EE2703 : Applied Programming Lab
Assignment 4
Fourier Approximations

Chagari Koushal Kumar Reddy
EE20B023

March 1, 2022

Contents

1 Aim

The aim of this assignment is to :

- Plot the functions e^x and $\cos(\cos(x))$ for $x \in [-4\pi, 2\pi)$.
- Find the first 51 coefficients of both functions using integration and least squares approximation and plot them.
- Plot the function values using both the set of coefficients for $x \in [-4\pi, 2\pi)$.

2 Theory

2.1 Functions: e^x and $\cos(\cos(x))$

These two functions are to be fitted using the fourier series coefficients over the interval $[0, 2\pi]$ as follows:

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

where the coefficients are given by the formulae:

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

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

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

2.2 Fourier coefficients vector

The first 51 coefficients of the given functions are calculated using the integration formulae given above. The column vector is of the form:

$$\begin{pmatrix} a_0 \\ a_1 \\ b_1 \\ \dots \\ a_{25} \\ b_{25} \end{pmatrix}$$

We know from the fourier series coefficients that:

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

Hence in matrix form the equation would be:

$$\begin{pmatrix} 1 & \cos(x_1) & \sin(x_1) & \dots & \cos(25x_1) & \sin(25x_1) \\ 1 & \cos(x_2) & \sin(x_2) & \dots & \cos(25x_2) & \sin(25x_2) \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & \cos(x_{400}) & \sin(x_{400}) & \dots & \cos(25x_{400}) & \sin(25x_{400}) \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ b_1 \\ \dots \\ a_{25} \\ b_{25} \end{pmatrix} \approx \begin{pmatrix} f(x_1) \\ f(x_2) \\ \dots \\ f(x_{400}) \end{pmatrix}$$

By using these matrices we can find out the coefficients by least squares method also.

2.3 Least Squares Approximation

We know the integration formulae for the fourier coefficients and we used it to find the approximate function values in the previous section.
We could also find a least squares solution for the coefficients vector by forming the matrix and forming the function values vector beforehand.

$$\begin{pmatrix} 1 & \cos(x_1) & \sin(x_1) & \dots & \cos(25x_1) & \sin(25x_1) \\ 1 & \cos(x_2) & \sin(x_2) & \dots & \cos(25x_2) & \sin(25x_2) \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & \cos(x_{400}) & \sin(x_{400}) & \dots & \cos(25x_{400}) & \sin(25x_{400}) \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ b_1 \\ \dots \\ a_{25} \\ b_{25} \end{pmatrix} \approx \begin{pmatrix} f(x_1) \\ f(x_2) \\ \dots \\ f(x_{400}) \end{pmatrix}$$

Here the matrix with the sines and cosines is called 'A', the coefficients vector 'c' and the function values vector $'b'$. Once we have matrices 'A', 'b', we can find the least sqaures fit for vector 'c'.
The values we get for coefficients in this manner won't be the exact same values we obtained using the integration formulae. But these coefficient values minimize the least square error.

3 Procedure

According to the assignment, the following objectives must be completed:

1. Functions e^x and $\cos(\cos(x))$ should be plotted with 'x' ranging in the interval $[-4\pi, 2\pi)$. The expected Fourier plots should also be plotted in the respective figures. Two user defined functions must also be created to return the values of e^x, $\cos(\cos(x))$ for a given vector 'x'.

2. The first 51 Fourier coefficients must be obtained in the following vector form:

$$\begin{pmatrix} a_0 \\ a_1 \\ b_1 \\ \dots \\ a_{25} \\ b_{25} \end{pmatrix}$$

   These Fourier coefficients are to be obtained using the "quad" function of the "scipy.integrate" module using the two functions:

$$u(x,k) = f(x)\cos(kx)$$

$$v(x,k) = f(x)\sin(kx)$$

   Python "quad" function:

```
from scipy.integrate import quad
# Library that contains the integrator

integ = quad(u,0,2*np.pi,args=(k))
# Quad function is the integrator function
```

3. The Fourier coefficients obtained from the integration formulae should be plotted using *semilog* and *loglog* plot functions of "matplotlib.pyplot" module with respect to their indices.

   Since we need to plot the logarithm of the Fourier coefficients with respect to their indices, logarithm of absolute values of the Fourier coefficients are plotted since the fourirer coefficients might be negative sometimes.

4. The Fourier coefficients should now be calculated using another method, namely the least squares approach. The matrix equation is:

$$\begin{pmatrix} 1 & \cos(x_1) & \sin(x_1) & \dots & \cos(25x_1) & \sin(25x_1) \\ 1 & \cos(x_2) & \sin(x_2) & \dots & \cos(25x_2) & \sin(25x_2) \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & \cos(x_{400}) & \sin(x_{400}) & \dots & \cos(25x_{400}) & \sin(25x_{400}) \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ b_1 \\ \dots \\ a_{25} \\ b_{25} \end{pmatrix} = \begin{pmatrix} f(x_1) \\ f(x_2) \\ \dots \\ f(x_{400}) \end{pmatrix}$$

with $x \in [0, 2\pi)$. We can calculate the least sqaures solution using the python function "lstsq" of the "scipy.linalg" module. Here we are considering the number of samples as 400.

Python "lstsq" function:

```
import scipy
# Library that contains the lstsq function

exp_lstsq_coeff = scipy.linalg.lstsq(A,b)[0]
# lstsq function is the least squares function
```

5. The coefficients obtained by integration and by least squares approximation should be compared and the largest deviation should also be printed.

6. The function values should be found using the Fourier series equation:

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

The function values obatined using both sets of coefficients(Integration and least squares approximation) are to be compared to the expected fourier plots.

4 Code and Functions

4.1 Functions *exp_(x)* and *cos_cos_(x)*

```
1  def exp_(val):
2      return np.exp(val)
3
4  def cos_cos_(val):
5      return np.cos(np.cos(val))
```

Both these functions take an input array (or scalar) 'x' and returns the corresponding function values as a output array (or scalar).

For example if the input array 'x' is:

$$\begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix}$$

then the output array for the *exp_(x)* function is:

$$\begin{pmatrix} e^1 \\ e^2 \\ e^3 \\ e^4 \end{pmatrix}$$

4.2 Function *fourier_coeff(n,func)*

```
1    def fourier_coeff(n,func):
2        coeff = np.zeros(n)
3        f_cos = lambda x,k : func(x)*np.cos(k*x)
4        f_sin = lambda x,k : func(x)*np.sin(k*x)
5        coeff[0] = (quad(func,0,2*pi))[0]/(2*pi)
6        # For calculating the coefficients a1,a2,a3 ..., a1 - 1, a2 - 3, a3 - 5
7        for i in range(1,n,2):
8            coeff[i] = ((quad(f_cos,0,2*pi,args = (i//2 + 1)))[0])/pi
9        for i in range(2,n,2):
10           coeff[i] = ((quad(f_sin,0,2*pi,args = (i/2)))[0])/pi
11   return coeff
```

This function does the integration process and returns the coefficients. The parameters passed to the function are:

- n = Number of Fourier coefficients to be calculated
- func = Function whose Fourier coefficients are calculated

4.3 Function *matrix_gen(num,x)*

```
1    def matrix_gen(num,x):
2        # Used to generate A matrix in the question
3        n = x.shape[0]
4        mat = np.ones((n,1))
5        for i in range(1,num+1):
6            mat = np.c_[mat,np.cos(i*x)]
7            mat = np.c_[mat,np.sin(i*x)]
8        return mat
```

This function takes a column vector 'x' and returns the following matrix:

$$\begin{pmatrix} 1 & \cos(x_1) & \sin(x_1) & \dots & \cos(px_1) & \sin(px_1) \\ 1 & \cos(x_2) & \sin(x_2) & \dots & \cos(px_2) & \sin(px_2) \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & \cos(x_n) & \sin(x_n) & \dots & \cos(px_n) & \sin(px_n) \end{pmatrix}$$

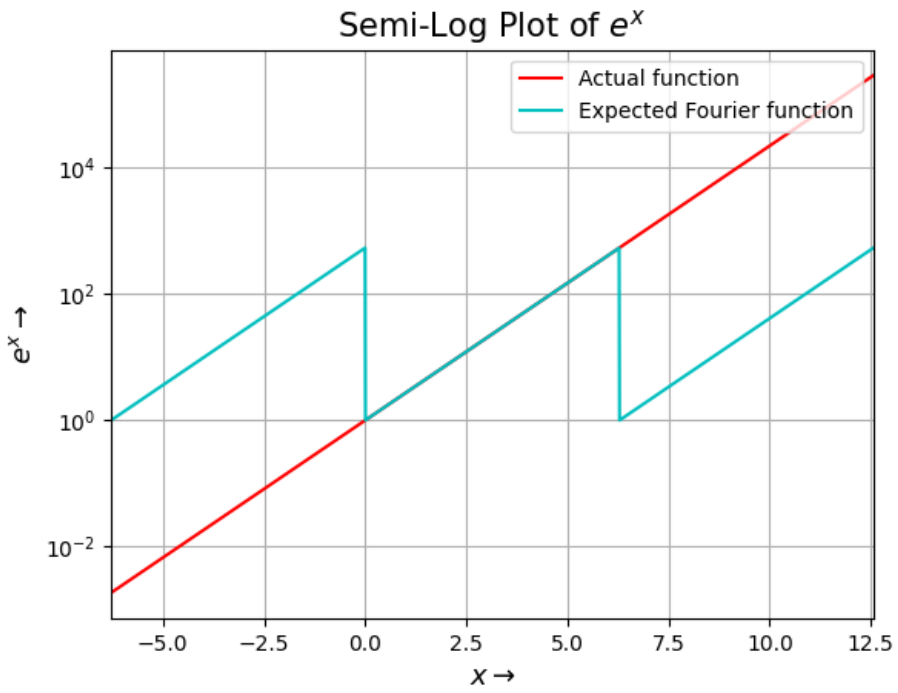
where 'p' is the maximum index of the coefficient the user desires. The arguments are:

- num = p (or) the maximum index of the coefficient
- x = Column vector containing the discrete values of the independent variable x.

5 Observations and Plots

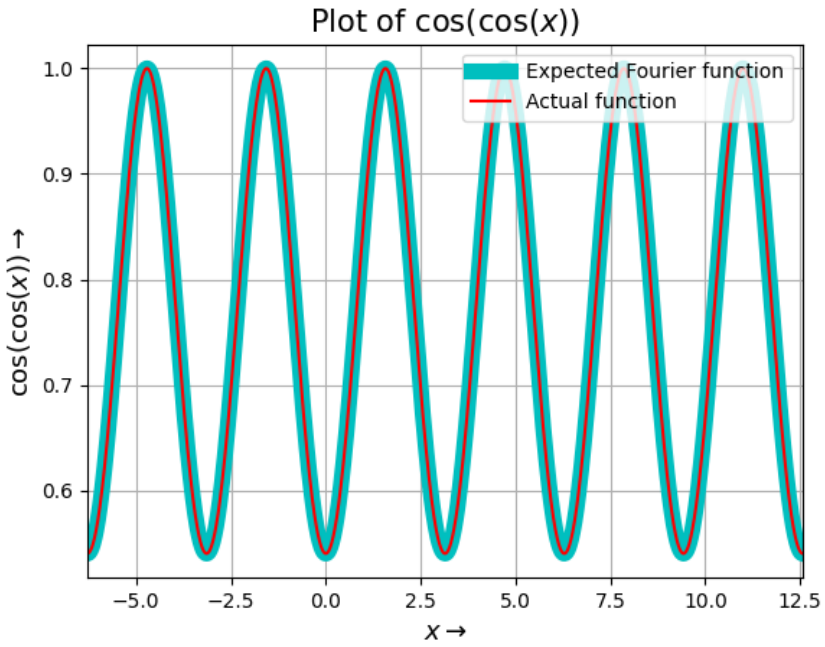
5.1 Exact vs Expected Fourier plots

5.1.1 Figure 1: For e^x function:



Since the fourier coefficients are calculated over an interval $[0, 2\pi)$, the function generated by fourier coefficients is also periodic with period 2π

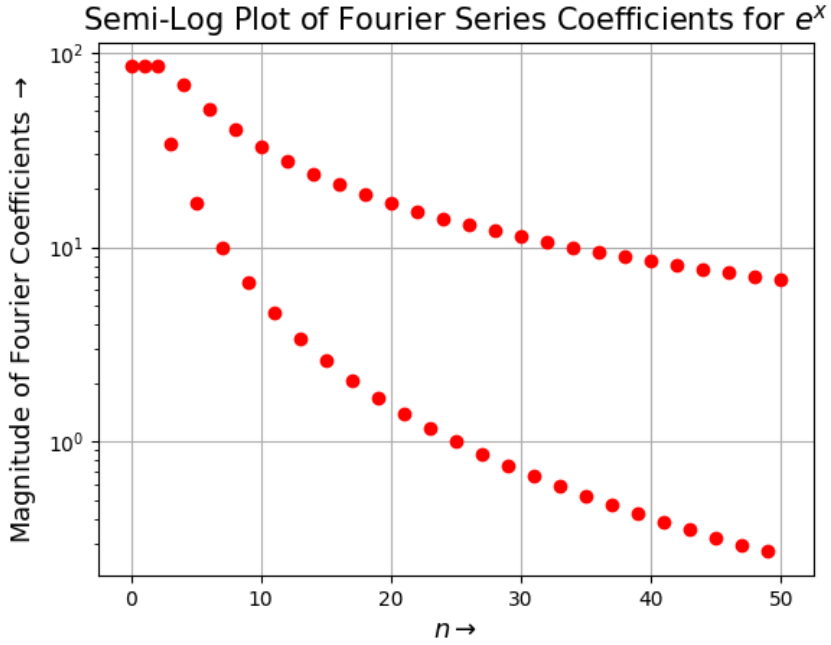
5.1.2 Figure 2: For $\cos(\cos(x))$ function:



$\cos(\cos(x))$ is already periodic with period π. Hence the expected fourier plot is exactly overlapping with the exact function plot.

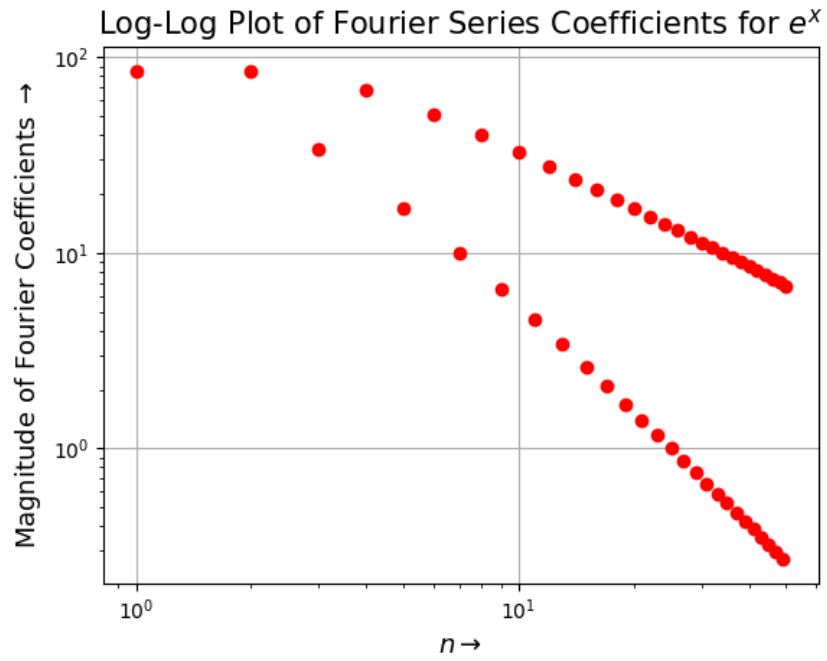
5.2 Fourier coefficients obtained by integration

5.2.1 Figure 3: Semi-logy plot for the integration coefficients of e^x:



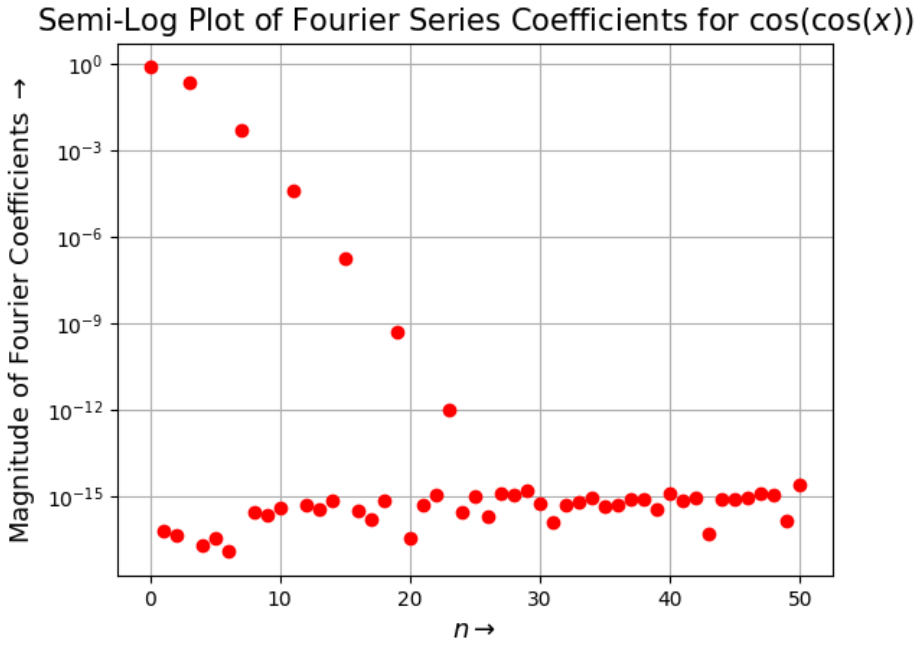
For function e^x, the coefficient magnitudes seem to decrease with n in the semilog plot

5.2.2 Figure 4: Loglog plot for the integration coefficients of e^x:



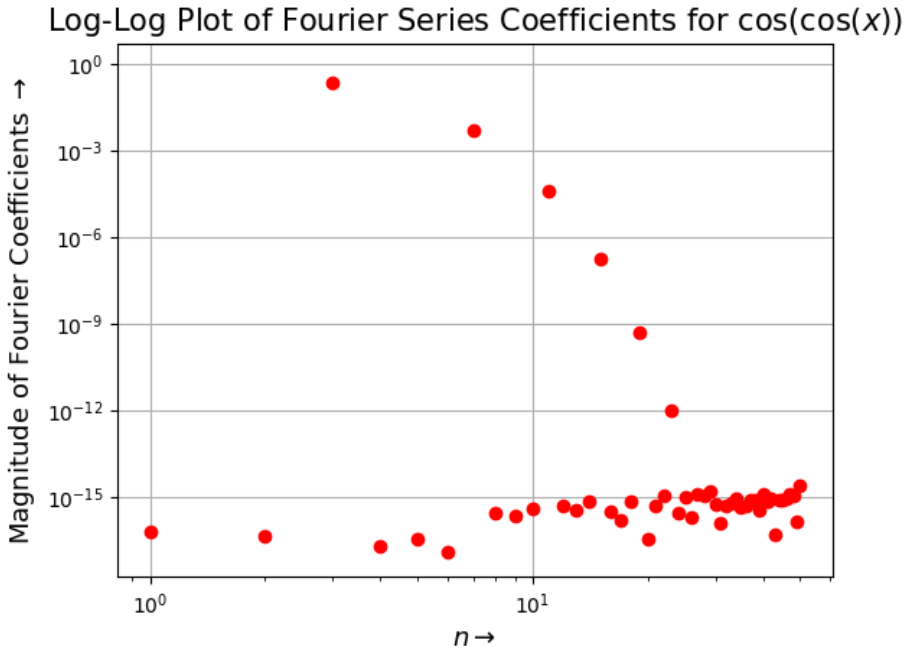
For function e^x, the coefficient magnitudes seem to decrease linearly with n in the loglog plot

5.2.3 Figure 5: Semi-logy plot for the integration coefficients of $\cos(\cos(x))$:



For function $\cos(\cos(x))$, the coefficient magnitudes seem to decrease linearly with n in the semilog plot

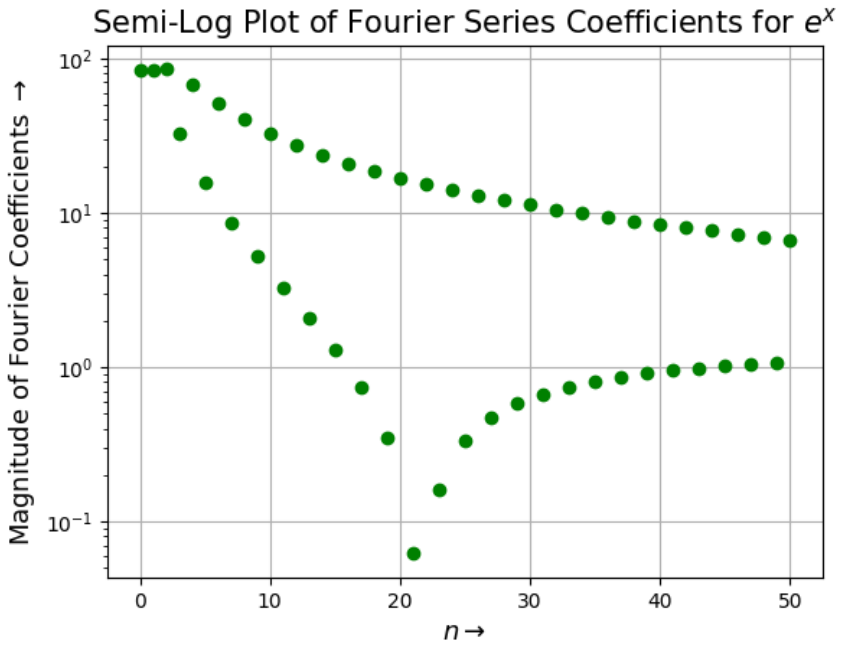
5.2.4 Figure 6: Loglog plot for the integration coefficients of $\cos(\cos(x))$:



For function $\cos(\cos(x))$, the coefficient magnitudes seem to decrease with n in the loglog plot

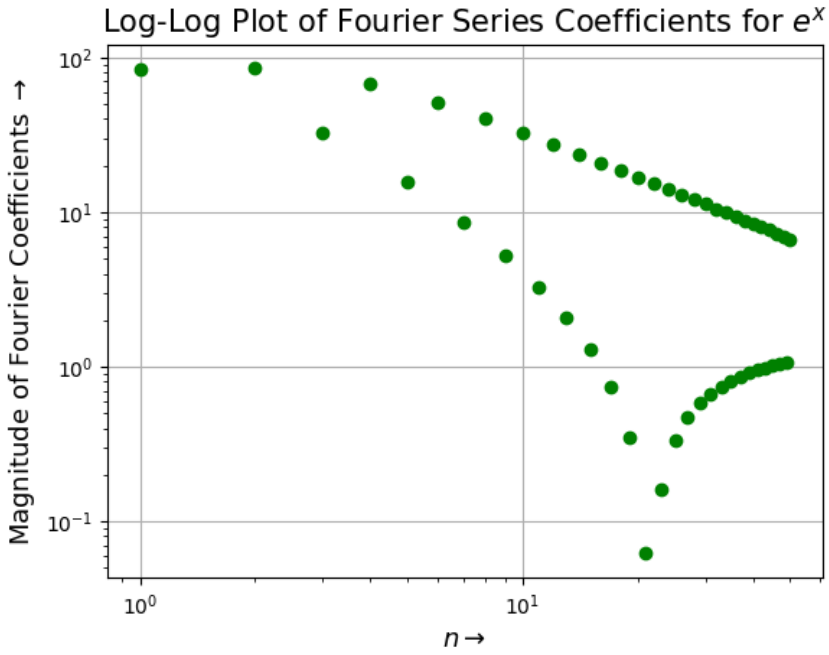
5.3 Fourier coefficients obtained by least squares method

5.3.1 Figure 7: Semi-logy plot for the least square coefficients of e^x:



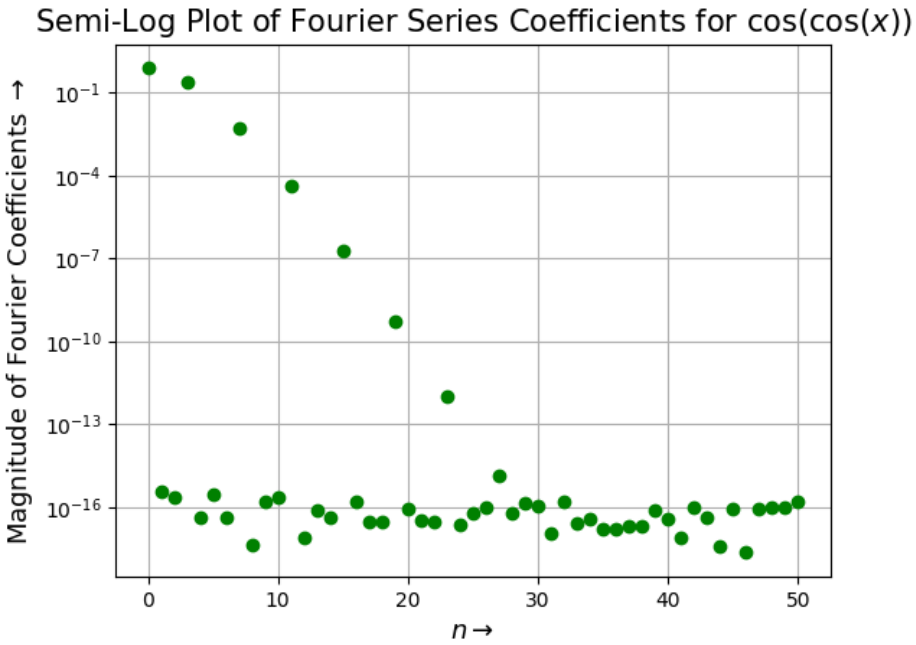
For function e^x, the coefficient magnitudes seem to decrease with n in the semilog plot

5.3.2 Figure 8: Loglog plot for the least square coefficients of e^x:



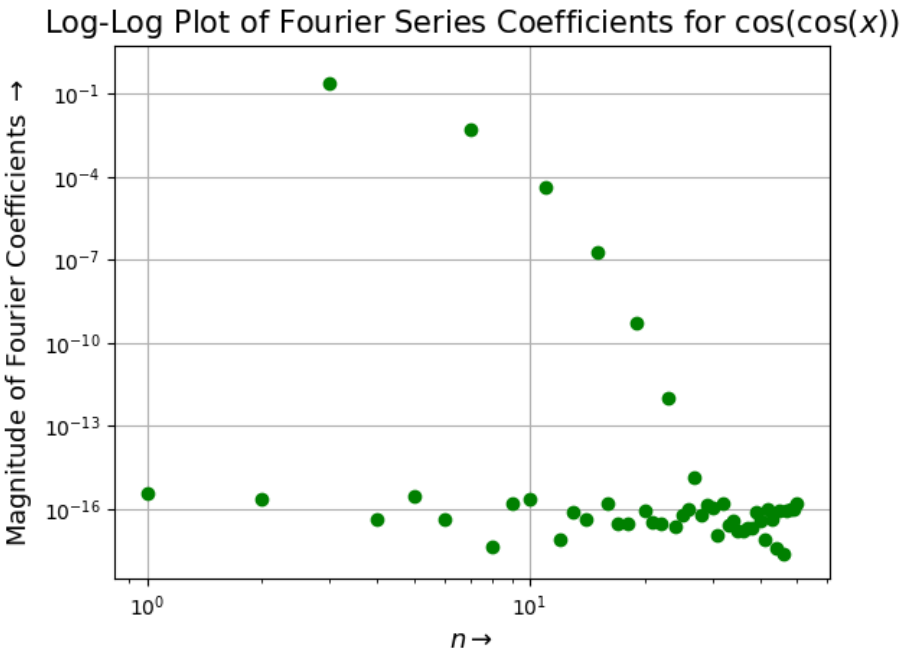
For function e^x, the coefficient magnitudes seem to decrease linearly with n in the loglog plot

5.3.3 Figure 9: Semi-logy plot for the least square coefficients of $\cos(\cos(x))$:



For function $\cos(\cos(x))$, the coefficient magnitudes seem to decrease linearly with n in the semilog plot

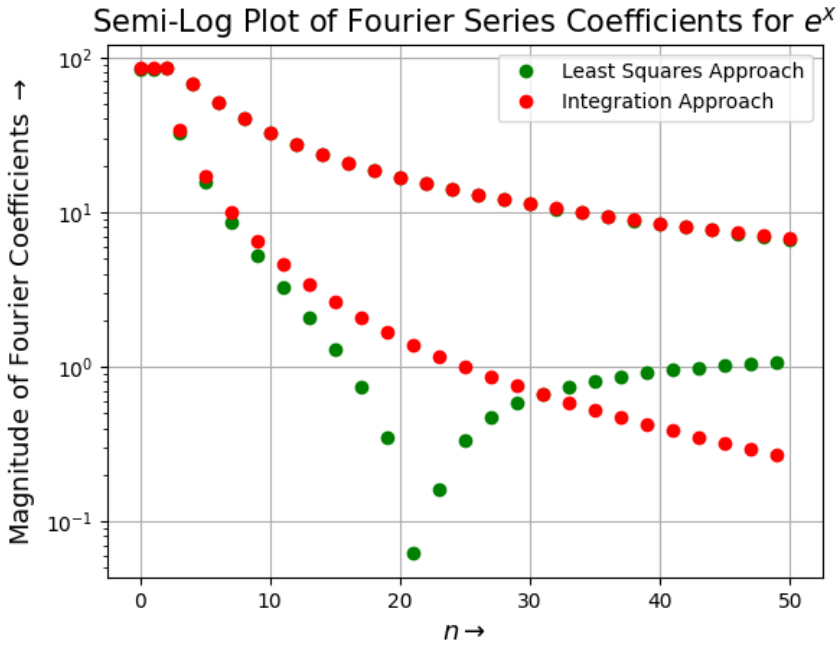
5.3.4 Figure 10: Loglog plot for the least square coefficients of $\cos(\cos(x))$:



For function $\cos(\cos(x))$, the coefficient magnitudes seem to decrease with n in the loglog plot

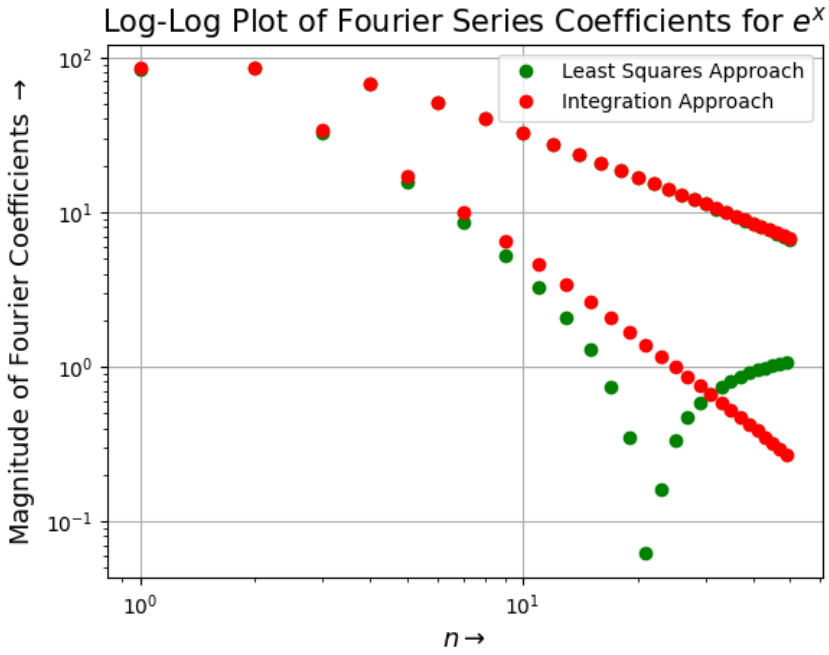
5.4 Fourier coefficients obtained by integration vs least squares approximation:

5.4.1 Figure 11: Semi-logy plot for the coefficients of e^x:



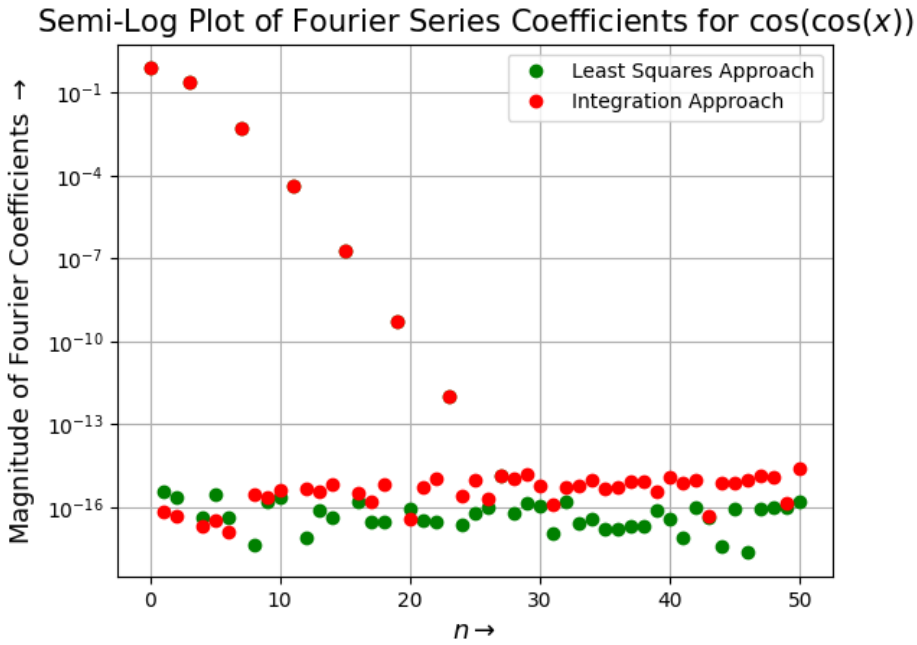
For function e^x, the coefficient magnitudes seem to decrease with n in the semilog plot

5.4.2 Figure 12: Loglog plot for the coefficients of e^x:



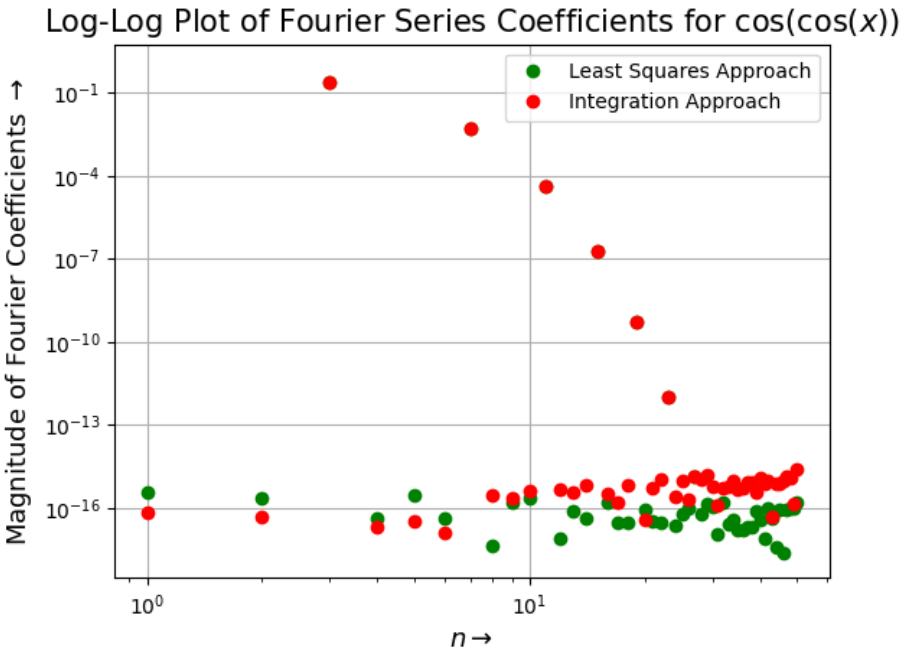
For function e^x, the coefficient magnitudes seem to decrease linearly with n in the loglog plot

5.4.3 Figure 13: Semi-logy plot for the coefficients of $\cos(\cos(x))$:



For function $\cos(\cos(x))$, the coefficient magnitudes seem to decrease linearly with n in the semilog plot

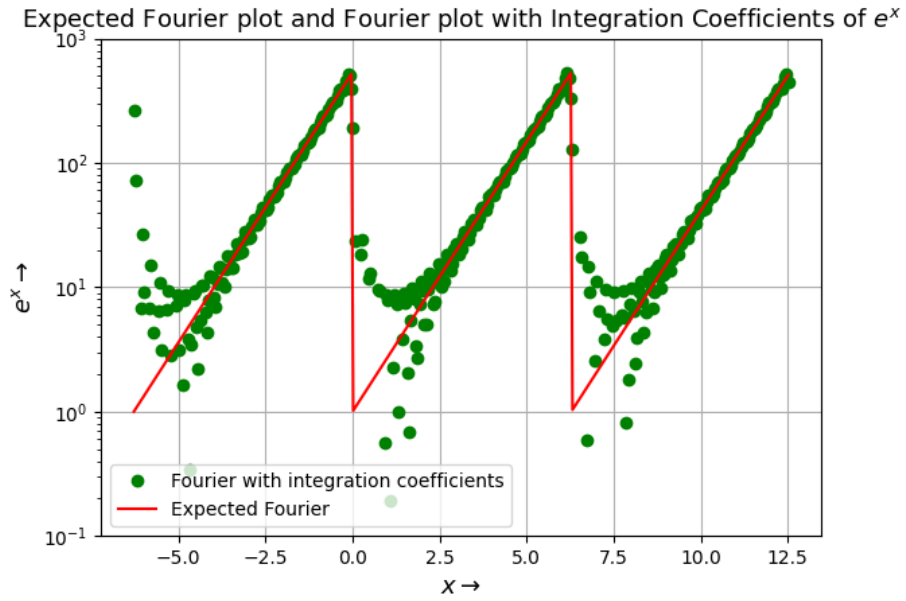
5.4.4 Figure 14: Loglog plot for the coefficients of $\cos(\cos(x))$:



For function $\cos(\cos(x))$, the coefficient magnitudes seem to decrease with n in the loglog plot

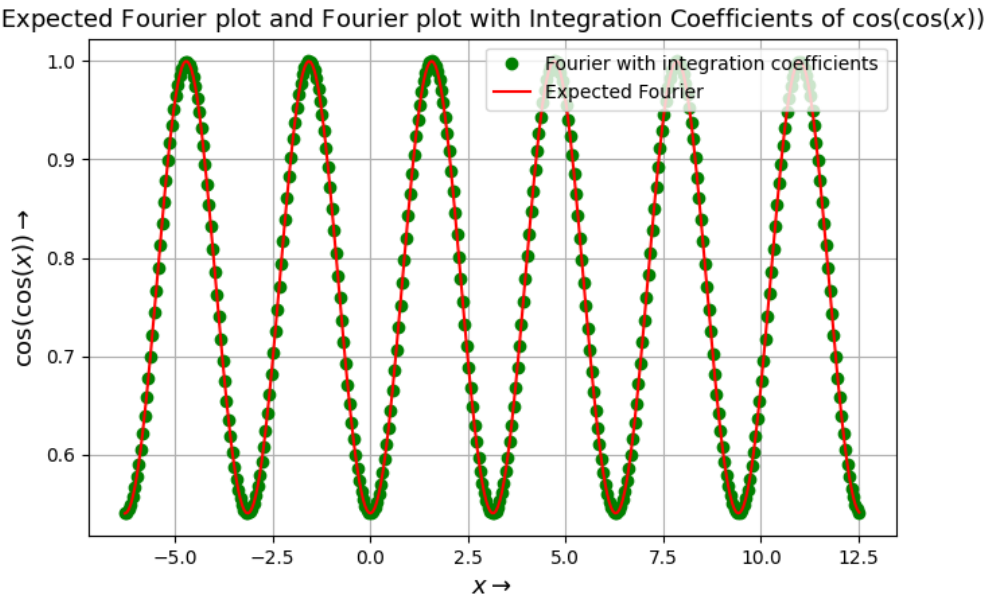
5.5 Expected Fourier plot along with Fourier plot obtained using integration coefficients

5.5.1 Figure 15: For e^x function:



The function plot generated using fourier coefficients (Integration method) almost match the expected fourier plot, except at the discontinuous points of e^x

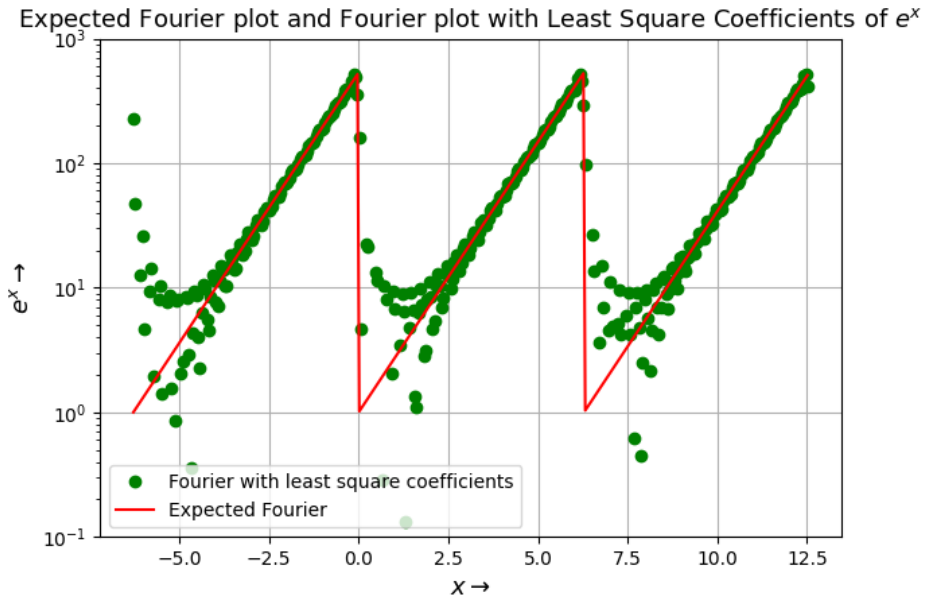
5.5.2 Figure 16: For $\cos(\cos(x))$ function:



The function plot generated using fourier coefficients (Integration method) matches the expected fourier plot almost exactly

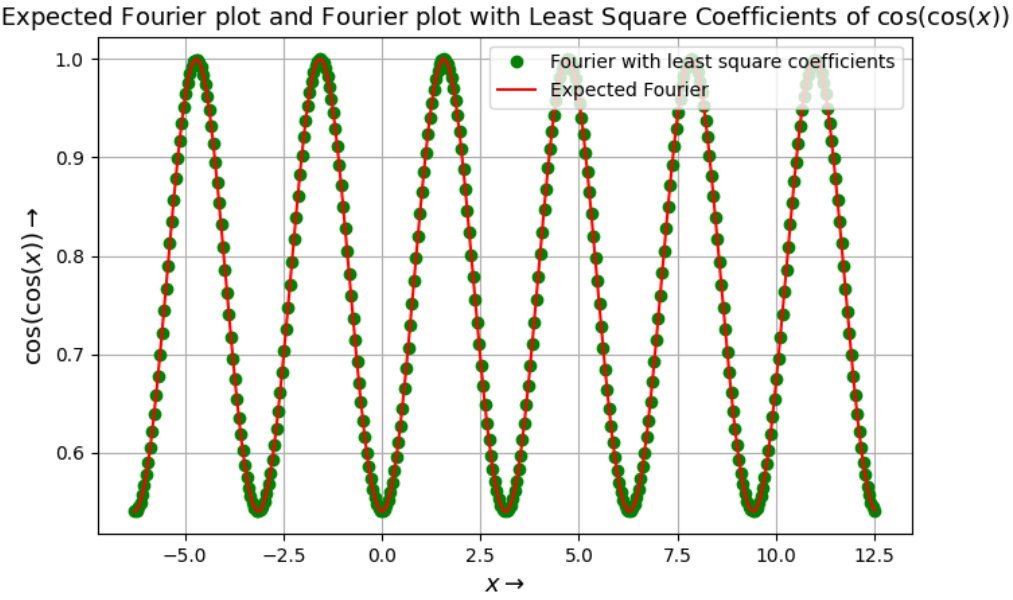
5.6 Expected Fourier plot along with Fourier plot obtained using least square coefficients

5.6.1 Figure 17: For e^x function:



The function plot generated using fourier coefficients (Least squares method) almost match the expected fourier plot, except at the discontinuous points of e^x

5.6.2 Figure 18: For $\cos(\cos(x))$ function:



The function plot generated using fourier coefficients (Least sqaures method) matches the expected fourier plot almost exactly

6 Results and Conclusions

6.1 Conclusions for question 3

a. We know that

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

which translates to

$$b_n = \int_0^{2\pi} \cos(\cos(2\pi - x))sin(n(2\pi - x))dx$$

because of the definite integral identity

$$\int_a^b f(x)dx = \int_a^b f(a + b - x)dx$$

Hence,

$$b_n = -\int_0^{2\pi} \cos(\cos(x))sin(nx)dx \tag{2}$$

because $\cos(2\pi - x) = \cos(x)$ and $\sin(n(2\pi - x)) = -\sin(nx)$
Adding equations (1) and (2), we get $2b_n = 0$. Hence $b_n = 0$. Theoretically $b_n = 0$ for all n. However because of computer precision the values are not exactly zero but very low (in the order of 10^{-17}). Hence they decay faster as compared to a_n values.

b. For e^x:

$$a_0 = \frac{e^{2\pi} - 1}{2\pi}$$

$$a_n = \frac{e^{2\pi} - 1}{\pi(n^2 + 1)}$$

$$b_n = \frac{-n(e^{2\pi} - 1)}{\pi(1 + n^2)}$$

For $\cos(\cos(x))$, the exact formulae for the coefficients are not known. So we use the Taylor series approximation:

$$\cos(\cos(x)) = 1 - \frac{\cos^2 x}{2!} + \frac{\cos^4 x}{4!} - \frac{\cos^6 x}{6!} + \ldots, |\cos(x)| < 1$$

From the above expansion, if we express the $\cos^2 x$ terms in terms of $\cos 2x$, $\cos^2 2x$ in terms of $\cos 4x$ and so on, we see that the fourier coefficients decay exponentially. Whereas the fourier coefficients of e^x decay as $\frac{1}{n^2}$. Hence the coefficients of e^x do not decay as quickly as the coefficients of $\cos(\cos(x))$.

c. From part (b) we saw:

$$a_n = \frac{e^{2\pi} - 1}{\pi(n^2 + 1)}$$

$$b_n = \frac{-n(e^{2\pi} - 1)}{\pi(1 + n^2)}$$

Hence for large values of 'n', we can safely assume that the denominator is approximately n^2. Hence, we have:

$$\log(|a_n|) = \log(\frac{e^{2\pi} - 1}{\pi}) - 2\log(n)$$

$$\log(|b_n|) = \log(\frac{e^{2\pi} - 1}{\pi}) - \log(n)$$

That's why we almost have a linear plot in Figure 4, which is the loglog plot of the fourier coefficients of e^x.

The coefficients of $\cos(\cos(x))$ converge to an exponentially decaying function. That's why we have a linear plot in Figure 5, which is the semilogy plot of the fourier coefficients of $\cos(\cos(x))$.

6.2 Conclusions for question 6

The fourier coefficients obtained by the integration and least squares method are different from each other because they have been calculated using different methods. They will have deviations from one another. However both of them fit the function quite accurately as seen in Figures 15, 16, 17, 18. The python code snippet used to calculate the deviation is:

```
1     exp_coeff_diff = abs(exp_fourier_coeff - exp_lstsq_coeff)
2     print("The largest deviation between the fourier coefficients and least
      square coefficients of exp(x) is %f" %(np.amax(exp_coeff_diff)))
3
4     cos_cos_coeff_diff = abs(cos_cos_fourier_coeff - cos_cos_lstsq_coeff)
5     print("The largest deviation between the fourier coefficients and least
      square coefficients of cos(cos(x)) is ",end = '')
6     print(np.amax(cos_cos_coeff_diff))
```

The results obtained are:

- The largest deviation between the fourier coefficients and least square coefficients of e^x is 1.332731
- The largest deviation between the fourier coefficients and least square coefficients of $\cos(\cos(x))$ is 2.758669e-15

6.3 Conclusions for question 7

In figures 16 and 18 (Plot of expected fourier for $\cos(\cos(x))$ vs plot obatined from fourier coefficients), there is no much deviation and the green circles almost lie on the expected fourier plot. However we have a lot of deviations from the expected fourier plots in figures 15 and 17 (Plot of expected fourier for e^x vs plot obtained from fourier coefficients). Especially the green circles are randomly placed near the discontinuities of e^x. This is a result of Gibbs phenomena. It says that the nth partial sum of the Fourier series will have large oscillations at the jump discontinuities. As a result of this e^x was showing large variations at jumps.