The Mutually Orthogonal Legendre Polynomials

Shashvat Jain
(2020PHY1114)(20068567054)

February 22, 2022

Lab Report for Assignment No. 2b

College Roll No :	2020PHY1114
University Roll NoName:	20068567054
Unique Paper Code:	32221401
Paper Title:	Mathematical physics III Lab
Course and Semester :	B.Sc.(H) Physics Sem IV
Due Date:	Jan 15,2022
Date of Submission:	Jan 14,2022
Lab Report File Name:	mp3A2b_2020PHY1114.pdf
Partner's Name:	-
Partner's College Roll No.:	-

1 Theory

Polynomials or Polynomial functions $P_1(x) : \mathbb{R} \to \mathbb{R}$ and $P_2(x) : \mathbb{R} \to \mathbb{R}$ are called **orthogonal** in the closed interval $[a, b]$ if and only if $\int_a^b P_1 P_2 dx = 0$. The idea is analogous to the orthogonality of finite-dimensional vectors where we define orthogonality of two vectors by a null valued dot product. Since a polynomial function is a real-valued mapping into $\mathbb{R}$ it is a part of the vector space of all functions from $\mathbb{R} \to \mathbb{R}$. We define an operation called the standard **inner-product** just like the dot-product which when found to be zero declares the orthogonality of the constituent elements of the space.

The Orthogonality condition for Legendre polynomials P_k where k give the order of the polynomial is given by,

$$\int_{-1}^{1} P_m(x)P_n(x)\,dx = \left\{ \begin{array}{ll} \frac{2}{2n+1} & \text{for } m = n \\ 0 & \text{for } m \neq n \end{array} \right\} = \frac{2}{2n+1}\delta_{mn}, \tag{1}$$

Given starting basis of linearly independent functions $B = \{1, x, x^2, x^3, x^4, \dots\}$, let us take the constant polynomial function $L_0(x) = 1$ to be the first element in the newly made basis L orthogonal in the interval $[-1, 1]$. Now,

$$\begin{aligned} L_1 &= x - \frac{< x, 1 >}{< 1, 1 >}1 \\ &= x - \frac{\int_{-1}^{1} x dx}{\int_{-1}^{1} dx} = x \end{aligned} \tag{2}$$

$$\begin{aligned} L_2 &= x^2 - \frac{< x, x^2 >}{< x, x >}x - \frac{< x^2, 1 >}{< 1, 1 >}1 \\ &= x^2 - \frac{\int_{-1}^{1} x^3 dx}{\int_{-1}^{1} x^2 dx}x - \frac{\int_{-1}^{1} x^2 dx}{\int_{-1}^{1} dx} = x^2 - \frac{1}{3} \end{aligned} \tag{3}$$

Therefore $L = \{1, x, x^2 - 1/3\}$. Now since $\frac{2}{3}L_3$ is a scaled version of L_3 it also is orthogonal to all other elements of L, the orthogonal basis $L' = \left\{1, x, \frac{1}{2}(3x^2 - 1)\right\}$ have all its elements as solutions to the Legendre differential equation.

By Definition,

$$f(x) = \sum_{n=0}^{\infty} C_n P_n(x) \tag{4}$$

To find a coefficient C_k Multiplying both sides by P_k and integrating both sides with respect to x from -1 to 1.

$$\begin{aligned} &\int_{-1}^{1} P_k(x)f(x)dx = \int_{-1}^{1} P_k \sum_{n=0}^{\infty} C_n P_n dx \\ \implies &\int_{-1}^{1} P_k(x)f(x)dx = \sum_{n=0}^{\infty} \int_{-1}^{1} P_k(x)C_nP_n(x)dx \\ \implies &\int_{-1}^{1} P_k(x)f(x)dx = \sum_{n=0}^{\infty} \int_{-1}^{1} P_k(x)C_nP_n(x)dx \\ \implies &\int_{-1}^{1} P_k(x)f(x)dx = \int_{-1}^{1} C_0P_k(x)P_0(x)dx + \int_{-1}^{1} C_1P_k(x)P_1(x)dx + \dots + \int_{-1}^{1} C_kP_k(x)P_k(x)dx + \dots \end{aligned}$$

From 1 all terms except one with $n = k$ go to zero, leaving behind,

$$\int_{-1}^{1} P_k(x)f(x)dx = C_k\frac{2}{2k+1}$$

$$\boxed{C_k = \frac{2k+1}{2}\int_{-1}^{1} P_k(x)f(x)dx} \tag{5}$$

Since all Legendre polynomials are orthogonal to each other, The set of all Legendre polynomials of order less than equal to $n \in N$, since they are linearlty independent, span the function space of all polynomials of order less than equal to n. Therefore the series expansion of any polynomial function of order n will have atmost $n(\{P_n, P_{n-1}, P_{n-2}\dots P_0\}) = n+1$ terms.

(α)Using the above relation 5 we can deduce the coefficients which come out to be:

$$\begin{aligned}C_0 &= \frac{1}{2}\int_{-1}^{1}(1)(2+3x+2x^4)dx\\ &= \frac{1}{2}\left(\frac{24}{5}\right)\\ &= \frac{12}{5}\end{aligned}$$

$$\begin{aligned}C_1 &= \frac{3}{2}\int_{-1}^{1}(x)(2+3x+2x^4)dx\\ &= \frac{3}{2}(2)\\ &= 3\end{aligned}$$

$$\begin{aligned}C_2 &= \frac{5}{2}\int_{-1}^{1}\frac{1}{2}(3x^2-1)(2+3x+2x^4)dx\\ &= \frac{5}{2}\left(\frac{16}{35}\right)\\ &= \frac{8}{7}\end{aligned}$$

$$\begin{aligned}C_3 &= \frac{7}{2}\int_{-1}^{1}\frac{1}{2}(5x^3-3x)(2+3x+2x^4)dx\\ &= \frac{7}{2}(0) = 0\end{aligned}$$

$$\begin{aligned}C_4 &= \frac{9}{2}\int_{-1}^{1}\frac{1}{8}(35x^4-30x^2+3)(2+3x+2x^4)dx\\ &= \frac{9}{2}\left(\frac{32}{315}\right)\\ &= \frac{16}{35}\end{aligned}$$

$$f(x) = \left(\frac{12}{5}\right)P_0 + (3)P_1 + \left(\frac{8}{7}\right)P_2 + (0)P_3 + \left(\frac{16}{35}\right)P_4$$

(β) We have to find the first 5 terms of the expansion so,

$$\begin{aligned}C_0 &= \frac{1}{2}\int_{-1}^{1}(1)(\cos(x)\sin(x))dx\\ &= \frac{1}{2}(0)\\ &= 0\end{aligned}$$

$$\begin{aligned}C_1 &= \frac{3}{2}\int_{-1}^{1}(x)(\cos(x)\sin(x))dx\\ &= \frac{3}{2}(0.43540)\\ &= 0.6531\end{aligned}$$

$$\begin{aligned}C_2 &= \frac{5}{2}\int_{-1}^{1}\frac{1}{2}(3x^2-1)(cos(x)sin(x))dx\\ &= \frac{5}{2}(0)\end{aligned}$$

$$
\begin{aligned}
C_3 &= \frac{7}{2}\int_{-1}^{1}\frac{1}{2}(5x^3-3x)(cos(x)sin(x))dx\\
&= \frac{7}{2}(-0.060722)\\
&= -0.212527
\end{aligned}
$$

$$
\begin{aligned}
C_4 &= \frac{9}{2}\int_{-1}^{1}\frac{1}{8}(35x^4-30x^2+3)(cos(x)sin(x))dx\\
&= \frac{9}{2}(0)\\
&= 0
\end{aligned}
$$

$$f(x) = (0)P_0 + (0.6531)P_1 + (0)P_2 + (-0.212527)P_3 + (0)P_4$$

2 Analysis

NOTE: f_n is the notation used for the Legendre series approximate of f calculated upto first n terms or the n-th partial sum of the series $\sum_{n=0}^{\infty} C_n P_n$

For problem 2(a):
The first array in figure represents the non-zero coefficients of the series corresponding to the order of polynomials in the second array. [2.43.1.142857140.45714286][0124] i.e. $C_0 = 2.4, C_1 = 3, C_2 = 1.14285714$

For problem 2(b):
Max difference between values of $f_2 - f_1$: 2.0517636768997254
RMS ERROR between values of $f_2 - f_1$: 0.17091050427990304
Max difference between values of $f_3 - f_2$: 4.440892098500626e-16
RMS ERROR between values of $f_3 - f_2$: 2.9123522237328004e-17
Accuracy of 6 significant digits reached with number of terms equal to: 2

```
planck@PC:~/Desktop/mp-3/integral$ python3 A2b-2020PHY1114.py
[2.4        3.         1.14285714 0.45714286] [0 1 2 4]
Max difference between values of f_2-f_1: 2.0517636768997254
RMS ERROR between values of f_2-f_1: 0.17091050427990304
Max difference between values of f_3-f_2: 4.440892098500626e-16
RMS ERROR between values of f_3-f_2: 2.9123522237328004e-17
Accuracy of 6 significant digits reached with number of terms equal to: 2
```

From the first subplot for $f(x) = 2x^4 + 3x + 2$ we can clearly see that f_5 overlaps the graph of the true values of f, hence representing the function f exactly, verified using zero RMSE in the domain for 50 nodes where the series was calculated. Therefore we verified that a polynomial of order k can be represented exactly as a weighted sum of Legendre polynomials of order $\leq k$ (k+1 terms.)
From the second plot for $f(x) = \sin(x)\cos(x)$ we see that since the function is not a polynomial function, the approximation of f using legendre polynomials upto first 8 terms produces a very small error and 10 terms produce even smaller values of RMS, this trend is justified as it is in agreement with the pri-ori that f is exactly represented by the weighted sum of Legendre polynomials as no. of terms go to infinity.

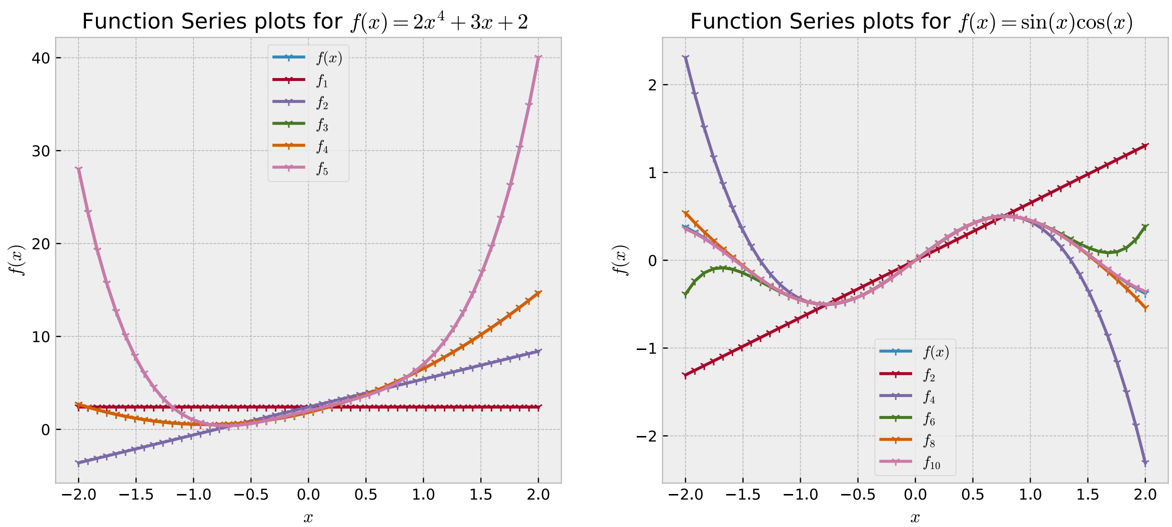

Figure 1: Functions obtained as a linear combination of Legendre polynomials plotted in the range $[-2, 2]$ for different number of terms.

3 Appendix

3.1 Code

```
from Myintegration import *
from scipy.special import legendre
from scipy.special import eval_legendre

def inp(f1,f2,a,b,n):
    return(MyLegQuadrature(lambda x: f1(x)*f2(x),a,b,n))

def legcoeff(func,n):
    n_arr = np.arange(0,n)
    coeffs = np.vectorize(lambda n: (2*n+1)/2 * inp(legendre(n),func,-1,1,6))
    return(coeffs(n_arr))

def fourier_leg(f,n):
    coeff = legcoeff(f,n)
    n_arr = np.arange(0,len(coeff))
    return(np.vectorize(lambda x: eval_legendre(n_arr,x).dot(coeff)))

if __name__ == "__main__":
    print([f"{d:.8g}" for d in legcoeff(lambda x: 2*x**4 + 3*x + 2,5,8)])
```

```
from fourier_legendre import *
import numpy as np
from scipy.special import eval_legendre
import matplotlib.pyplot as plt

def main():
    part1 = lambda x: 2*x**4 + 3*x + 2
    part1_coefs = np.round_(legcoeff(part1,5),14)
    print(part1_coefs[part1_coefs!=0],np.where(part1_coefs!=0)[0])

    part2 = lambda x : np.cos(x)*np.sin(x)
    x_space = np.linspace(-np.pi,np.pi)

    leg_apr_f = fourier_leg(part2,1)
    y_apr_0 = leg_apr_f(x_space)
```

```
    for n_i in np.arange(2,10000):
        leg_apr_f = fourier_leg(part2,n_i)
        y_apr_1 = leg_apr_f(x_space)
        err = max(abs(y_apr_1-y_apr_0))
        rms = np.sqrt(np.sum((y_apr_1-y_apr_0)**2))/y_apr_0.shape[0]
        print(f"Max difference between values of f_{n_i}-f_{n_i-1}:",err)
        print(f"RMS ERROR between values of f_{n_i}-f_{n_i-1}:",rms)
        if err <= 0.5/10**6 * np.max(np.abs(y_apr_1)):
            print("Accuracy of 6 significant digits reached with number of terms equal to:",n_i-1)
            break
        y_apr_0 = y_apr_1.copy()


    x_space = np.linspace(-2,2)
    fig,(ax1,ax2) = plt.subplots(1,2)
    nplt1 = np.arange(1,6)
    ax1.plot(x_space,part1(x_space),marker='1',label="$f(x)$")
    for n_i in nplt1:
        leg_apr = fourier_leg(part1,n_i)
        ax1.plot(x_space,leg_apr(x_space),label=f"$f_{n_i}$",marker="1")
    ax1.set_title("Function Series plots for $f(x) =2x^4 + 3x + 2 $ ")
    ax1.set_xlabel("$x$")
    ax1.set_ylabel("$f(x)$")
    nplt2 = np.arange(2,12,2)
    ax2.plot(x_space,part2(x_space),marker='1',label="$f(x)$")
    for n_i in nplt2:
        leg_apr = fourier_leg(part2,n_i)
        ax2.plot(x_space,leg_apr(x_space),label="$f_{"+f"{n_i}"+"}$",marker="1")
    ax2.set_title("Function Series plots for $f(x) =\sin(x)\cos(x)$ ")
    ax2.set_xlabel("$x$")
    ax2.set_ylabel("$f(x)$")
    ax1.legend()
    ax2.legend()
    plt.show()


if __name__ =="__main__":
    plt.style.use("bmh")
    from matplotlib import use
    use("WebAgg")
    main()
```