\frac{1}{3}$ method minimum nodal points required for the numerical value to converge with the analytical values were around ≈ 200.

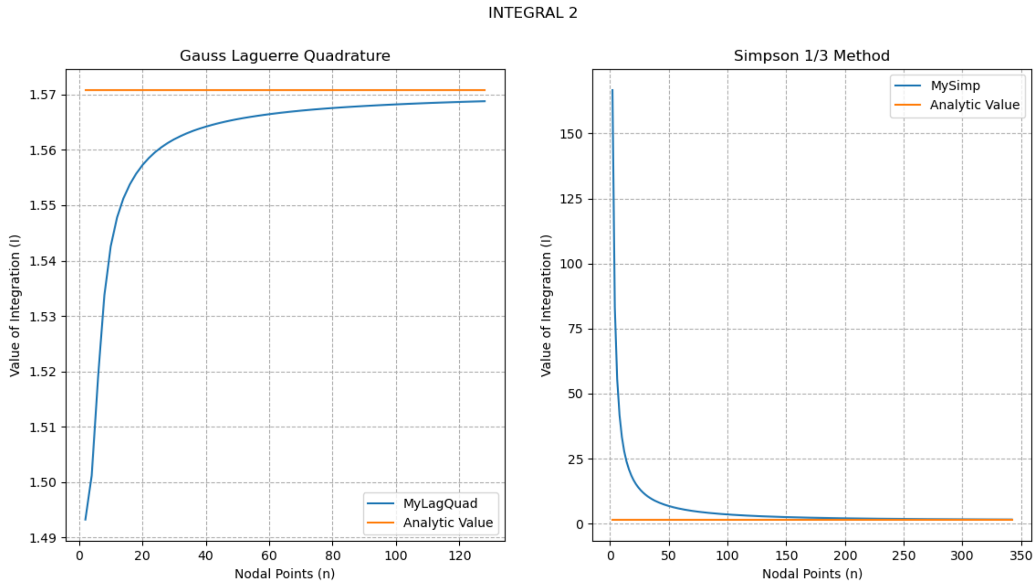

Figure 2: Comparing Gauss Laguerre Quadrature with Simpson $\frac{1}{3}$ for Second Integral

For the second integral, however, it can be seen in the above graph (2) that the numerical value computed by the n - point Gauss Laguerre quadrature method does not seem to approach the analytical value even for $n = 128$, whereas for the Simpson $\frac{1}{3}$ method, the numerical value does start to approach more for fewer nodal points than the n - point Gauss Laguerre quadrature method.

This is purely due to the fact that for the Simpson $\frac{1}{3}$ method of integration, we picked an upper limit of just 1000, which is in no way comparable to $+\infty$. Therefore, the Simpson $\frac{1}{3}$ method seems to approximate the function better than the n - point Gauss Laguerre quadrature method for this particular case but in general, we believe that an accurate comparison between the two can not be made.