Report - Assignment 3

- Akshay Anand (EE16B046)

February 24, 2018

Abstract

In this week's assignment, the fourier coefficients of two functions ($\exp(x)$ & $\cos(\cos(x))$) over the interval 0 to 2π were calculated using the integral definition of the fourier coefficients (integrated using scipy.integrate.quad) as well as by least square fitting (using numpy.linalg.lstsq). These two coefficients were first plotted together. The properties of the graphs obtained were noted (both in semilog-y axis and loglog axis). Then, the coefficients obtained through least square estimation were used to calcuate the values of the functions and this was plotted alongside the actual values of the functions and the difference was noted.

Formula for fourier series used:

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

Libraries Used

```
import numpy as np
import matplotlib.pyplot as plt
from numpy.linalg import lstsq
import math
from scipy.integrate import quad
```

1 Function Definition

The two functions whose fourier coefficients are to be calculated are defined in a python function.

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

The above defined functions are then plotted in the interval $[-2\pi, 4\pi)$.

```
x = np.linspace(-2*math.pi, 4*math.pi, 600)
x2 = np.linspace(0, 2*math.pi, 200)
ft = f1(x2)
exp = np.concatenate((ft,ft,ft))
plt.semilogy(x, f1(x),'r')    # exp(x) is plotted in a semilog axis.
plt.semilogy(x, exp,'b')
plt.title('Semilog plot of exp(x)')
plt.xlabel('x\nFigure 1')
plt.ylabel('log plot of $e^{x}$')
plt.legend(('Actual function', 'Expected function using fourier transform'))
plt.grid()
plt.show()
```

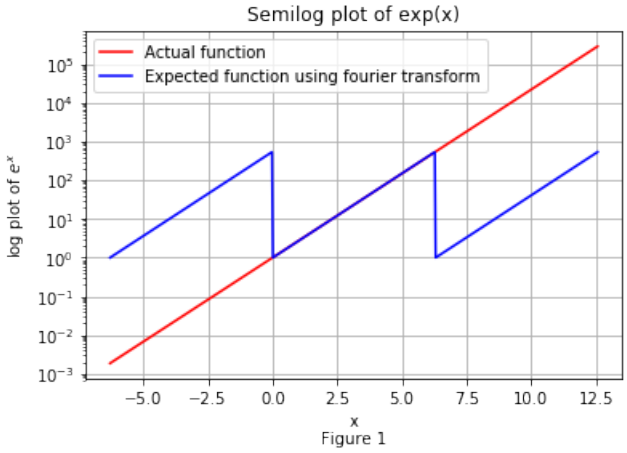


Figure 1

```
ft = f2(x2)
exp = np.concatenate((ft,ft,ft))
plt.plot(x, exp, 'g')
plt.plot(x, f2(x), 'r') # cos(cos(x)) is plotted in a linear axis.
plt.title('Plot of cos(cos(x))')
plt.xlabel('x\nFigure 2')
plt.ylabel('cos(cos(x))')
plt.legend(('Actual function', 'Expected function using fourier transform'))
plt.grid()
plt.show()
```

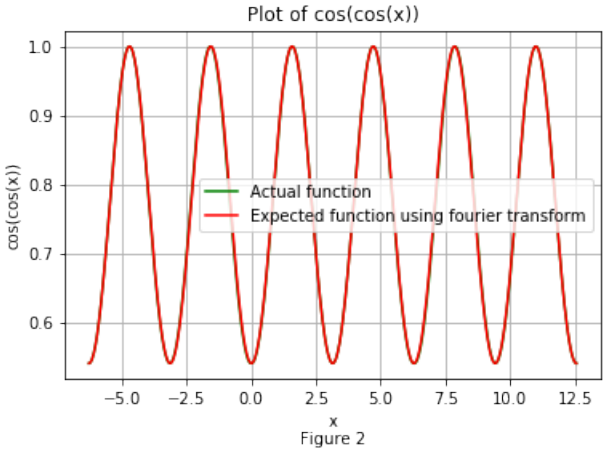


Figure 2

From the graph, we see that expected graph using fourier transform does not match completely with the graph of $\exp(x)$ whereas it does with that of $\cos(\cos(x))$ and hence $\cos(\cos(x))$ is periodic whereas $\exp(x)$ is not.

2 Fourier coefficients using integration

First, the functions to integrate according to the integral equations are defined.

The integration equations used:

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

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

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

```
def uf1(x, k):      # an calculation for f1
    return np.exp(x)*np.cos(k*x)
def vf1(x, k):      # bn calculation for f1
    return np.exp(x)*np.sin(k*x)


def uf2(x, k):      # an calculation for f2
    return np.cos(np.cos(x))*np.cos(k*x)
def vf2(x, k):      # bn calculation for f2
    return np.cos(np.cos(x))*np.sin(k*x)
```

Now, the loop to actually integrate the above functions and get the first 51 coefficients for the 2 functions are defined.

```
n = np.arange(1,52)
f1Coeff = []     #List of all the coefficients
af1 = []         #List of only an
bf1 = []         #List of only bn
f1Coeff.append(quad(f1, 0, 2*math.pi)[0] / (2*math.pi))   #Calculate a0 for f1
af1.append(f1Coeff[0])
for k in range(1,26):
    f1Coeff.append(quad(uf1, 0, 2*math.pi, args = (k))[0] / (math.pi))  #Calculate an for f1
    af1.append(quad(uf1, 0, 2*math.pi, args = (k))[0] / (math.pi))
    f1Coeff.append(quad(vf1, 0, 2*math.pi, args = (k))[0] / (math.pi))  #Calculate bn for f1
    bf1.append(quad(vf1, 0, 2*math.pi, args = (k))[0] / (math.pi))


f2Coeff = []     #List of all the coefficients
af2 = []         #List of only an
bf2 = []         #List of only bn
f2Coeff.append(quad(f2, 0, 2*math.pi)[0] / (2*math.pi))  #Calculate a0 for f2
af2.append(f2Coeff[0])
for k in range(1,26):
    f2Coeff.append(quad(uf2, 0, 2*math.pi, args = (k))[0] / (math.pi))  #Calculate an for f2
    af2.append(quad(uf2, 0, 2*math.pi, args = (k))[0] / (math.pi))
    f2Coeff.append(quad(vf2, 0, 2*math.pi, args = (k))[0] / (math.pi))  #Calculate bn for f2
    bf2.append(quad(vf2, 0, 2*math.pi, args = (k))[0] / (math.pi))
```

3 Comparing plots

The magnitude of fourier coefficients of both functions are plotted in a semilog plot as well as a log log plot and the properties of the graph obtained are noted.

```
plt.semilogy(n, np.absolute(f1Coeff), 'ro')
plt.title('Semilog plot of coefficients of exp(x)')
plt.xlabel('n\nFigure 3')
plt.ylabel('f1Coeff')
plt.grid()
plt.show()
```

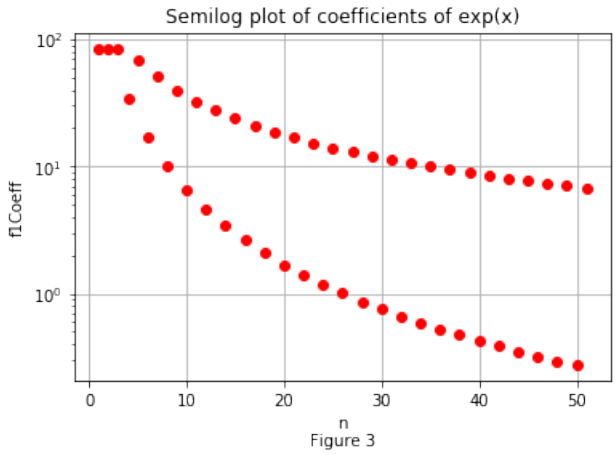


Figure 3

```
plt.loglog(n, np.absolute(f1Coeff), 'ro')
plt.title('Log log plot of coefficients of exp(x)')
plt.xlabel('n\nFigure 4')
plt.ylabel('f1Coeff')
plt.grid()
plt.show()
```

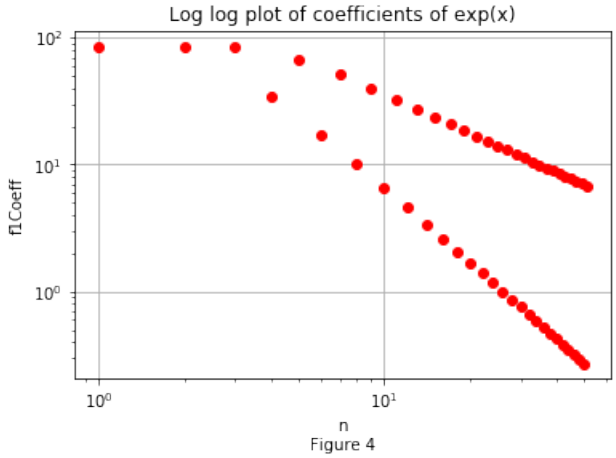


Figure 4

```
plt.semilogy(n, np.absolute(f2Coeff), 'ro')
plt.title('Semilog plot of coefficients of cos(cos(x))')
plt.xlabel('n\nFigure 5')
plt.ylabel('f2Coeff')
plt.grid()
plt.show()
```

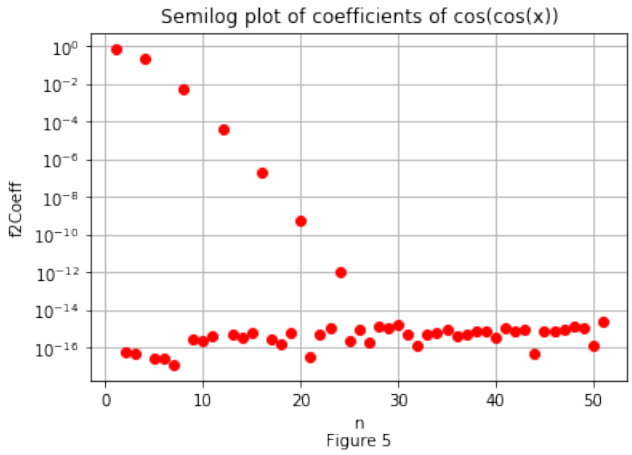


Figure 5

```
plt.loglog(n, np.absolute(f2Coeff), 'ro')
plt.title('Log log plot of coefficients of cos(cos(x))')
plt.xlabel('n\nFigure 6')
plt.ylabel('f2Coeff')
plt.grid()
plt.show()
```

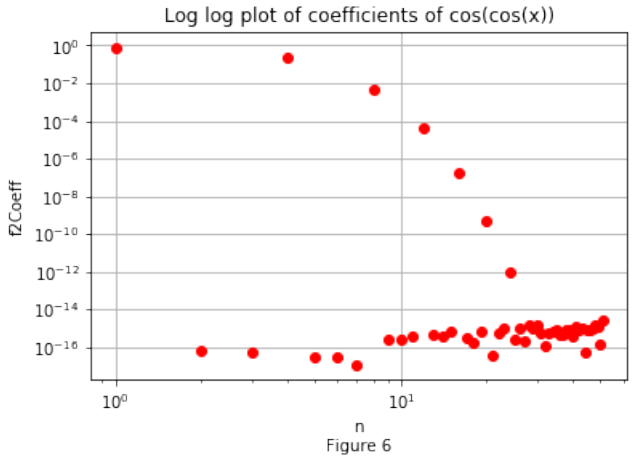


Figure 6

4 Least Squares Approximation of coefficients

The vector x is initialised from 0 to 2π in 400 steps and the actual value of the functions are evaluated at eaxh of these points and stored in a vector b. Then the 2D matrix in the matrix equivalent of the following equation is initialised and the value of the coefficients are obtained through solving the resulting matrix equation by least squares estimation.

$$a_0 + \sum_{n=1}^{25} a_n \cos(nx_i) + \sum_{n=1}^{25} b_n \sin(nx_i) \approx f(x_i)$$

```
x=np.linspace(0,2*math.pi,401)   # vector x is initialised
x=x[:-1]
b1=f1(x)                         # b1 is used to store actual value of f1 at all points in x
b2=f2(x)                         # b2 is used to store actual value of f2 at all points in x
A=np.zeros((400,51)) # The 2D matrix is initialised and assigned values in the following lines
A[:,0]=1
for k in range(1,26):
    A[:,2*k-1]=np.cos(k*x)
    A[:,2*k]=np.sin(k*x)
c1=lstsq(A,b1)[0]                # The coefficients of f1 is estimated using lstsq
c2=lstsq(A,b2)[0]                # The coefficients of f2 is estimated using lstsq
```

5 Plotting coefficients

The coefficients obtained using least squares approach is plotted alongside the coefficients obtained using integration using green dots in Figures 3,4,5 & 6.

```
plt.semilogy(n, np.absolute(f1Coeff), 'ro')
plt.semilogy(n, np.absolute(c1), 'go')
plt.title('Semilog plot of coefficients of exp(x)')
plt.legend(('Integration method', 'Least square estimation'))
plt.xlabel('n\nFigure 3')
plt.ylabel('f1Coeff')
plt.grid()
plt.show()
```

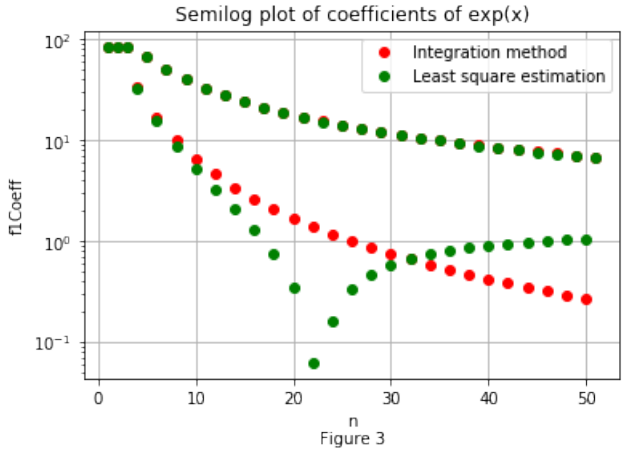


Figure 3

```
plt.loglog(n, np.absolute(f1Coeff), 'ro')
plt.loglog(n, np.absolute(c1), 'go')
plt.title('Log log plot of coefficients of exp(x)')
plt.legend(('Integration method', 'Least square estimation'))
plt.xlabel('n\nFigure 4')
plt.ylabel('f1Coeff')
plt.grid()
plt.show()
```

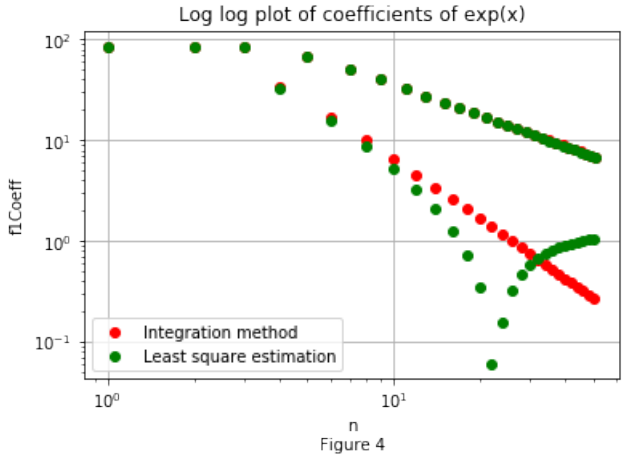


Figure 4

```
plt.semilogy(n, np.absolute(f2Coeff), 'ro')
plt.semilogy(n, np.absolute(c2), 'go')
plt.title('Semilog plot of coefficients of cos(cos(x))')
plt.legend(('Integration method', 'Least square estimation'))
plt.xlabel('n\nFigure 5')
plt.ylabel('f2Coeff')
```

```
plt.grid()
plt.show()
```

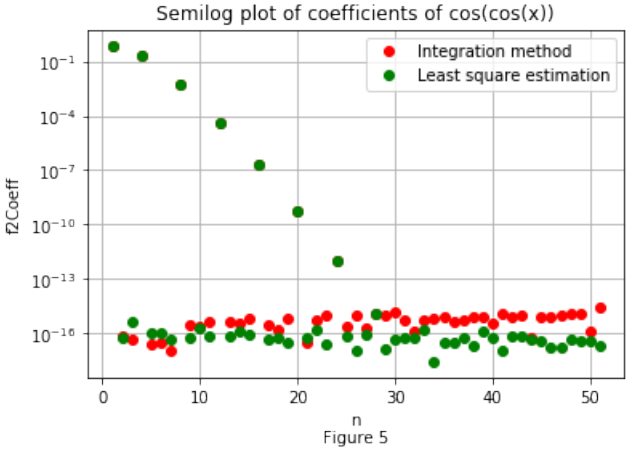


Figure 5

```
plt.loglog(n, np.absolute(f2Coeff), 'ro')
plt.loglog(n, np.absolute(c2), 'go')
plt.title('Log log plot of coefficients of cos(cos(x))')
plt.legend(('Integration method', 'Least square estimation'))
plt.xlabel('n\nFigure 6')
plt.ylabel('f2Coeff')
plt.grid()
plt.show()
```

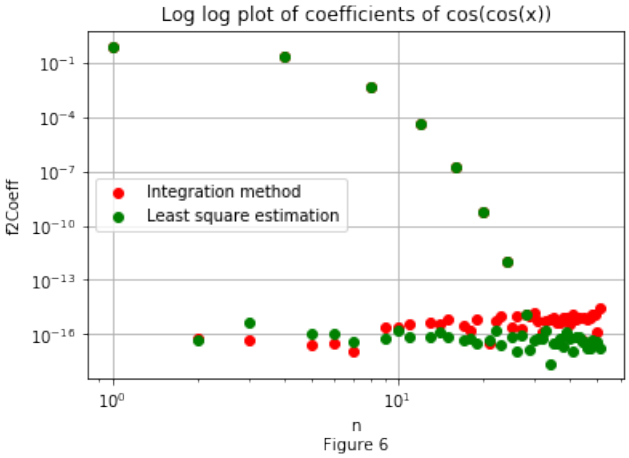


Figure 6

The maximum deviation of the calculated coefficients to the estimated coefficients are then found out.

```
print ("The maximum deviation between coefficients are -")
print ("exp(x) :", max(np.absolute(c1-f1Coeff)))
print ("cos(cos(x)) :", max(np.absolute(c2-f2Coeff)))
```

The output was found to be:

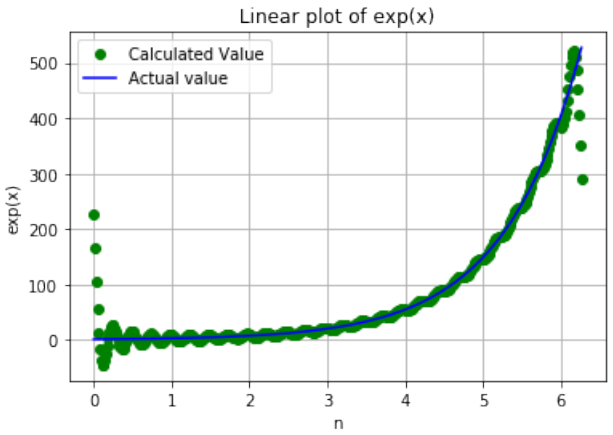
The maximum deviation between coefficients are -
exp(x) : 1.33273087034
cos(cos(x)) : 2.57586713575e-15

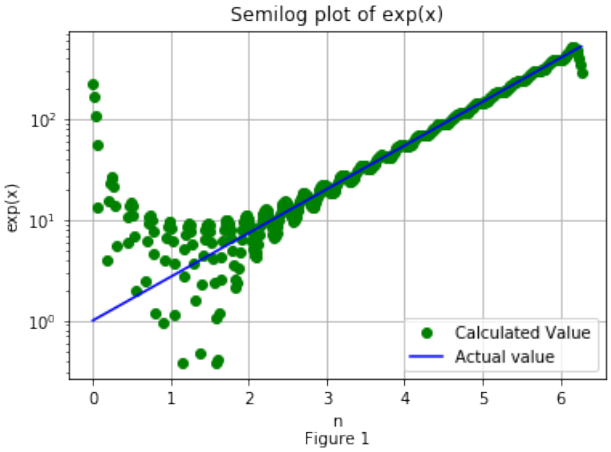
6 Compare actual and computed values of functions

From the first 51 fourier coefficients that were calculated using least squares estimation, the value of the functions in the interval is calculated and plotted alongside the actual value of the function in that interval.

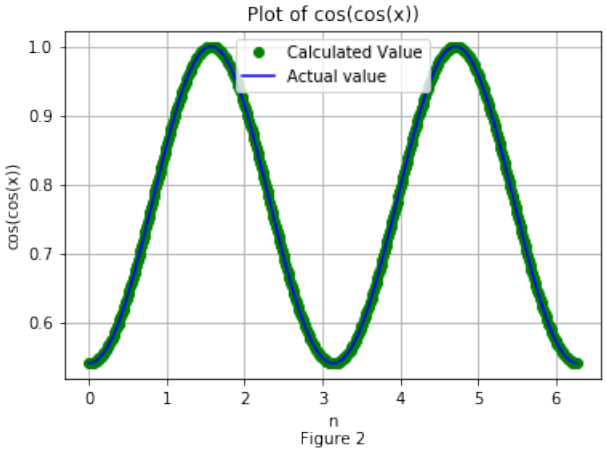
```
plt.plot(x, np.matmul(A,c1), 'go')
plt.plot(x, f1(x), 'b')
```

```
plt.title('Linear plot of exp(x)')
plt.legend(('Calculated Value', 'Actual value'))
plt.xlabel('n')
plt.ylabel('exp(x)')
plt.grid()
plt.show()   # Compares the linear plot of exp(x)
plt.semilogy(x, np.matmul(A,c1), 'go')
plt.semilogy(x, f1(x), 'b')
plt.title('Semilog plot of exp(x)')
plt.legend(('Calculated Value', 'Actual value'))
plt.xlabel('n\nFigure 1')
plt.ylabel('exp(x)')
plt.grid()
plt.show()  # Compares the semilog plot of exp(x)
```






```
plt.plot(x, np.matmul(A,c2), 'go')
plt.plot(x, f2(x), 'b')
plt.title('Plot of cos(cos(x))')
plt.legend(('Calculated Value', 'Actual value'))
plt.xlabel('n\nFigure 2')
plt.ylabel('cos(cos(x))')
plt.grid()
plt.show()
```



7 Answers to questions

Qn.3: From the initial graphs of Figures 3,4,5 and 6, it is observed that:

1. The b_n coefficients in the second function are nearly 0 as the function $\cos(\cos(x))$ is an even function and therefore depends more strongly on the cosine component of the fourier series (which is even) than the sine component (which is odd). Hence the sine coefficients (b_n) are nearly 0.
2. The coefficients of $\exp(x)$ will depend on higher frequencies also as it is not a periodic function and therefore to estimate it, even to a reasonable accuracy, a large number of high frequency components would also be required whereas $\cos(\cos(x))$ is a periodic function as well as sinusoidal in nature and therefore can be estimated accurately even with low frequency components and hence the latter's coefficients decay quickly with n whereas the former's does not.
3. The coefficients of $\exp(x)$ is of the form n^a and therefore looks linear only in a loglog plot whereas that of $\cos(\cos(x))$ is of the form $\exp(a)$ and therefore looks linear in a semilog plot.

Qn.6: From the final graphs - Figures 3,4,5 and 6, we see that coefficients found using the two methods don't agree and they shouldn't also. This is because the ones calculated using integration are the actual values (with minimal error inherent in the quad function) whereas for calculating using least square estimation, the actual values of the functions were equated with the fourier series expansion containing just the first 25 terms of each summation and therefore compounding this error with the one inherent in least square approximation, we get coefficients that deviate from the actual values.

Qn.7: From the final graphs of Figures 1 and 2, we see that there is significant deviation in Figure 1 whereas almost no deviation in Figure 2. This is because, as mentioned earlier, as $\exp(x)$ is not periodic, it depends on higher frequencies of fourier series also to converge to the actual function whereas only the first 25 frequencies were used to estimate using least square estimation and hence significant deviation is there. In the case of $\cos(\cos(x))$ though, as it is periodic and higher frequency coefficients were very low, with the first 25 frequencies itself the series converges to a very close estimate of the actual function. Also, the sudden changes at the end of Figure 1 is due to Gibb's phenomenon in which sudden discontinuities in graph cannot be modeled by sine or cosine functions which are continuous in nature.

8 Inferences

Thus, from the above plots, it is observed that a periodic function which is sinusoidal in nature can be predicted accurately using the low frequency components of its fourier series whereas to predict a non-periodic function, higher frequency components are also necessary. Also, for what its worth, least square approximation does indeed provide a very good estimate of the actual solution of the matrix equation (as seen in the graph of $\cos(\cos(x))$). The reason it failed in that of $\exp(x)$ is mainly due to the fact that only the low frequency components of its fourier series were taken.