Assignment 4: Fourier Approximations

EE19B049(Pragada Jahnavi)

March 10, 2021

Abstract

The goal of this assignment is the following.

- To find two functions e^x and $cos(cos(x))$ using the Fourier series with two methods.
- Method1(Using Integrals) and Method2(Using lstsq()).
- To plot graphs to understand the above

1 The functions e^x and $cos(cos(x))$

The following python snippet is used to declare the functions e^x and $cos(cos(x))$. The x values are also declared from -2π to 4π.

```
def f1(x):
return exp(x)
def f2(x):
return cos(cos(x))

x = linspace(-2*pi,4*pi,1200)
act1 = f1(x)
act2 = f2(x)
```

The following code is used to plot the graphs of e^x and $cos(cos(x))$.

```
figure(1)
semilogy(x,f1(x),label='Actual value')
title("exp(x)")
xlabel(r'$x\rightarrow$',size=15)
ylabel(r'$e^x\rightarrow$',size=15)
grid(True)
legend()
```

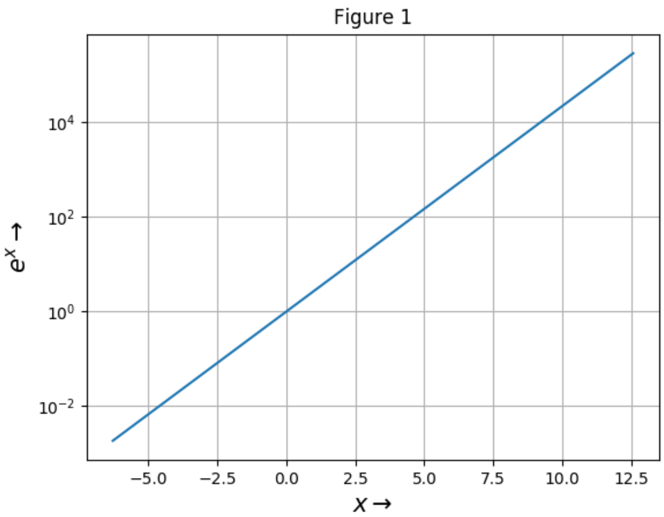

Figure 1: Data plot

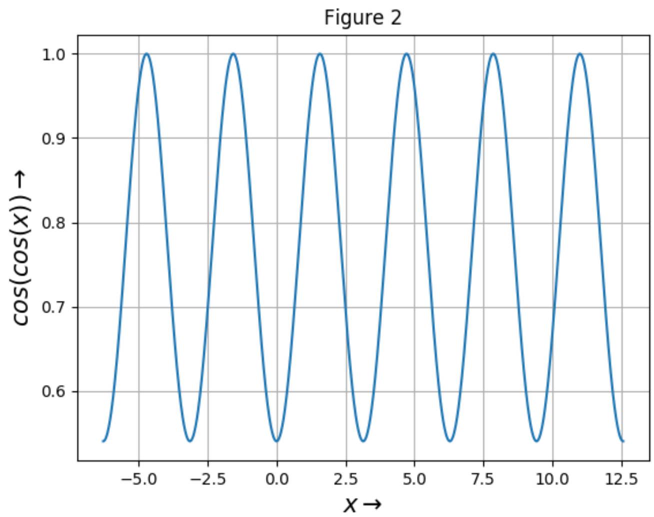

Figure 2: Data plot

The plots of e^x and $cos(cos(x))$ are as shown below:

As it is evident from the plots, e^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. The plots showing the periodic extensions of e^x and $cos(cos(x))$ are as shown below:

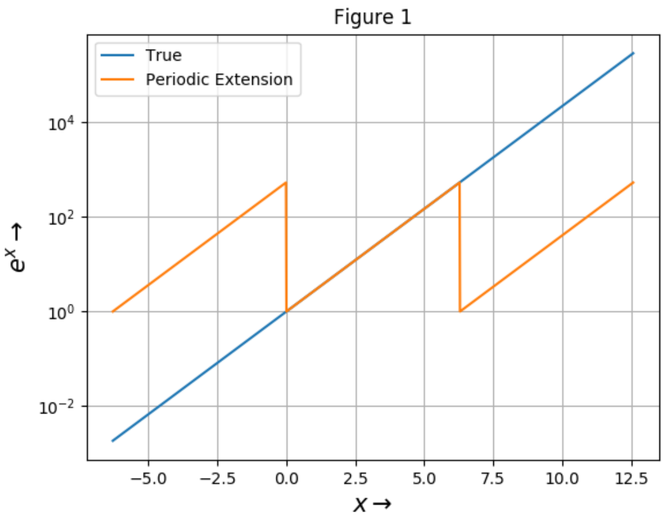

Figure 3: Data plot

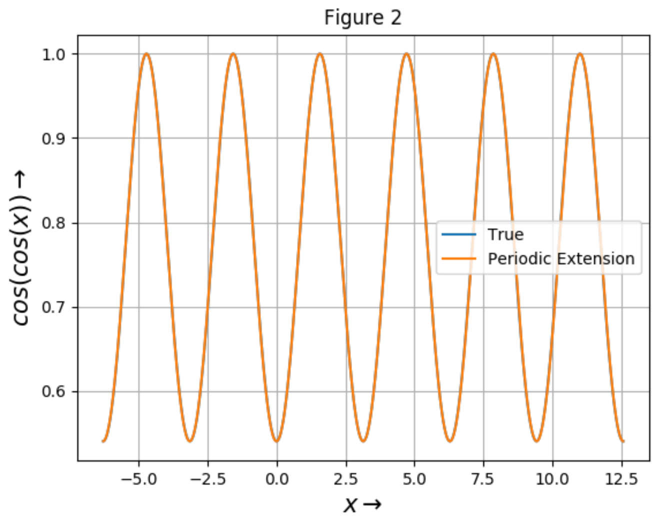

Figure 4: Data plot

2 The Fourier coefficients

The fourier series used to approximate a function is as follows:

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

The equations used here to find the Fourier coefficients are as follows:

$$a_0 = \frac{1}{2\pi} \int_0^{2\pi} f(x)dx \tag{2}$$

$$a_n = \frac{1}{\pi} \int_0^{2\pi} f(x)\cos(nx)dx \tag{3}$$

$$b_n = \frac{1}{\pi} \int_0^{2\pi} f(x)\sin(nx)dx \tag{4}$$

Hence, in python we will use the *quad()* function to perform an intergration function. First we'll have to create functions which contains the variable k also. The python code snippet for declaring the functions with an additional variable k is as follows:

```
def an1(x,n):
return f1(x)*cos(n*x)
def bn1(x,n):
return f1(x)*sin(n*x)
def an2(x,n):
return f2(x)*cos(n*x)
def bn2(x,n):
return f2(x)*sin(n*x)
```

The python code snippet for finding the fourier coefficients is as follows:

```
for k in range(26):
ak1 = integrate.quad(an1,0,2*pi,args=(k))[0]/pi #values of coefficients
bk1 = integrate.quad(bn1,0,2*pi,args=(k))[0]/pi
ak2 = integrate.quad(an2,0,2*pi,args=(k))[0]/pi
bk2 = integrate.quad(bn2,0,2*pi,args=(k))[0]/pi
    if k==0:
f1_ser += ak1/2
f2_ser += ak2/2
f1_coeff[index][0] = ak1/2
f2_coeff[index][0] = ak2/2
index += 1
else:
f1_ser += ak1*cos(k*x) + bk1*sin(k*x)
f2_ser += ak2*cos(k*x) + bk2*sin(k*x)
f1_coeff[index][0] = ak1
f2_coeff[index][0] = ak2
```

```
f1_coeff[index+1][0] = bk1
f2_coeff[index+1][0] = bk2
index +=2
```

3 The plots of Fourier coefficients

The semilog and log plots of the Fourier coefficients of e^x and $cos(cos(x))$ is as shown:

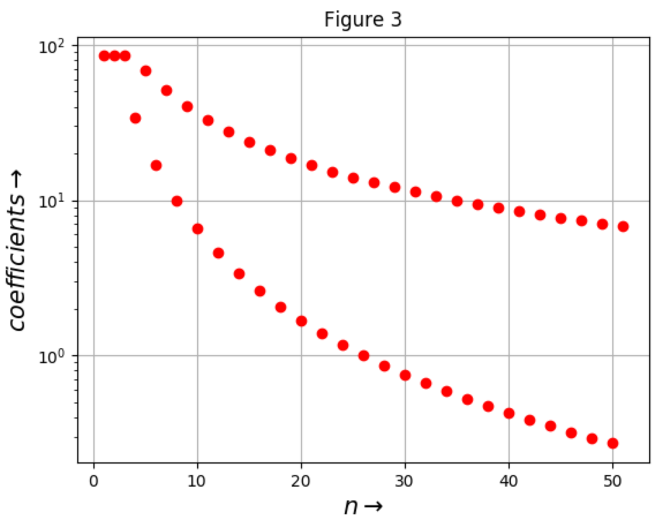

Figure 5: Semilog plot of the fourier coefficients of e^x

a. As it is evident from the plots, b_n is nearly zero for $cos(cos(x))$. This is because $cos(cos(x))$ is an even function, hence in the fourier series expansion, all the b_n terms should be zero for the series to be an even function.

b. The magnitude of the coefficients would represent how much of certain frequencies happen to be in the output. $cos(cos(t))$ does not have very many frequencies of harmonics, so it dies out quickly. However, since the periodic extension of e^t is discontinuous. To represent this discontinuity as a sum of continuous sinusoids, we would need high frequency components, hence coefficients do not decay as quickly.

c. The loglog plot is linear for e^t since Fourier coefficients of e^t decay with $1/n$ or $1/n^2$. The semilog plot seems linear in the $cos(cos(t))$ case as its fourier coefficients decay exponentially with n.

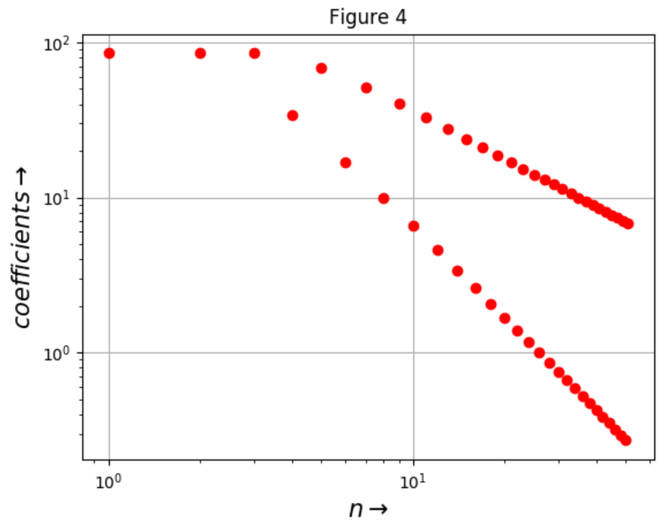

Figure 6: Log plot of the fourier coefficients of e^x

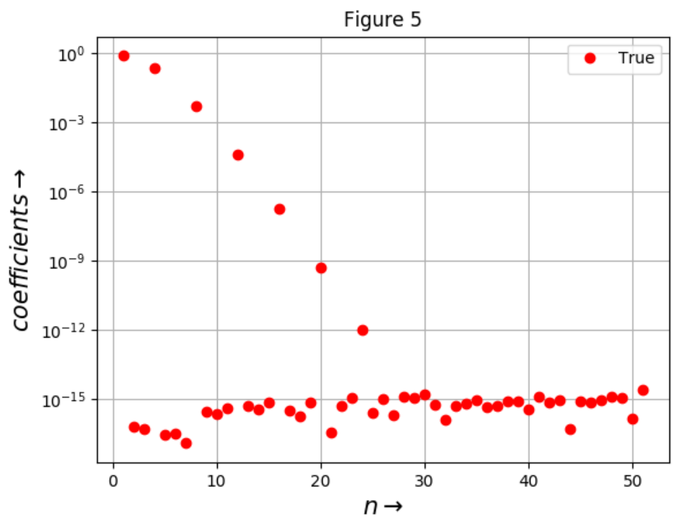

Figure 7: Semilog plot of the fourier coefficients of $cos(cos(x))$

4 The Least Squares Approach

For the least squares approach, we'll have to create matrices and then use *lstsq()* function inorder to get the most approximate values of the fourier coefficients.
The python code snippet to create the matrices and to get the least squared value of the coefficients is as follows:

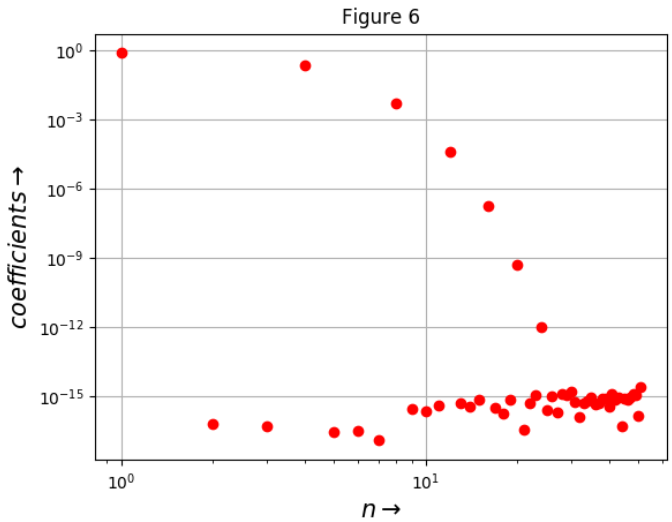

Figure 8: Log plot of the fourier coefficients of $cos(cos(x))$

```
y = linspace(0,2*pi,401)
y = y[:-1]

pred1 = f1(y)
pred2 = f2(y)

M = zeros((400,51))
M[:,0] = 1
for k in range(1,26):
M[:,2*k-1] = cos(k*y)
M[:,2*k] = sin(k*y)

f1_coeff_pred = lstsq(M,pred1,rcond=None)[0]
f2_coeff_pred = lstsq(M,pred2,rcond=None)[0]
```

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

5 Deviation from Actual Values

The least squares approach is still an approximate method and will definitely have a slight deviation from the actual value.
The following python code snippet is used to calculate the deviation:

```
diff1 = abs(f1_coeff - f1_coeff_pred)
diff2 = abs(f2_coeff - f2_coeff_pred)
```

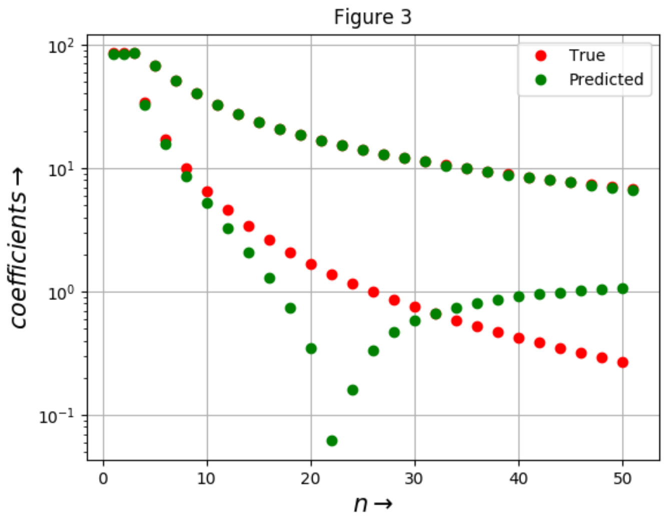

Figure 9: Semilog plots for e^x

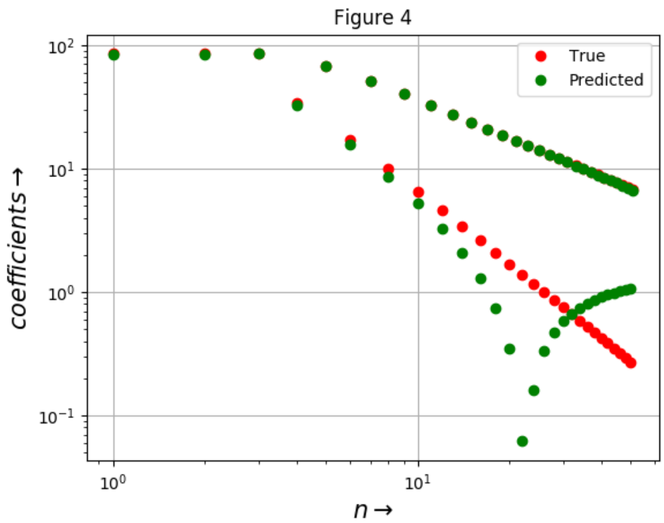

Figure 10: Log plots for e^x

```
maxdiff1 = diff1.max()
maxdiff2 = diff2.max()
```

The deviation in the case of e^x is 170.1304798
The deviation in the case of $cos(cos(x))$ is 0.9950046

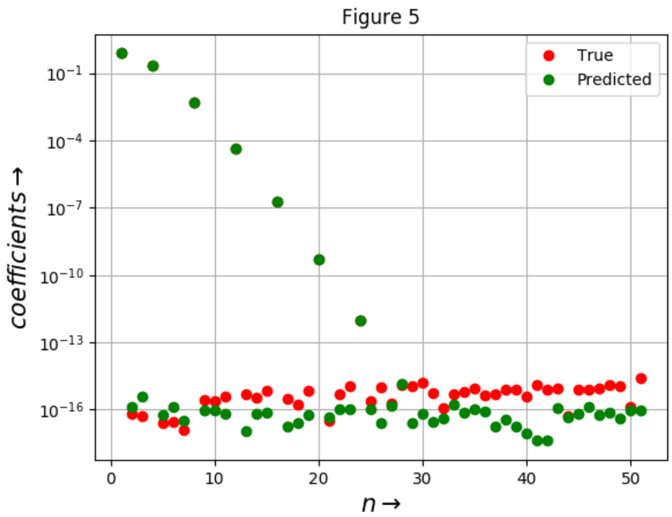

Figure 11: Semilog plots for $cos(cos(x))$

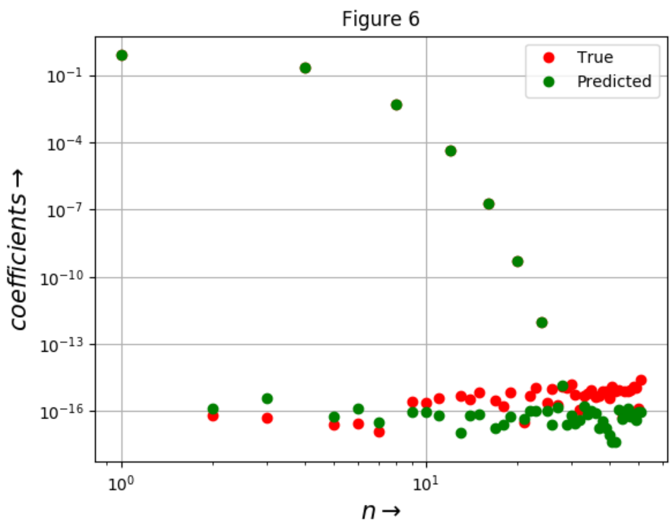

Figure 12: Log plots for $cos(cos(x))$

There is very good agreement in values in the case of $cos(cos(x))$ but a significant amount of difference in the case of e^t. 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. If we increased the number of samples to 10^6, the maximum deviation would reduce, but not vanish. The effect of this lack of samples is felt more near the discontinuity of the signal.

6 Estimated Functions

Using the predicted values of the fourier coefficients, we can calculate the functional values for both e^x and $cos(cos(x))$.
The plots showing both the actual and predicted functional values are as shown below:

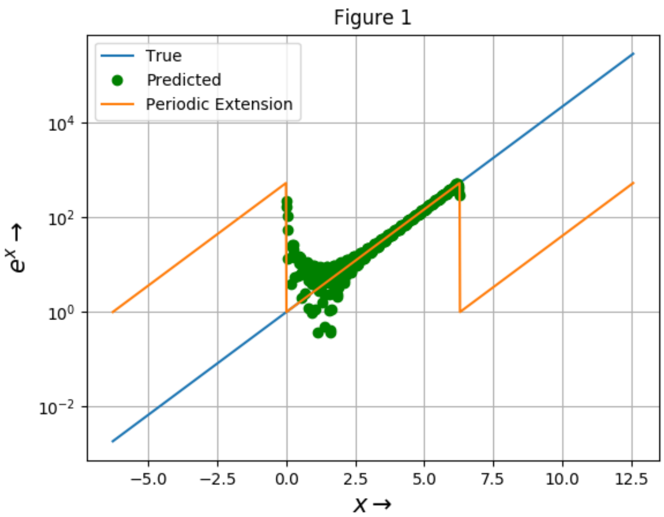

Figure 13: Actual and predicted values for e^x

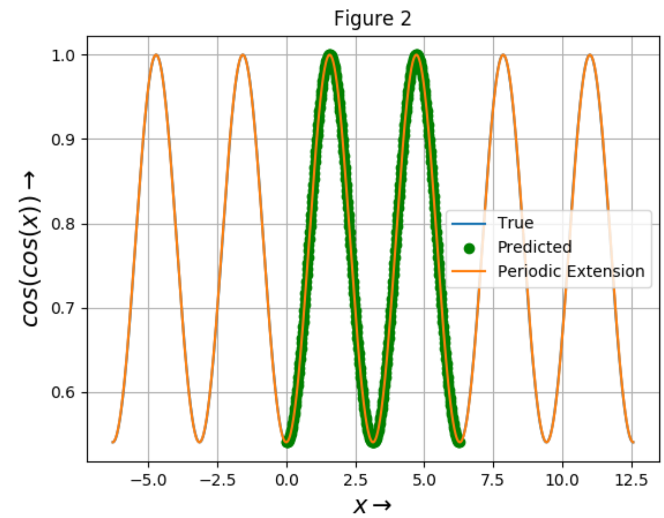

Figure 14: Actual and predicted values for $cos(cos(x))$

The $cos(cos(t))$ vs t graph, agrees almost perfectly, beyond the scope of the precision of the least squares fitter. The Fourier approximation of e^t does not agree very well to the ideal case near the discontinuity. The cause for this is the Gibbs Phenomenon, which can be described as below. 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. This is one of the causes of ringing artifacts in signal processing, and is very undesirable. Plotting the output on a linear graph would make this ringing much more apparent.

Conclusions

- We saw two different ways to calculate the Fourier series of a periodic signal.
- We saw how least squares fitting can be used to simplify the process of calculating the Fourier Series.
- We observed Gibbs phenomenon at the discontinuity in the Fourier approximation of e^t.