EE2703 : Applied Programming Lab
Assignment 4
Report

Kesava Aruna Prakash R L
EE20B061

February 26, 2022

Abstract

This week's assignment focuses on finding the Fourier coefficients of periodic functions using **integration** and **'Least Squares'** approach and approximating the Fourier sum to the actual values of the function.

1 Fourier Coefficients

Fourier coefficients are calculated according the formulae:

$$a_0 = \frac{1}{2\pi}\int_0^{2\pi} f(x)dx$$
$$a_n = \frac{1}{\pi}\int_0^{2\pi} f(x)cos(nx)dx$$
$$b_n = \frac{1}{\pi}\int_0^{2\pi} f(x)sin(nx)dx$$

An approximation to the actual function f(x) by using the coefficients $a_0, a_1, a_2.........a_{25}$ and $b_1, b_2, b_3, ...b_{25}$ and it is calculated according to :

$$f(x) \approx a_0 + \sum_{n=1}^{25} a_n + \sum_{n=1}^{25} b_n$$

2 Plots of the functions

The functions chosen for this analysis are e^x and $cos(cos(x))$.From now onwards, $f_1(x)$ will be used instead of e^x and $f_2(x)$ instead of $cos(cos(x))$.
The functions for calculating output for vector inputs:

```
def f1(input):
    return np.exp(input)
def f1p(input):
    return np.exp(input%(2*math.pi))
def f2(input):
    return np.cos(np.cos(input))
```

Passing an `numpy` array to the above functions will return a `numpy` array containing the data points. Using that the functions are plotted in $[-2\pi, 4\pi)$.

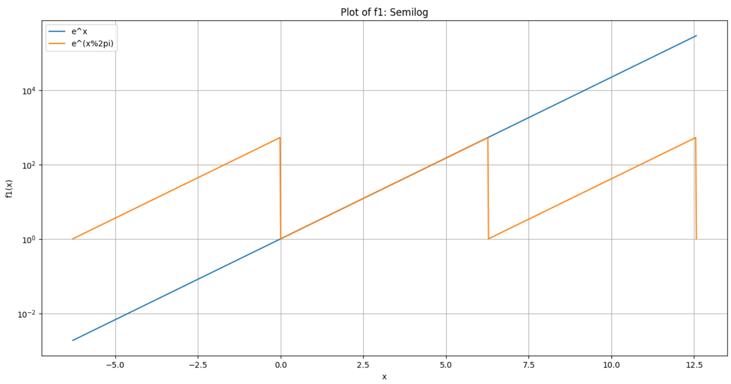

Figure 1: Plot of $f_1(x)$

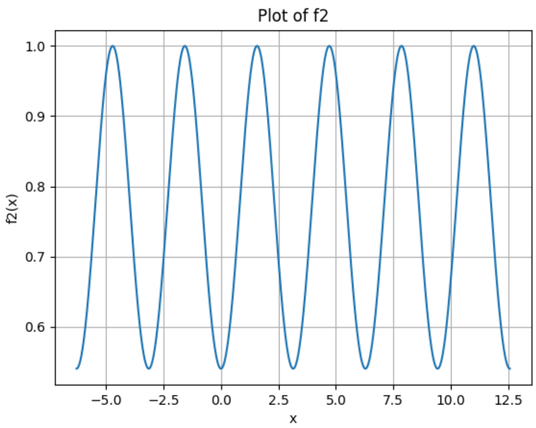

Figure 2: Plot of $f_2(x)$

3 Calculating the Fourier coefficients using Integration

`quad` function is used to perform integration in python.
Python code for calculating the coefficients:

```
def u(x,k,func):
    if func=="f1":
        return f1(x)*np.cos(k*x)
    elif func=="f2":
        return f2(x)*np.cos(k*x)
```

```
def v(x,k,func):
    if func=="f1":
        return f1(x)*np.sin(k*x)
    elif func=="f2":
        return f2(x)*np.sin(k*x)


def an(func,n):
    a_coeff=np.empty(n)
    if func=="f1":
        a_coeff[0]=sc.quad(u,0,2*math.pi,args=(0,'f1'))[0]/(2*math.pi)
        for i in range(1,n):
            a_coeff[i]=sc.quad(u,0,2*math.pi,args=(i,'f1'))[0]/(math.pi)
    elif func=="f2":
        a_coeff[0]=sc.quad(u,0,2*math.pi,args=(0,'f2'))[0]/(2*math.pi)
        for i in range(1,n):
            a_coeff[i]=sc.quad(u,0,2*math.pi,args=(i,'f2'))[0]/(math.pi)
    return a_coeff

def bn(func,n):
    b_coeff=np.empty(n)
    if func=="f1":
        for i in range(1,n):
            b_coeff[i]=sc.quad(v,0,2*math.pi,args=(i,'f1'))[0]/(math.pi)
    elif func=="f2":
        for i in range(1,n):
            b_coeff[i]=sc.quad(v,0,2*math.pi,args=(i,'f2'))[0]/(math.pi)
    return b_coeff
```

Plotting the coefficients in semilog and loglog plots:

Figure 3: Semilog plot of Fourier coefficients of f1(x)

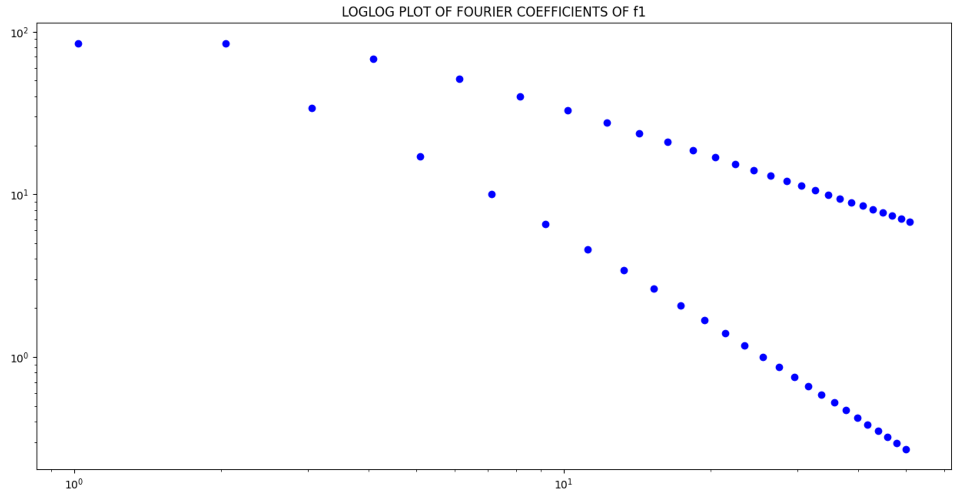

Figure 4: Loglog plot of Fourier coefficients of $f_1(x)$

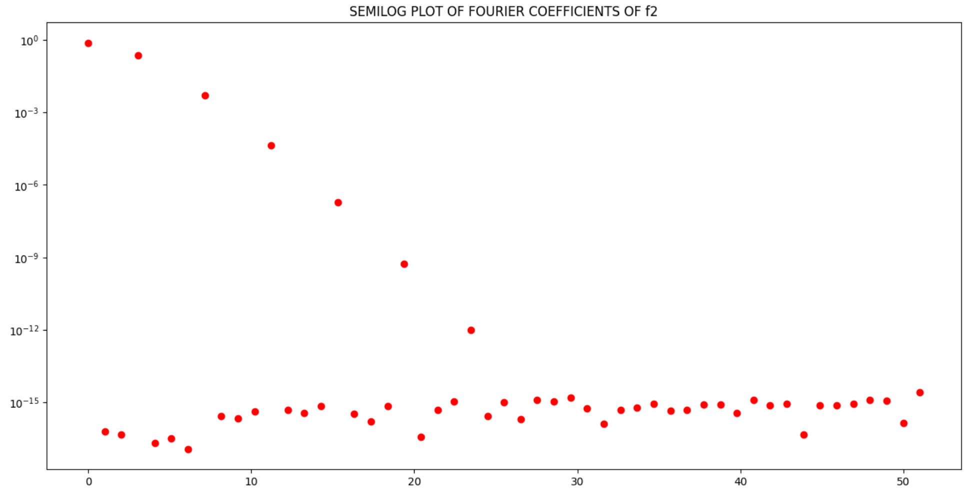

Figure 5: Semilog plot of Fourier coefficients of $f_2(x)$

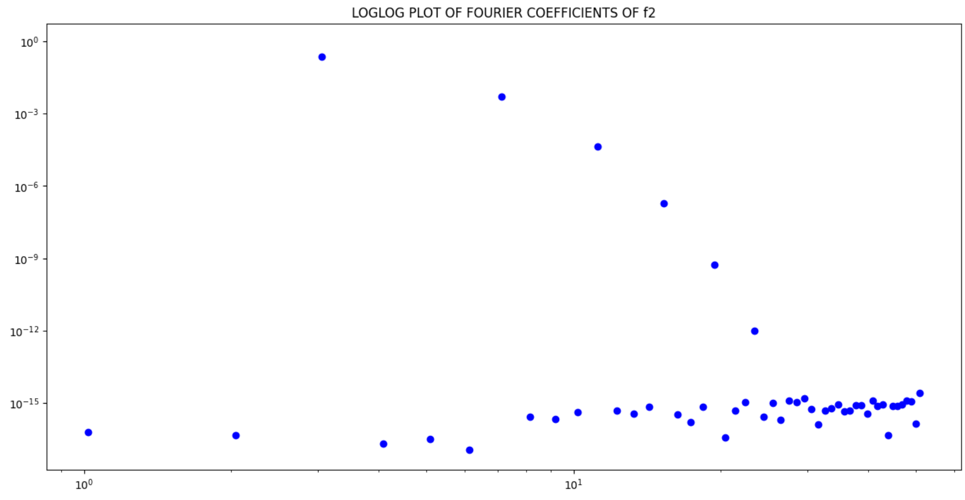

Figure 6: Loglog plot of Fourier coefficients of $f_2(x)$

1. For a_0 we get:

$$a_0 = \frac{1}{\pi}\int_{-\pi}^{\pi} e^x \, dx = \frac{1}{\pi}[e^x]_{-\pi}^{\pi} = \frac{e^\pi - e^{-\pi}}{\pi} = \frac{2\sinh(\pi)}{\pi}$$

2. For a_n we get (using two times integration by parts):

$$a_n = \frac{1}{\pi}\int_{-\pi}^{\pi} e^x \cos(nx) \, dx = \frac{2\cos(n\pi)\sinh(\pi)}{\pi(1+n^2)}$$

3. For b_n we get (using two times integration by parts):

$$b_n = \frac{1}{\pi}\int_{-\pi}^{\pi} e^x \sin(nx) \, dx = \frac{-2n\cos(n\pi)\sinh(\pi)}{\pi(1+n^2)}$$

Figure 7: Fourier coefficients of $f_1(x)$

- From the plots, we can see that b_n coefficients of $f_2(x)$ is almost zero. This is because $f_2(x)$ is an even function.
- Coefficients of $f_1(x)$ don't decay as much compared to the coefficients of $f_2(x)$ because $f_1(x)$ requires more number of Fourier components of higher frequencies to converge at the discontinuities.
- Figure 7 shows the dependence of n on the values of a_n and b_n. So approximately,

$$log(\frac{1}{n^2+1}) \approx -2log(n)$$
$$log(\frac{n}{n^2+1}) \approx -log(n)$$

  Hence Figure 4 has a linear nature.
- Figure 5 is linear because the Fourier coefficients of $f_2(x)$ are exponentially decreasing.

4 Least Squares Approach

The equation

$$f(x) \approx a_0 + \sum_{n=1}^{25} a_n + \sum_{n=1}^{25} b_n$$

is transformed into a product of matrices A and c.

$$Ac = b$$

A is defined like this:

$$\begin{bmatrix} 1 & cosx_1 & sinx_1 & cos2x_1 & .... & cos25x_1 & sin25x_1 \\ 1 & cosx_2 & sinx_2 & cos2x_2 & .... & cos25x_2 & sin25x_2 \\ 1 & cosx_3 & sinx_3 & cos2x_3 & .... & cos25x_3 & sin25x_3 \\ .. & .. & .. & .. & .... & .. & .. \\ 1 & cosx_{400} & sinx_{400} & cos2x_{400} & .... & cos25x_{400} & sin25x_{400} \end{bmatrix}$$

and b is the column matrix containing function values at x_1 to x_{400}.
Then best fit Fourier coefficients are calculated using `linalg.lstsq`
Plots comparing best fit coefficients and coefficients calculated from integration:

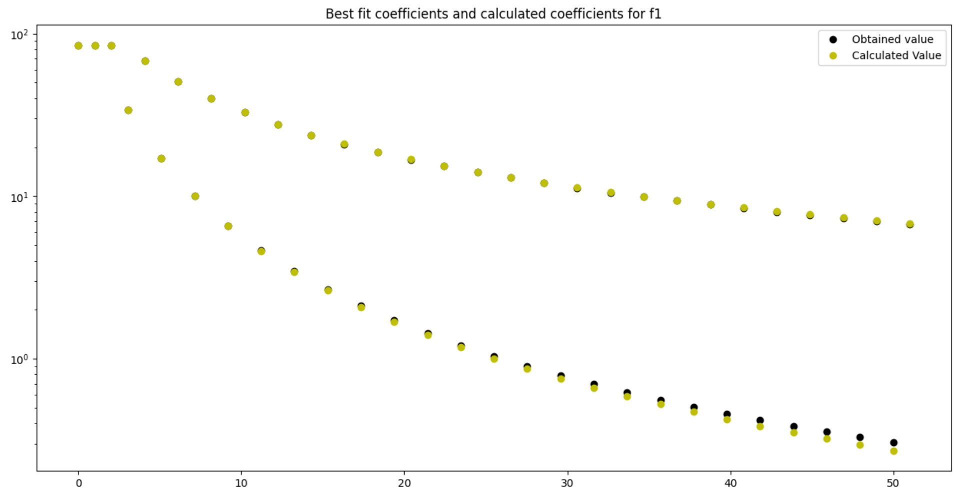

Figure 8: Best fit coefficients and previously calculated coefficients for f_1

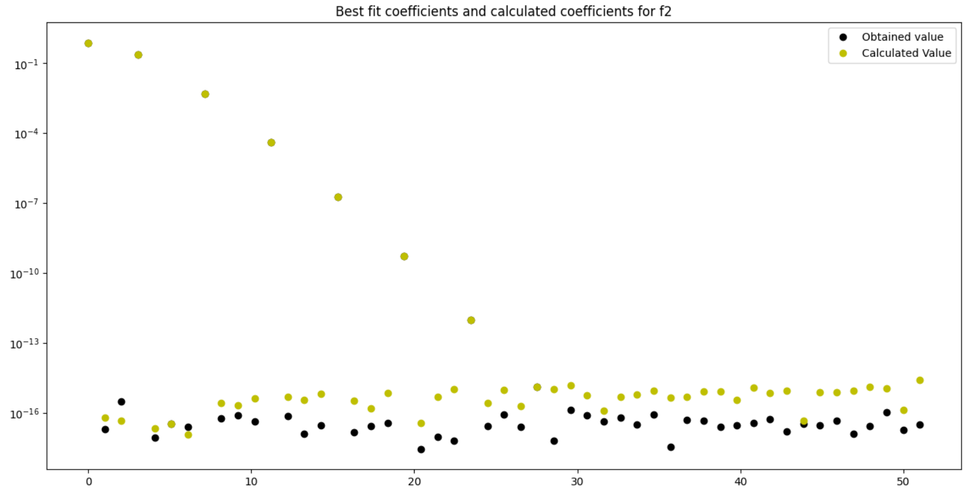

Figure 9: Best fit coefficients and previously calculated coefficients for f_2

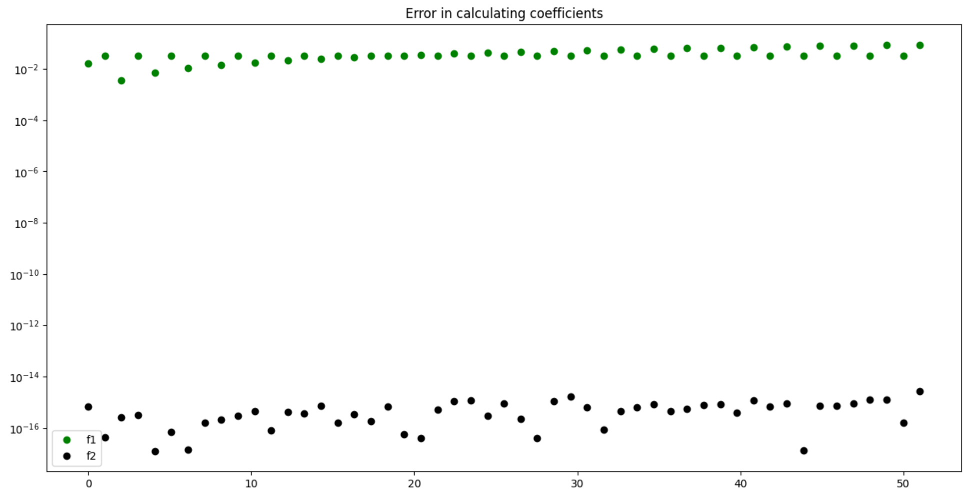

Figure 10: Difference between Best fit coefficients and previously calculated coefficients for f_1 and f_2

From Figure 9, we can see the magnitude of error in the values. The maximum deviation from coefficients calculated using integration and best fit coefficients is 0.08812 for f_1 and $2.63e-15$ for f_2.

5 Finding the function values for the calculated best fit coefficients

The best fit coefficients column matrix is multiplied with A to get a new b matrix. Plot of b matrix compared with the actual function values:

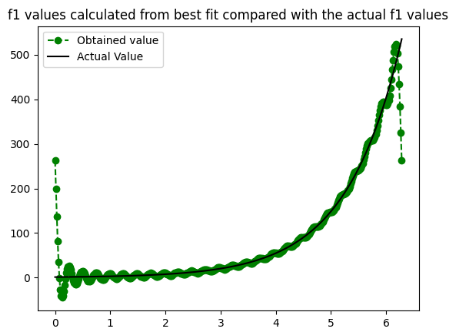

Figure 11: b column values and f_1

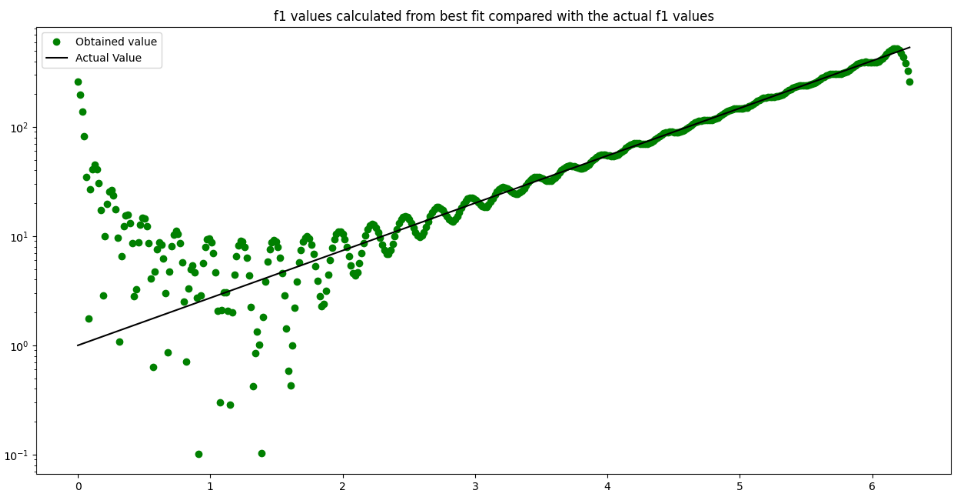

Figure 12: b column values and f_1 in semilog plot

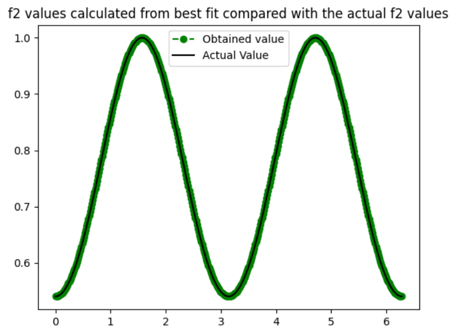

Figure 13: b column values and f_2

6 Conclusions and Inference

- Two approaches were used to calculate Fourier coefficients for functions.
- Error between the two approaches is very much negligible.
- Calculating Fourier series expansion for discontinuous functions is difficult and requires more number of coefficients for a close approximation.
- Fourier series approximation for periodic functions is really close to the actual function.