EE2703 : Applied Programming Lab
Assignment 4
Fourier Approximations

Adityan CG//EE19B003

March 11, 2021

1 Introduction

The assignment is about Fitting a perfectly suitable periodic function and a non periodic exponential function into Fourier series and observing the abnormalities. The two functions are given by,

$$f0(t) = exp(t)f1(t) = cos(cos(t))$$

Functions are written in python to pass either a vector or scalar to get the same type of output

2 Finding coefficients

To find the coefficients separate functions are written `u(x,k,b) and v(x,k,b)` where b represents the 1st or 2nd function and u and v for odd and even terms respectively.

The integration is done by using

```
scipy.integrate.quad(u,0,2*PI,args=(k,i))
```

where args passes extra arguments to the function that needs to be integrated. 51 Coefficients are stored.

3 Plotting Coefficients vs n

The coefficients are plotted vs n by using red and blue dots.For An and Bn separately.And Each function has 2 plots :

- Loglog plot
- Semilog plot

4 Least squares approach

Similar to the last assignment we convert the fourier series into a matrix. By making the values of x as such,

$$x_i = np.linspace(0, 2 * PI, 400)$$

into 400 steps. Then the least squares method is used to identify the best estimate of the coefficients. In my code I have used the matrix is designed for 50 coefficients along with the constant coefficient.

The Matrix formation and the lstsq is done by,

```
b = exp(x)   # f has been written to take a vector
A = np.zeros((400,101))     # allocate space for A
A[:,0]=1              # col 1 is all ones
for k in range(1,51):
A[:,2*k-1] = np.cos(k*x)    # cos(kx) column
A[:,2*k]   = np.sin(k*x)    # sin(kx) column #endfor

c0 = np.linalg.lstsq(A,b,rcond=None)[0]        # the '[0]' is to pull out the# best fit vector.
#lstsq returns a list.
```

From the C vectors I have separated the even(b) and odd(a) coefficients separately for both the functions as

- For exp(x): A0,B0 are Coefficients by direct integration
- a0,b0 are Best estimate of Coefficients
- For cos(cos(x)): A1,B1 are Coefficients by direct integration
- a1,b1 are Best estimate of Coefficients
- The error between the corresponding coefficients will be printed when the code runs.
- The plot for the estimated coefficients are shifted by 1 unit of n in the plot to differentiate between the estimated and integration coefficients as the order of error between the in cos(cos()) function is of the order **1e-15**

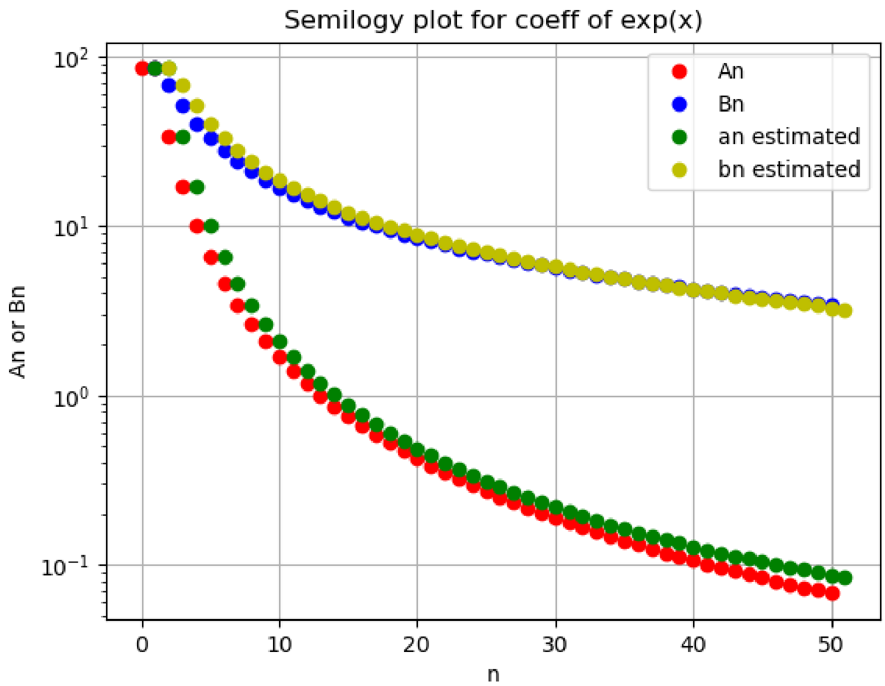

Figure 1: Put your sub-caption here

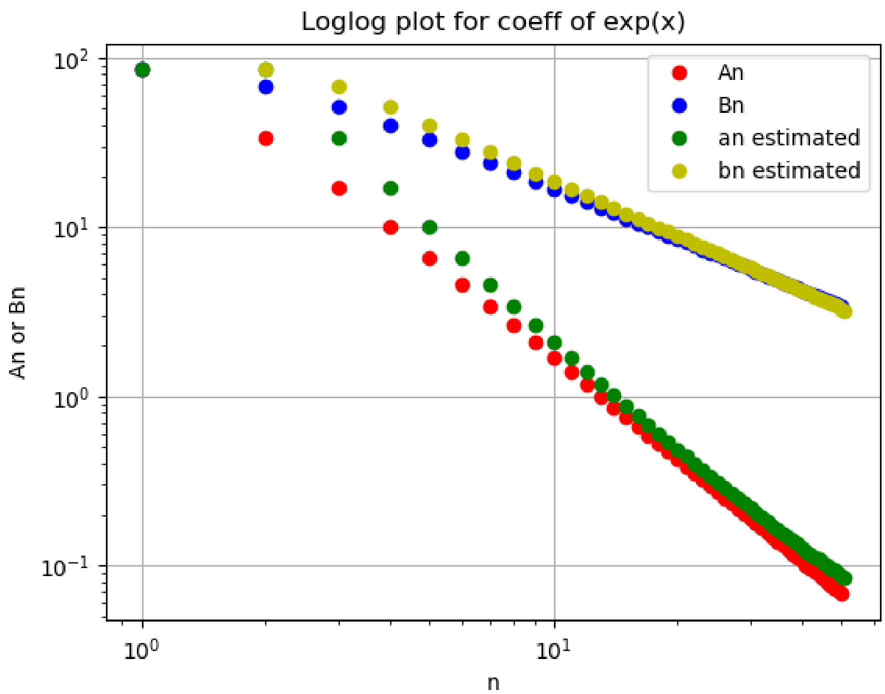

Figure 2: Put your sub-caption here

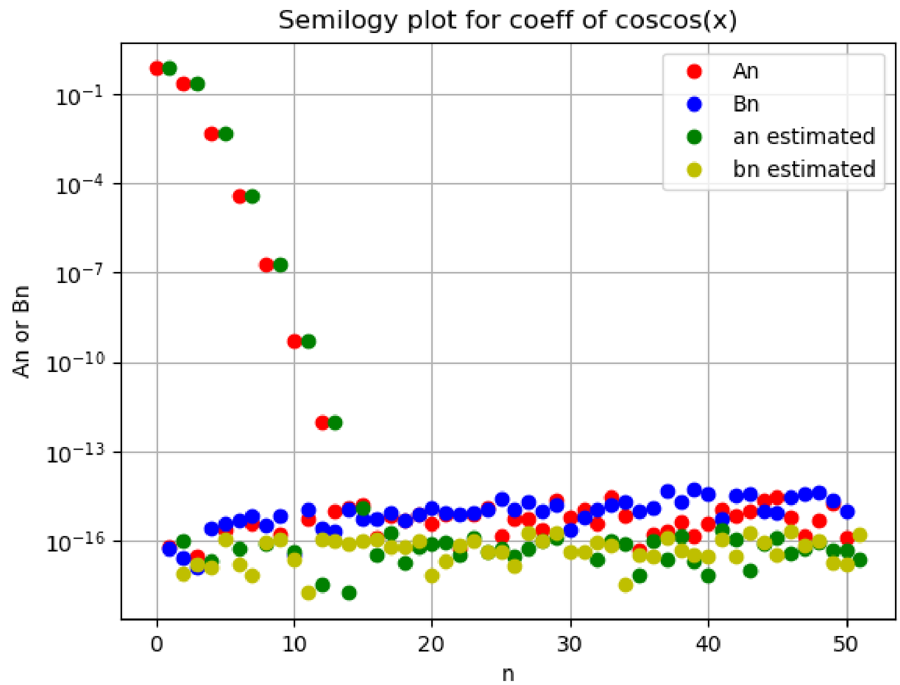

Figure 3: Put your sub-caption here

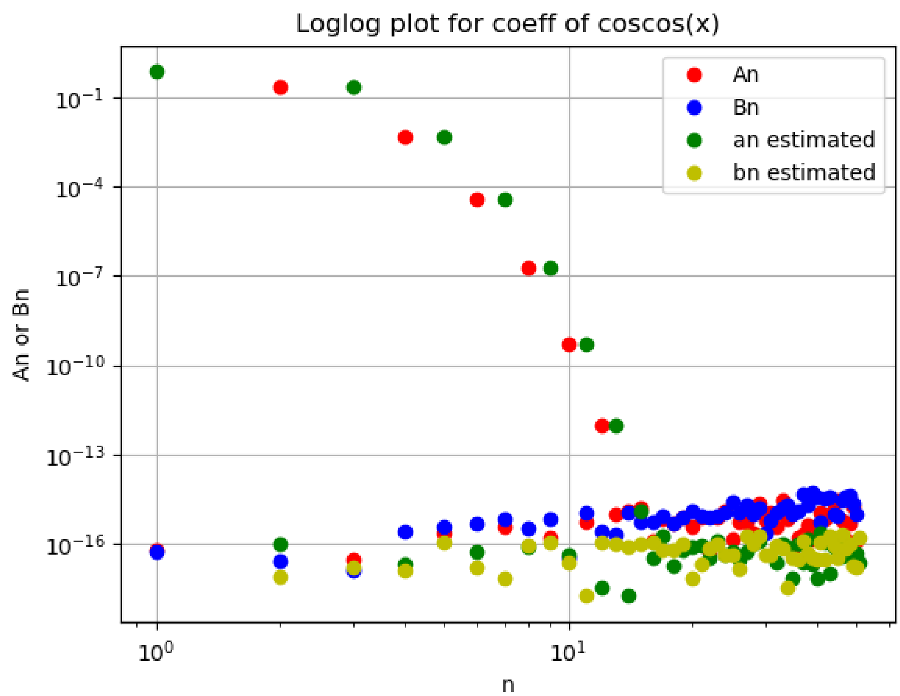

Figure 4: Put your sub-caption here

For the Questions asked in 3rd task of the Assignment,

(a)The Coefficients for the 2nd part are almost zero because cos(cos()) is an even function.

(b)In the first case the periodic extension of the function is sharp(Looks like a Triangular function).But in the second case the function is much smoother.

- According to **Riemann Lebesgue Lemma**, the smoother the function the faster the Decay.

(c)In figure 4 the log log plot is linear because the decay is proportional to n. Whereas the semilog plot in figure 5 has coefficients abruptly decaying. So it is non-linear.

5 Estimated vs Original Function

The Estimated Functional Values from the matrix multiplication of A*c1 and A*c0 are plotted along with the original function in Figures 1 and 2.

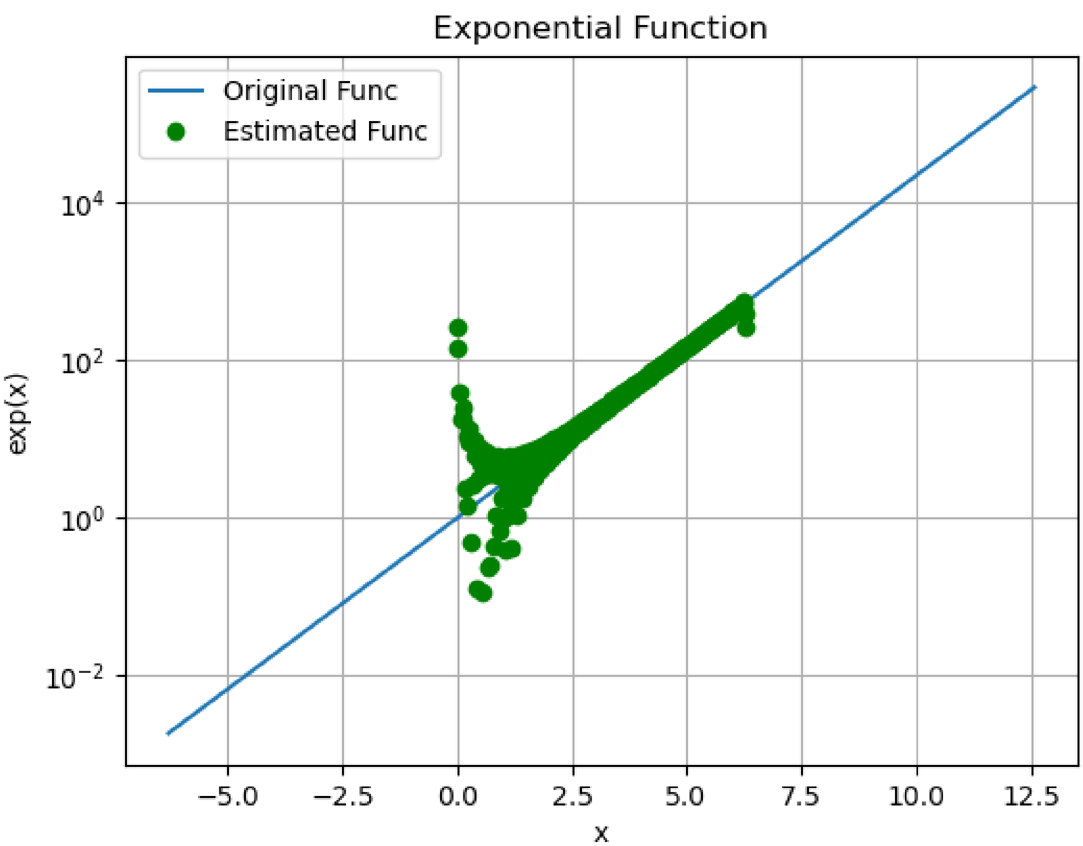

Figure 5

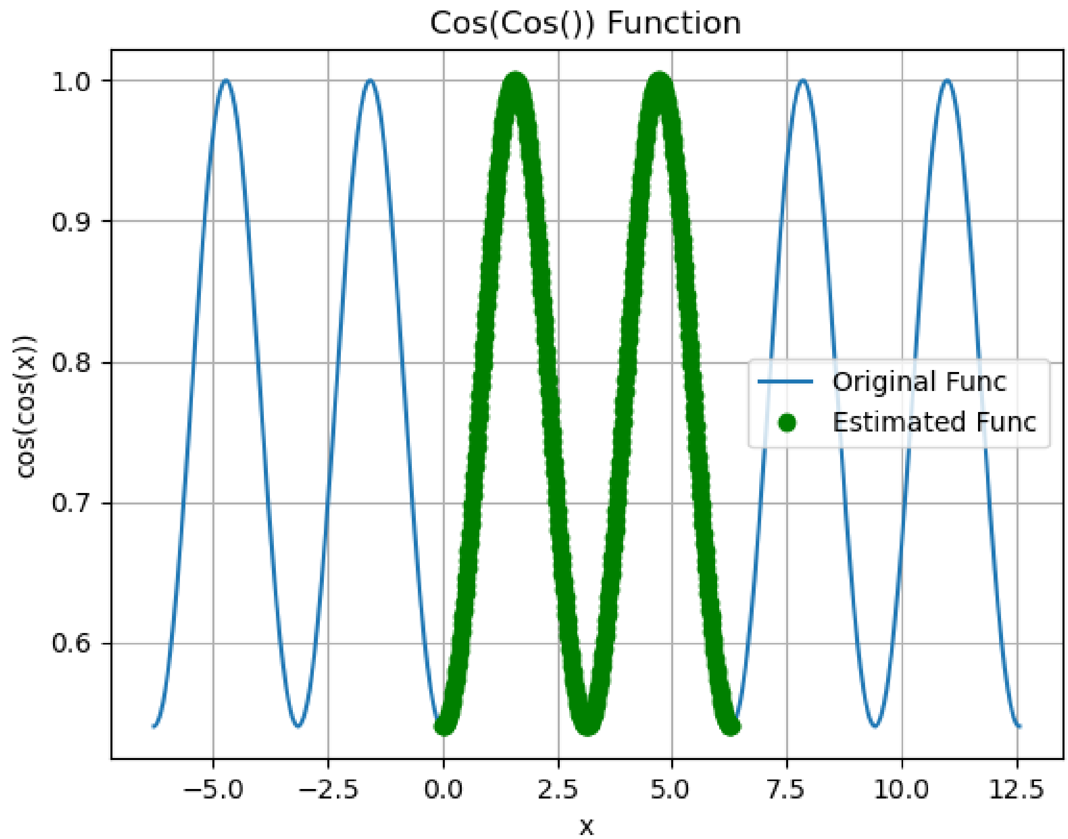

Figure 6

6 Conclusion

As the Coefficient Decay is slower in the first case, it has so much deviation in the intial levels, for the exponential Function in Figure 1.
Whereas in Figure 2 the green circles almost converge with the functional values. This is because the higher coefficients are almost zero(In orders of 1e-15), So we don't see any difference.
The Error difference in output when the code runs also depicts the same thing.