Assignment 4: Fourier Approximations

Hariharan P EE20B042

February 24, 2022

Brief Up

This assignment is computing Fourier transformation for exp(x) and cos(cos(x)).
Tasks:

- Computing the Fourier transform using actual integration.
- Computing the Fourier transform using least square approach.
- Plotting these coefficients obtained using the two methods .

1 Introduction to Fourier transformation:

$$f(x) = a_0 + \sum_{n=1}^{+\infty} \{a_n cos(nx) + b_n sin(nx)\} \tag{1}$$

where ,

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

2 Creating functions exp(x) and cos(cos(x)):

We initialise the functions to create exp(x) and cos(cos(x)).The following python code is used to do that.

```
def e(x):
        return exp(x)

def coscos(x):
        return cos(cos(x))
```

```
#range of x values from -2   to 4
x_tot = linspace(-2*pi,4*pi,1200)

#the corresponding function values for those x values
```

We shall now plot these functions.

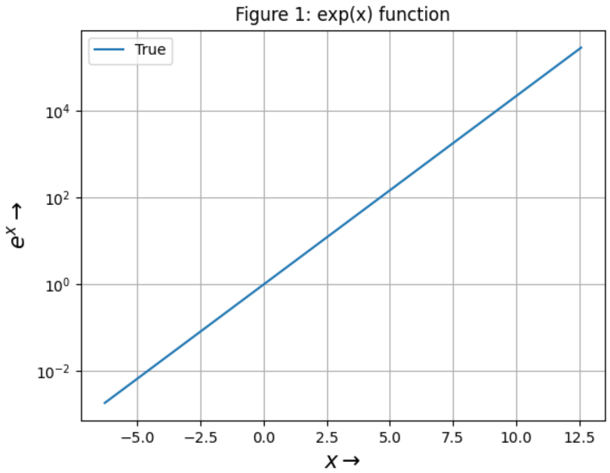

Figure 1: exp(x) in semilog

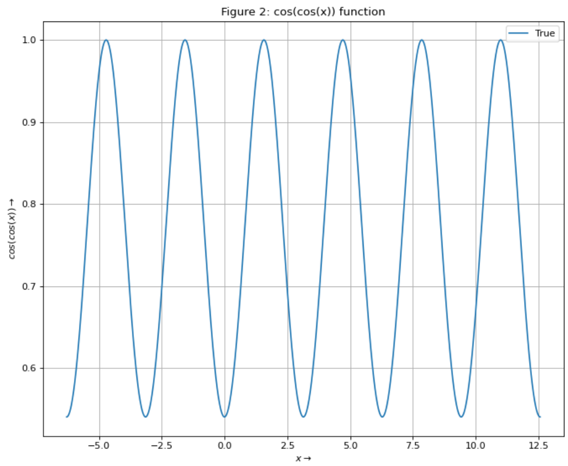

Figure 2: cos(cos(x)) in semilog

We could infer from the plots that exp(x) is aperiodic whereas cos(cos(x)) is periodic. The functions generated by the Fourier series should be a periodic extension of the actual functions for the corresponding time period.

```
coscos_val = coscos(x_tot)

#making exp() and coscos(x) periodic
p_exp_val = []
p_exp_val[0:400] = exp_val[400:800]
p_exp_val[400:800] = exp_val[400:800]
p_exp_val[800:1200] = exp_val[400:800]

p_coscos_val = []
p_coscos_val[0:400] = coscos_val[400:800]
```

The plots showing the periodic extensions of exp(x) and cos(cos(x)) are as shown below:

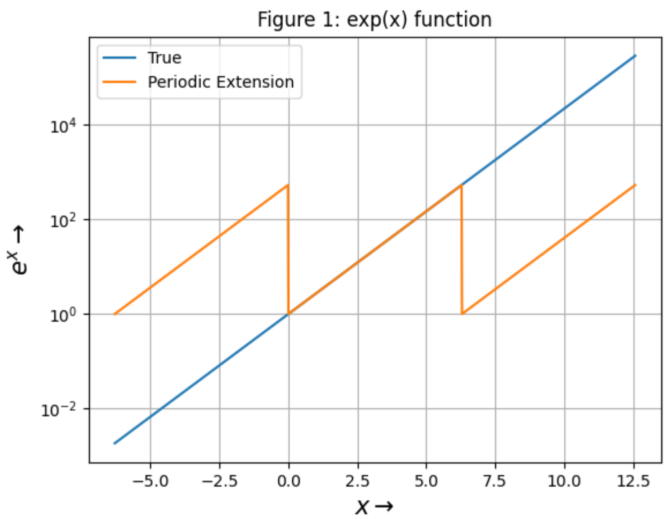

Figure 3: exp(x) in semilog

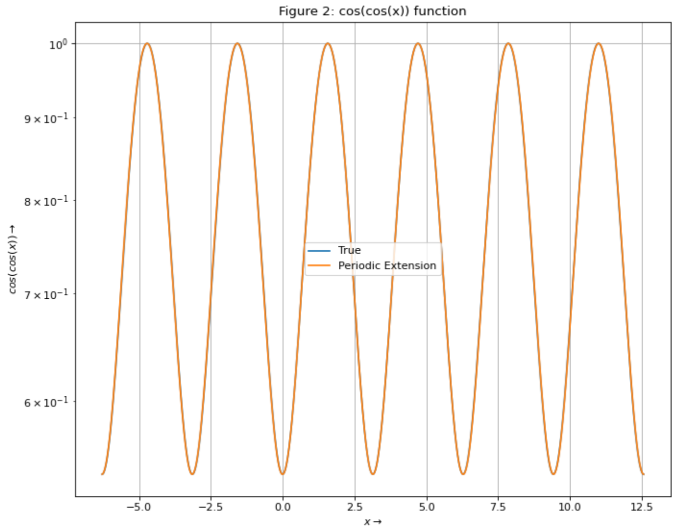

Figure 4: cos(cos(x)) in semilog

3 The Fourier coefficients:

We shall obtain the first 51 coefficients to analyse.Using the defined function shown below, we get the required no. of coefficients for the respective function.

```
def F_C(n,function):

    coeff= zeros(n)
    def fcos(x,k,f):
        return f(x)*cos(k*x)/pi
    def fsin(x,k,f):
        return f(x)*sin(k*x)/pi

    coeff[0] = integrate.quad(function,0,2*pi)[0]/(2*pi)

    for i in range(1,n):
        if(i%2==1):
            coeff[i] = integrate.quad(fcos,0,2*pi,
            args=(int(i/2)+1,function))[0]
        else:
            coeff[i] = integrate.quad(fsin,0,2*pi,
```

Now we obtain the coefficients for the exp(x) and cos(cos(x)) using the following code.

```
e_coeff=F_C(51,e)
coscos_coeff=F_C(51,coscos)
```

On plotting the coefficients value on semilog and loglog, the following plot is generated.

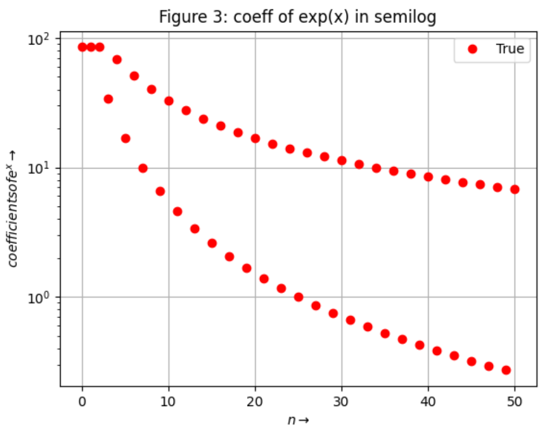

Figure 5: True values of exp(x) coefficients in semilog

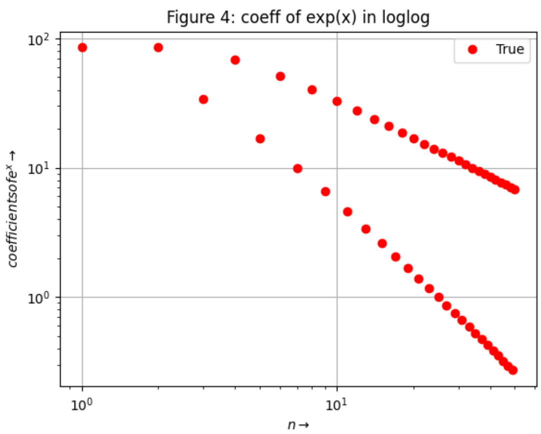

Figure 6: True values of exp(x) coefficients in loglog

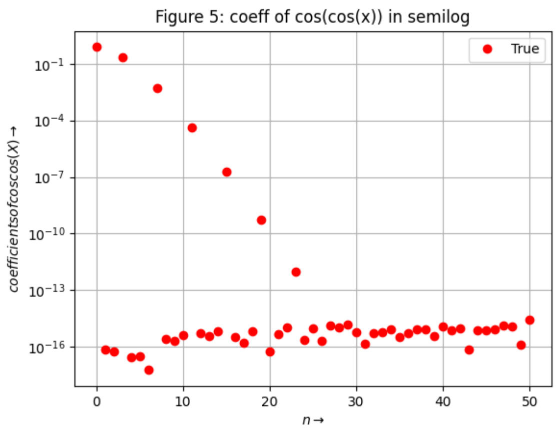

Figure 7: True values of cos(cos(x)) coefficients in semilog

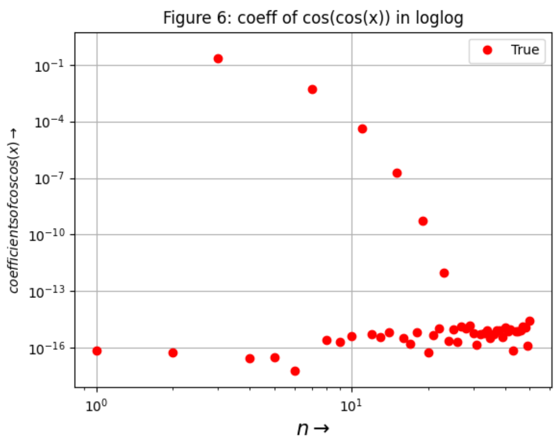

Figure 8: True values of cos(cos(x)) coefficients in loglog

4 Analysing using least square method:

We shall create a matrix A , of shape 51x400 as shown. using the below

$$\begin{pmatrix} 1 & \cos x_1 & \sin x_1 & \dots & \cos 25x_1 & \sin 25x_1 \\ 1 & \cos x_2 & \sin x_2 & \dots & \cos 25x_2 & \sin 25x_2 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & \cos x_{400} & \sin x_{400} & \dots & \cos 25x_{400} & \sin 25x_{400} \end{pmatrix}$$

Figure 9: matrix A

code.

```
A=zeros((400,51)) # initialising A
A[:,0]=1 # col 1 is all ones
for k in range(1,26):
    A[:,2*k-1]=cos(k*x) # cos(kx) column
    A[:,2*k]=sin(k*x) # sin(kx) column
```

we shall initialise matrix b with the real values of exp(x) and cos(cos(x) for 0 to 400 as the values of x. The code to create the matrix b is as follows.

```
e_b=e(x)
coscos_b=coscos(x)
```

Now we shall use "lstsq" function to deduce the estimated matrix that contains the coefficients.The code is as follows.

```
e_c=lstsq(A,e_b,rcond=None)[0]
coscos_c=lstsq(A,coscos_b,rcond=None)[0]
```

The least squares approach is still an approximate method and will definitely have a slight deviation from the actual value. We have calculated the deviation using the code below.

```
# The difference between the actual and predicted values are found
e_diff= abs(e_coeff - e_c)
coscos_diff = abs(coscos_coeff - coscos_c)
# The deviation between the actual and predicted values are found
e_dev = e_diff.max()
coscos_dev = coscos_diff.max()
```

The deviation between coefficients for exp(x) is 1.3327308703354106
The deviation between coefficients for cos(cos(x)) is 2.729038925840725e-15
Now we shall dot multiply the A matrix and the estimated matrix obtained using least square approach to get the estimated values of the function. The code is as follows.

```
estimated_e=dot(A,e_c)
estimated_coscos=dot(A,coscos_c)
```

The plots in order to show the differences between the actual and predicted values of the fourier coefficients are shown below

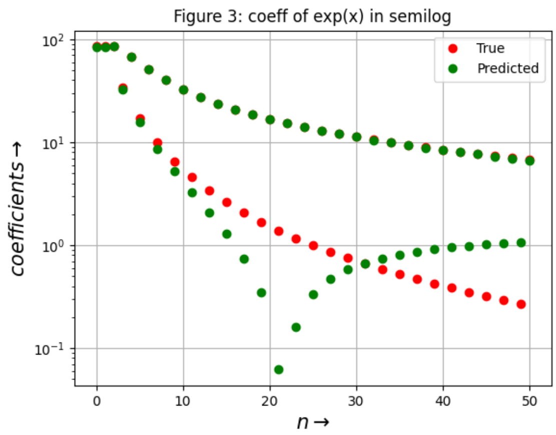

Figure 10: exp(x) coefficients in semilog

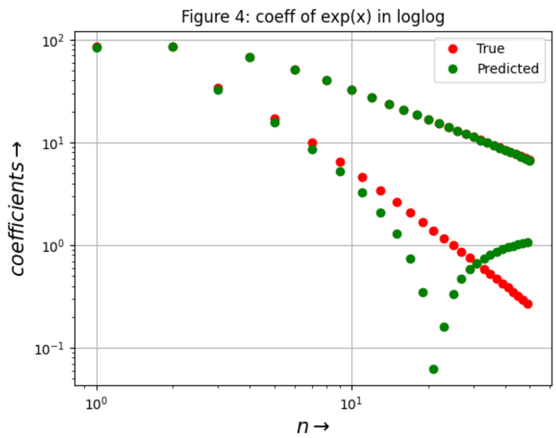

Figure 11: exp(x) coefficients in loglog

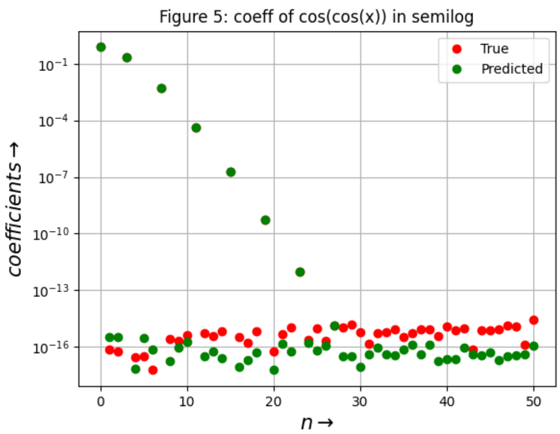

Figure 12: cos(cos(x)) coefficients in semilog

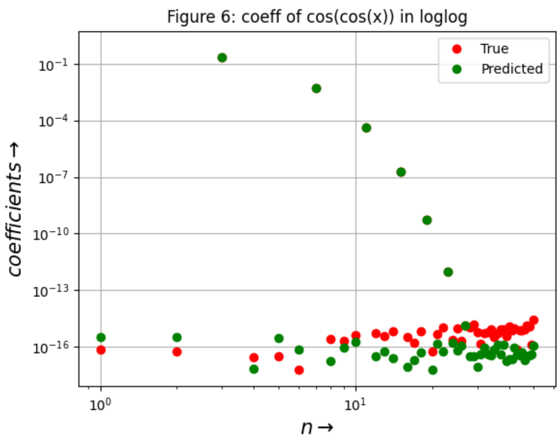

Figure 13: cos(cos(x)) coefficients in loglog

There is very good estimation in values in the case of cos(cos(x)) but a significant amount of difference in the case of exp(x). The reason for this is that the periodic extension of the exponential function is discontinuous, and hence would require a lot more samples to accurately determine its Fourier coefficients.The effect of this lack of samples is felt more near the discontinuity of the signal.

5 Estimated Fourier transformations :

We now plot the estimated values of exp(x) and cos(cos(x)) obtained along with the true values to throw light on the deviation or difference. cos(cos(x)

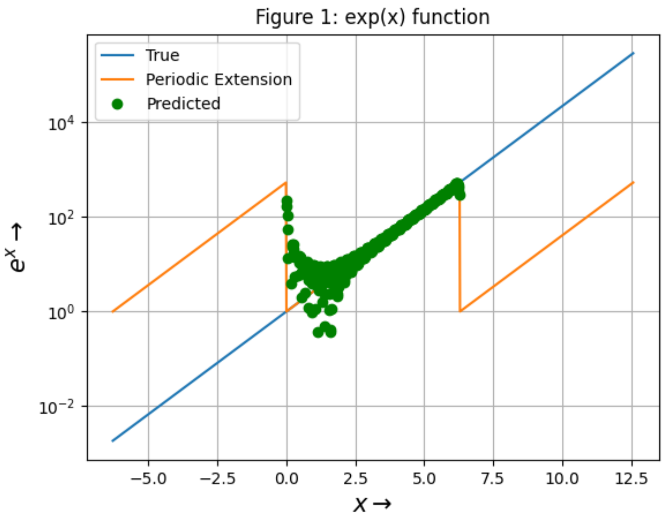

Figure 14: exp(x)

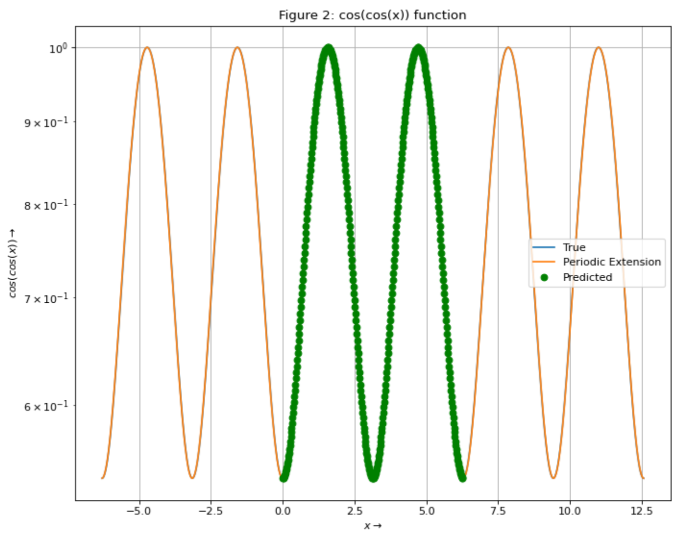

Figure 15: cos(cos(x))

agrees well with the estimation but exp(x) doesn't.the reason is the partial sums of the Fourier series will have large oscillations near the discontinuity of the function. These oscillations do not die out as n increases, but approaches a finite limit.

Conclusion

We performed the fourier transformation on two funtions.One on a continuous function, and the other on a function with finite discontinuities. The methods adopted in finding the respective Fourier coefficients have been direct evaluation of the Fourier series formula using integration,and the other being the Least Squared approach. We notice close matching of the two methods in case of cos(cos(x)) while, there is a larger discrepancy in exp(x). We highlight this discrepancy and draft a reason for it which is due to the discontinuity at the end of time period.