EE2703 : Applied Programming Lab
Week4 : Fourier Approximations

Aditya Suresh
EE20B004

March 11, 2022

AIM

- To plot and visualise fourier transform of e^x and $cos(cos(x))$
- To plot the corresponding graphs of true function and fourier approximation
- Obtain 51 coefficients(26 a_n's and 25 $b_n's$)
- find the above coefficient via **Least Squares approach** and compare them with true coefficients
- plot the function formed by the fourier transform via the coefficients

Input short summary of the question here which

Theory

We will fit two functions,e^x and $cos(cos(x))$ over the interval [0,2) using the fourier series

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

where ,

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

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

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

Define Python functions for e^x and $cos(cos(x))$

1. The following code is used as a function for the functions e^x and $cos(cos(x))$:

```
def exp(x):                    # function to find exponential of an array
    return np.exp(x)

def coscos(x):                 # Function to find cos(cos()) of an array
    return np.cos(np.cos(x))
```

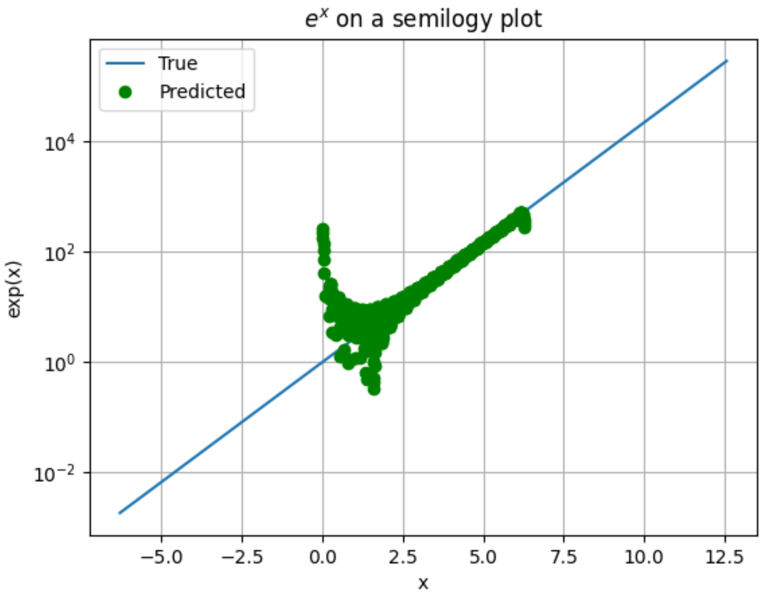

Figure 1: e^x

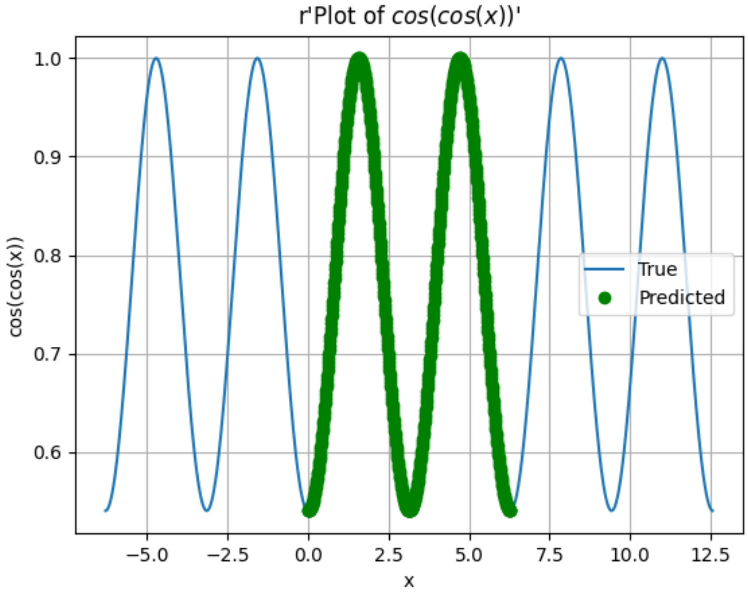

Figure 2: $cos(cos(x))$

2. The blue straight line represents the true value of e^x whereas the green dots represent the fourier approximation with 51 coefficients

3. As can be seen e^x is **Non periodic** and $cos(cos(x))$ is **periodic** so are the graphs of their respective fourier transforms.

Obtaining fourier transforms:

1. We obtain the 51 fourier coefficients($26a_n'sand25b_n's$) via the following code :

```
fcosine_1 = lambda x : exp(x) * cos(i*x)
fsine_1 = lambda x :   exp(x) * sin(i*x)

fcosine_2 = lambda x : coscos(x) * cos(i*x)
fsine_2 = lambda x :   coscos(x) * sin(i*x)

for i in range(n): # finding an's and bn's of coscos and exp via quad
    an = (quad(fcosine_1 , 0.0, 2*PI)[0]) / PI
    bn = (quad(fsine_1 , 0.0, 2*PI)[0]) / PI
```

```
an_1 = (quad(fcosine_2 , 0.0, 2*PI)[0]) / PI
bn_1 = (quad(fsine_2 , 0.0, 2*PI)[0]) / PI

An_exp.append(an)
Bn_exp.append(bn)
An_coscos.append(an_1)
Bn_coscos.append(bn_1)
```

Plots of fourier coefficients of e^x and $cos(cos(x))$

1. The following are the corresponding plots of coefficients in semilog and loglog

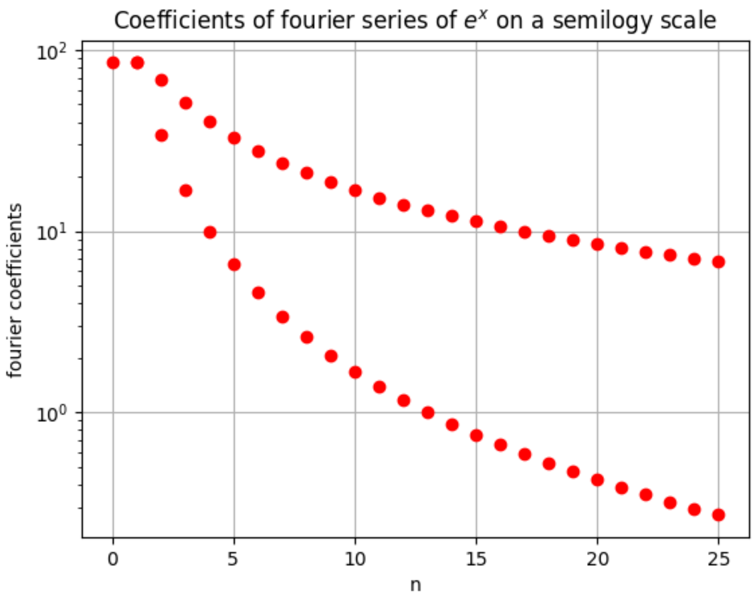

Figure 3: Coefficients of e^x on semilog

The bottom graph corresponds toa_n and the above to b_n

The bottom graph corresponds toa_n and the above to b_n

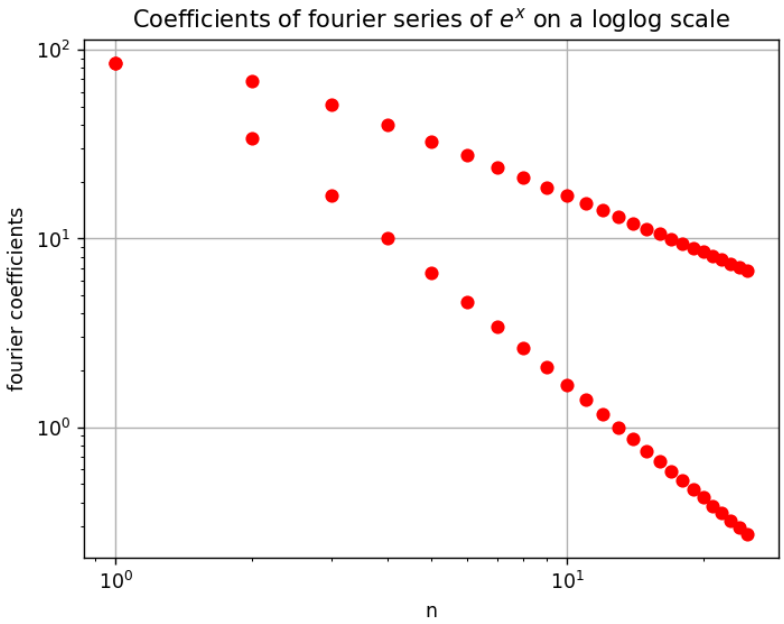

Figure 4: Coefficients of e^x on loglog

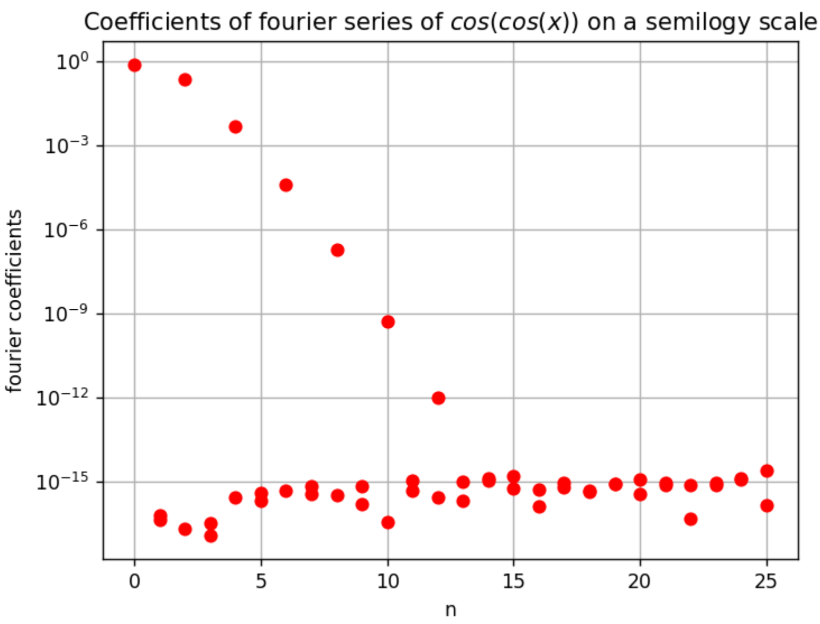

Figure 5: Coefficients of $cos(cos(x))$on semilog

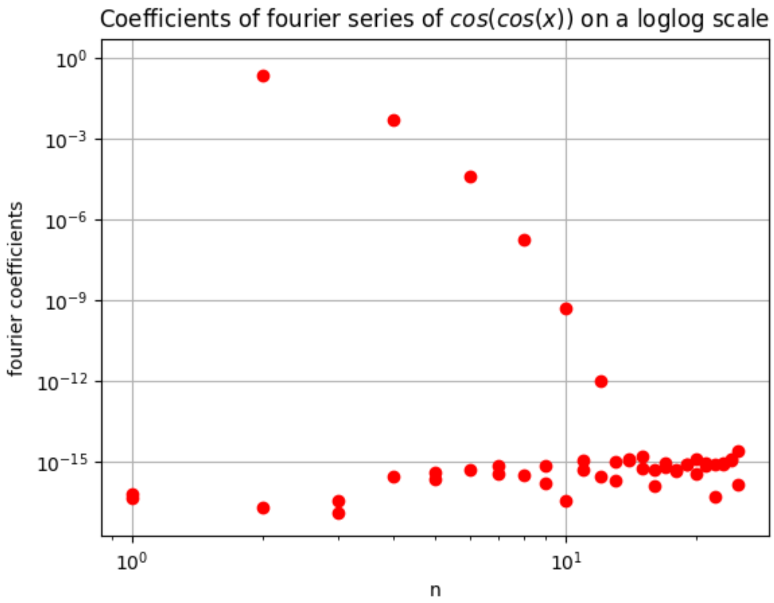

Figure 6: Coefficients of $cos(cos(x))$ on loglog

2. As can be seen, the values of b_n is nearly zero for $cos(cos(x))$ as it should be as the function is even(even function have zero b_n). The reason they are not exactly zero is probably because the integration is not precise enough.

3. The coefficients in first case dont decay as rapidly as second case because in case of cos(cos(x)) as the frequency of consideration is pi

4. The loglog graph looks linear in Figure 4 because of the fact the the fourier coefficients of e^x depend linearly on n.

Evaluating fourier coefficients via Least Squares approach

The following method via least square is used to calculate the fourier coefficients(NOTE - an and bn are calculated and stored separately).

```
x=linspace(0,2*pi,401)
x=x[:-1] # drop last term to have a proper periodic integral
b=f(x) # f has been written to take a vector
```

```
A=zeros((400,51)) # allocate space for 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
#endfor
c1=lstsq(A,b)[0] # the '[0]' is to pull out the
# best fit vector. lstsq returns a list.
```

Best fit via fourier tranform from Least Squares approach

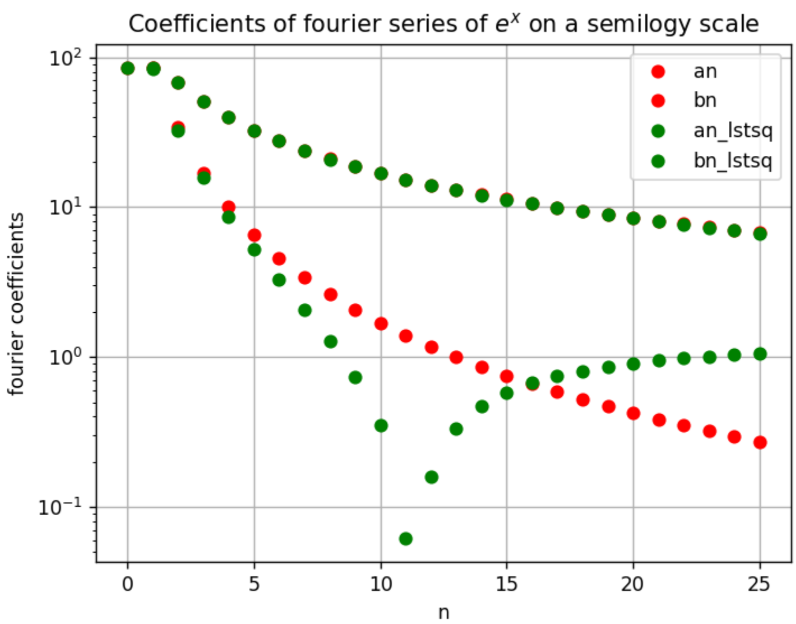

Figure 7: Coefficients of e^x (loglog)

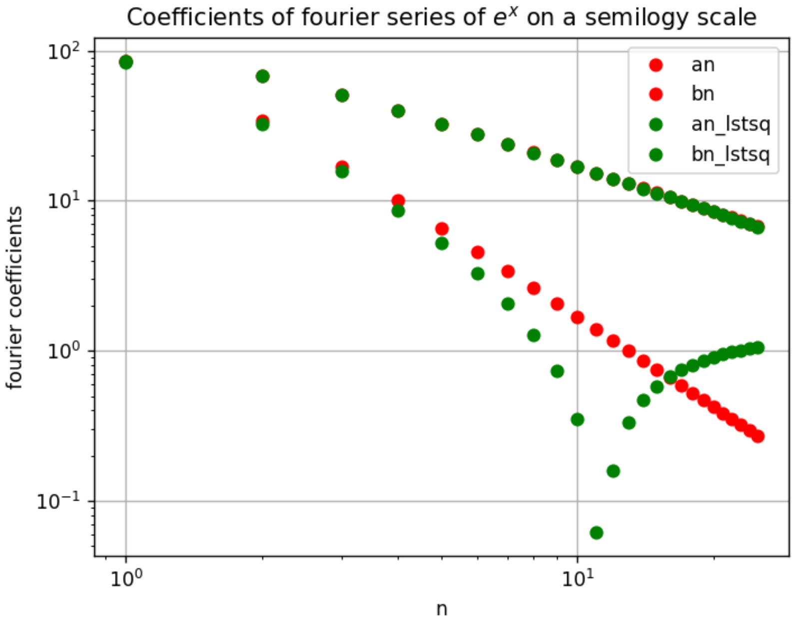

Figure 8: Coefficients of e^x (semilog)

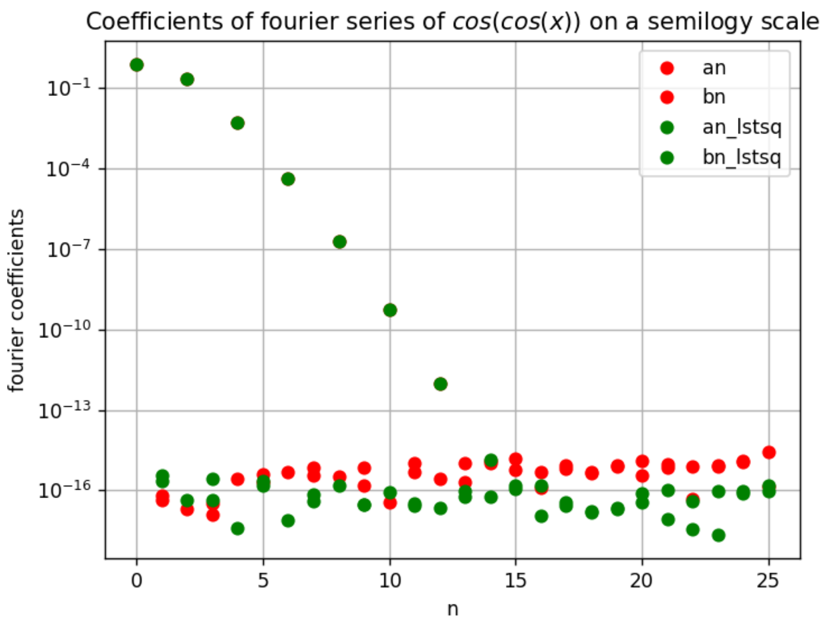

Figure 9: Coefficients of $cos(cos(x))$

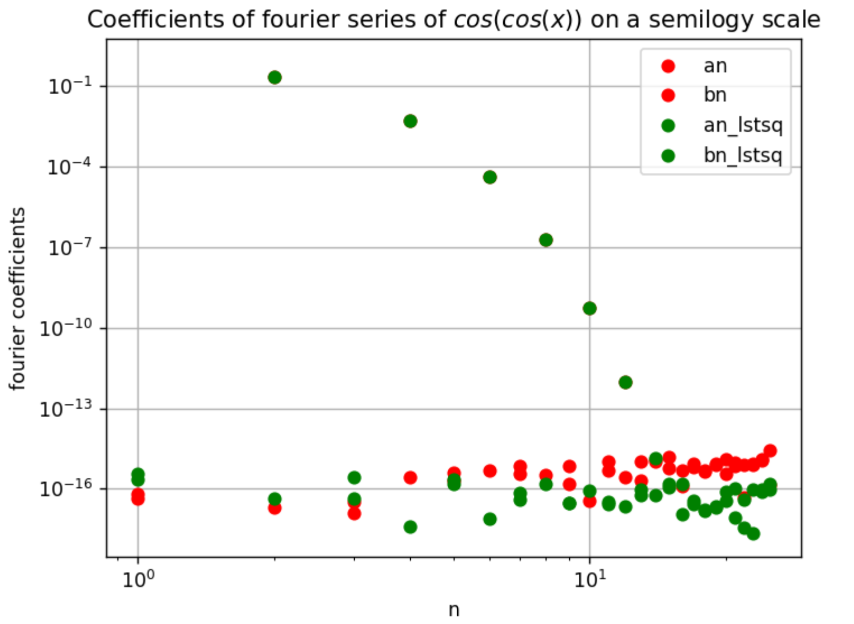

Figure 10: Coefficients of $cos(cos(x))$

Deviation between coefficients obtained via least squares and by the direct integration.

1. The following code has been used tofind it:

```
#for expx
A_diff = np.abs(A_temp - np.array(An_exp))
print("The max deviation is found for a", np.where( A_diff == np.amax(A_dif

B_diff = np.abs(B_temp - np.array(Bn_exp))
print("The max deviation is found for b", np.where( B_diff == np.amax(B_dif


#for coscosx
A_diff = np.abs(A_temp - np.array(An_coscos))
print("\nThe max deviation is found for a", np.where( A_diff == np.amax(A_di

B_diff = np.abs(B_temp - np.array(Bn_coscos))
print("The max deviation is found for b", np.where( B_diff == np.amax(B_dif
```

```
b_new_coscos = np.dot(A,c1)
```

---

2. The maximum deviation are

   (a) In e^x's a_n coefficients : The max deviation is found for a [1] = 1.3327308703353395

   (b) In e^x's b_n coefficients : The max deviation is found for b [25] = 0.08768050912536829

   (c) In $cos(cos(x))$'s a_n coefficients : The max deviation is found for a [15] = 1.7175489006562513e-15

   (d) In $cos(cos(x))$'s a_n coefficients : The max deviation is found for b [25] = 2.7586698812539523e-15

Plotting the graph of fourier tranform obtained via least square method :

1. The required plot is obtained as follows:

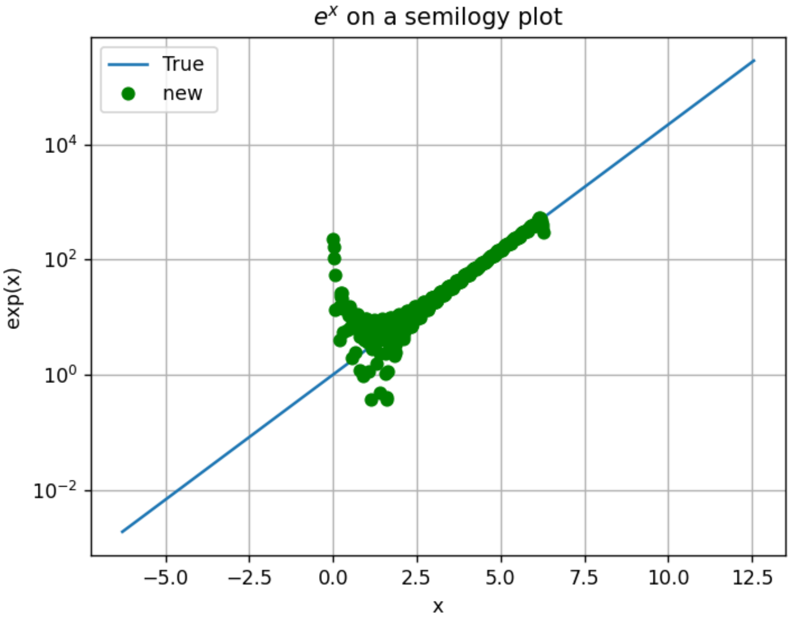

Figure 11: Fourier transform along with true graph ofe^x

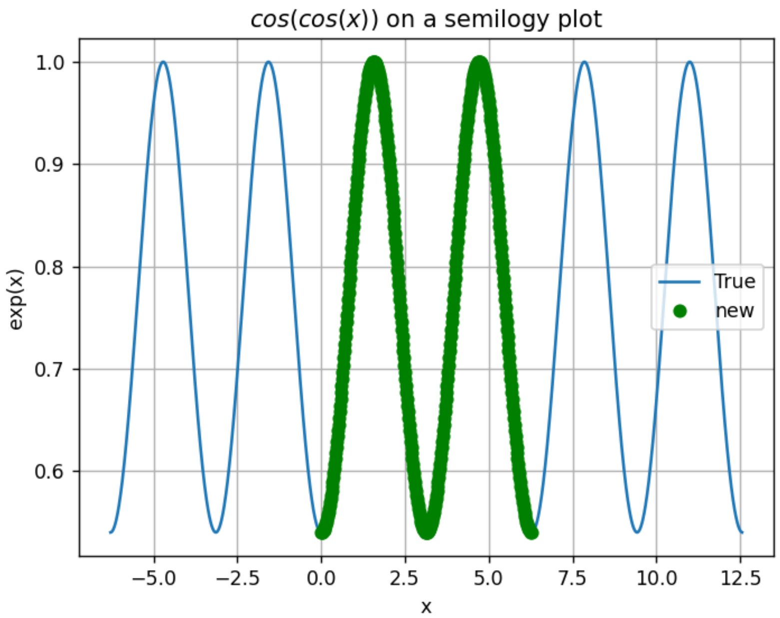

Figure 12: Fourier transform along with true graph of$cos(cos(x))$

2. As seen the transform of e^x has a lot more deviation when compared to $cos(cos(x))$. One reason for this could be the fact that since e^x is aperiodic so need very high number of coefficients too get good approximation. Whereas , $cos(cos(x))$ is periodic with period of π making its dependence on higher harmonics very low.

Conclusions

From this assignment we learnt the various methods as well deviations in using fourier transform. While $cos(cos(x))$ has a fourier transform giving very good approximation while the aperiodic e^x has higher deviations and more dependence on higher harmonics.