EE2703 : Applied Programming Lab
Assignment 4
Fourier Approximations

Keerthana Rachuri
EE20B102

March 11, 2022

1 Introduction:

A Fourier series is an expansion of a periodic function $f(x)$ in terms of an infinite sum of sines and cosines.Fourier series make use of the orthogonality relationships of the sine and cosine functions. The computation and study of Fourier series is known as harmonic analysis and is extremely useful as a way to break up an arbitrary periodic function into a set of simple terms that can be plugged in, solved individually, and then recombined to obtain the solution to the original problem or an approximation to it to whatever accuracy is desired or practical.

2 Actual Functions:

2.1 Exponential Function $\Rightarrow$ $f(x) = e^x$:

The function $f(x) = e^x$ is generated using:

```
1
2 # Function exp(x)
3 def exp(x):
4     return np.exp(x)
```

2.1.1 True Value verses Fourier series Expansion values of e^x:

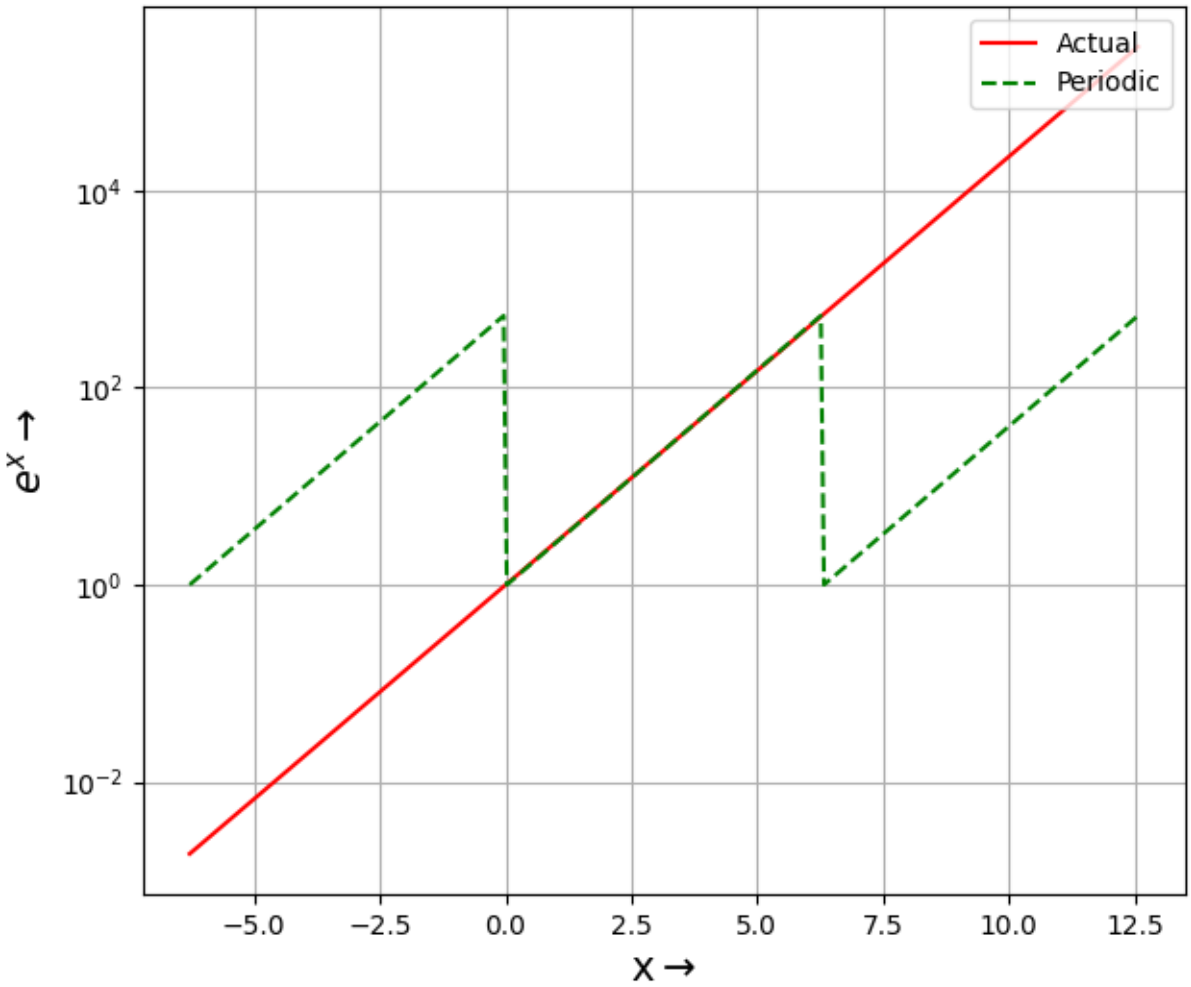

Figure 1.1: e^x in semi-log scale along with expected Fourier e^x

2.2 cos(cos(x)) Function $\Rightarrow$ $g(x) = cos(cos(x))$:

2.2.1 Linear scale Plotting:

The function $g(x) = cos(cos(x))$ is generated using:

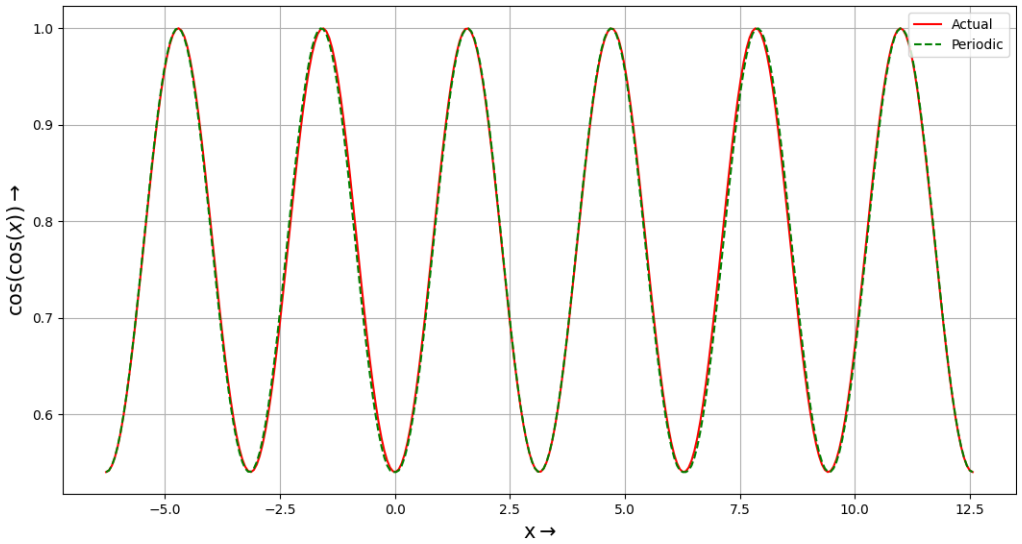

Figure 1.2: $cos(cos(x))$ in linear scale

The function $f(x) = e^x$ not periodic where as $g(x) = cos(cos(x))$ is periodic with period π.

The plot in Figure-3 is generated by:

```
#cos(cos(x)) in linear scale
pylab.figure(figsize=(7,6))
# Creating A New Figure
pylab.plot(xActual,coscos(xActual),'-r',label='Actual')                    #
  Ploting y vs x As Lines And Markers
pylab.plot(xActual,coscos(xExpected),'--g',label='Periodic')               #
  Ploting y vs x As Lines And Markers
pylab.xlabel(r'x$\rightarrow$',fontsize=15)                                #
  Setting The Label For The x-axis
pylab.ylabel(r'$\cos(\cos(x))\rightarrow$',fontsize=15)                    #
  Setting The Label For The y-axis
pylab.legend(loc='upper right')                                            #
  Placing A Legend On The Top Right Corner Of The Graph
pylab.grid(True)                                                           #
  Displaying The Grid
pylab.show()
```

3 Fourier Coefficients:

The Fourier coefficients are calculated using the given formulae:

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

Now we separately generate the first 51 Fourier coefficients of the $f(x)$ and $g(x)$.

3.1 Exponential Function $\Rightarrow$ $f(x) = e^x$:

3.1.1 Code for Finding $f(x)$:

The integration function is calculated for $f(x) = e^x$:

```
# Function to Find Fourier Coefficient
def findingCoeff(n,func):
    coeff=np.zeros(n)                                          # Constructing A Array Filled With Zeros
    fourier=function[func]
    a=lambda x,k:fourier(x)*np.cos(k*x)                        # Cosine Function
    b=lambda x,k:fourier(x)*np.sin(k*x)                        # Sine Function
    coeff[0]=quad(fourier,0,2*np.pi)[0]/(2*np.pi)              # DC Coefficient
    for i in range(1,n,2):
        coeff[i]=quad(a,0,2*np.pi,args=((i+1)/2))[0]/np.pi     # Sine Coefficient
    for i in range(2,n,2):
        coeff[i]=quad(b,0,2*np.pi,args=(i/2))[0]/np.pi         # Cosine Coefficient
    return coeff
    coefficientExp=findingCoeff(51,'exp(x)')                   # Fourier Coefficients of exp(x)

```

3.1.2 Plot of $f(x)$:

3.1.2.1 Semi-log-y plot of $f(x)$:

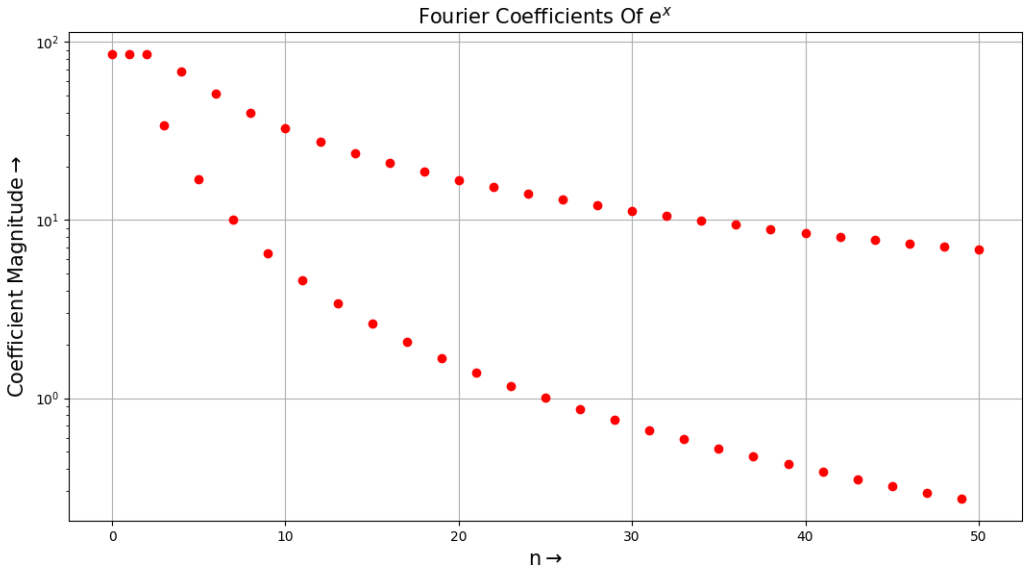

Figure 1.3: Fourier Coefficients of $f(x)$ by direct integration in semilog scale

The plot in Figure-1.4 is generated by:

```
1
2 #Fourier Coefficients of f(x) by direct integration in semilog scale
3 pylab.figure(figsize=(7,6))                                          #
    Creating A New Figure
4 pylab.semilogy(range(51),np.abs(coefficientExp),'ro')                # Making
     A Plot With Log Scaling On The y-axis
5 pylab.xlabel(r'n$\rightarrow$',fontsize=15)                          #
    Setting The Label For The x-axis
6 pylab.ylabel(r'Coefficient Magnitude$\rightarrow$',fontsize=15)      #
    Setting The Label For The y-axis
7 pylab.title('Fourier Coefficients Of $e^{x}$',fontsize=15)           #
    Setting Title Of The Graph
8 pylab.grid(True)                                                     #
    Displaying The Grid
9 pylab.show()                                                         #
    Displaying The Figure
10
```

3.1.2.2 loglog plot of $f(x)$:

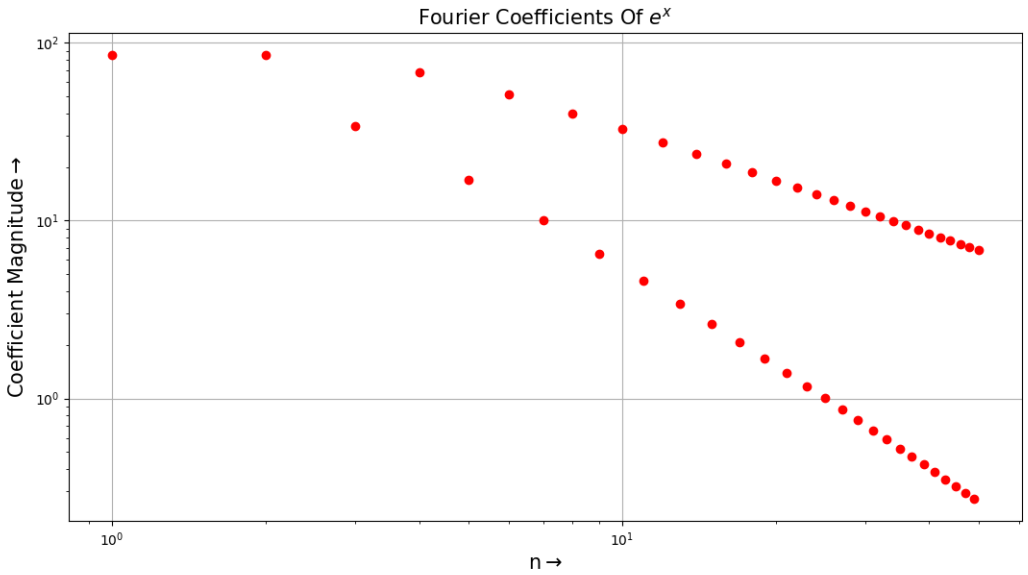

Figure 1.4: Fourier Coefficients of $f(x)$ by direct integration in loglog scale

The plot in Figure-1.4 is generated by:

```
#Fourier Coefficients of f(x) by direct integration in loglog scale
pylab.figure(figsize=(7,6))                                          #
    Creating A New Figure
pylab.loglog(range(51),np.abs(coefficientExp),'ro')                  # Making
     A Plot With Log Scaling On Both The Axis
pylab.xlabel(r'n$\rightarrow$',fontsize=15)                          #
    Setting The Label For The x-axis
pylab.ylabel(r'Coefficient Magnitude$\rightarrow$',fontsize=15)      #
    Setting The Label For The y-axis
pylab.title('Fourier Coefficients Of $e^{x}$',fontsize=15)           #
    Setting Title Of The Graph
pylab.grid(True)                                                     #
    Displaying The Grid
pylab.show()

```

3.2 cos(cos(x)) Function $\Rightarrow$ $g(x) = cos(cos(x))$:

3.2.1 Code for Finding $cos(cos(x))$:

The integration function is calculated for $g(x) = cos(cos(x))$:

```
function={'exp(x)':exp,'cos(cos(x))':coscos}
# Function to Find Fourier Coefficient
```

```
def findingCoeff(n,func):
    coeff=np.zeros(n)                                          # Constructing
    A Array Filled With Zeros
    fourier=function[func]
    a=lambda x,k:fourier(x)*np.cos(k*x)                              # Cosine
    Function
    b=lambda x,k:fourier(x)*np.sin(k*x)                              # Sine
   Function
    coeff[0]=quad(fourier,0,2*np.pi)[0]/(2*np.pi)              # DC
   Coefficient
    for i in range(1,n,2):
        coeff[i]=quad(a,0,2*np.pi,args=((i+1)/2))[0]/np.pi     # Sine
   Coefficient
    for i in range(2,n,2):
        coeff[i]=quad(b,0,2*np.pi,args=(i/2))[0]/np.pi         # Cosine
   Coefficient
    return coeff
```

3.2.2 Plot of $g(x)$:

3.2.2.1 Semi-log-y plot of $g(x)$:

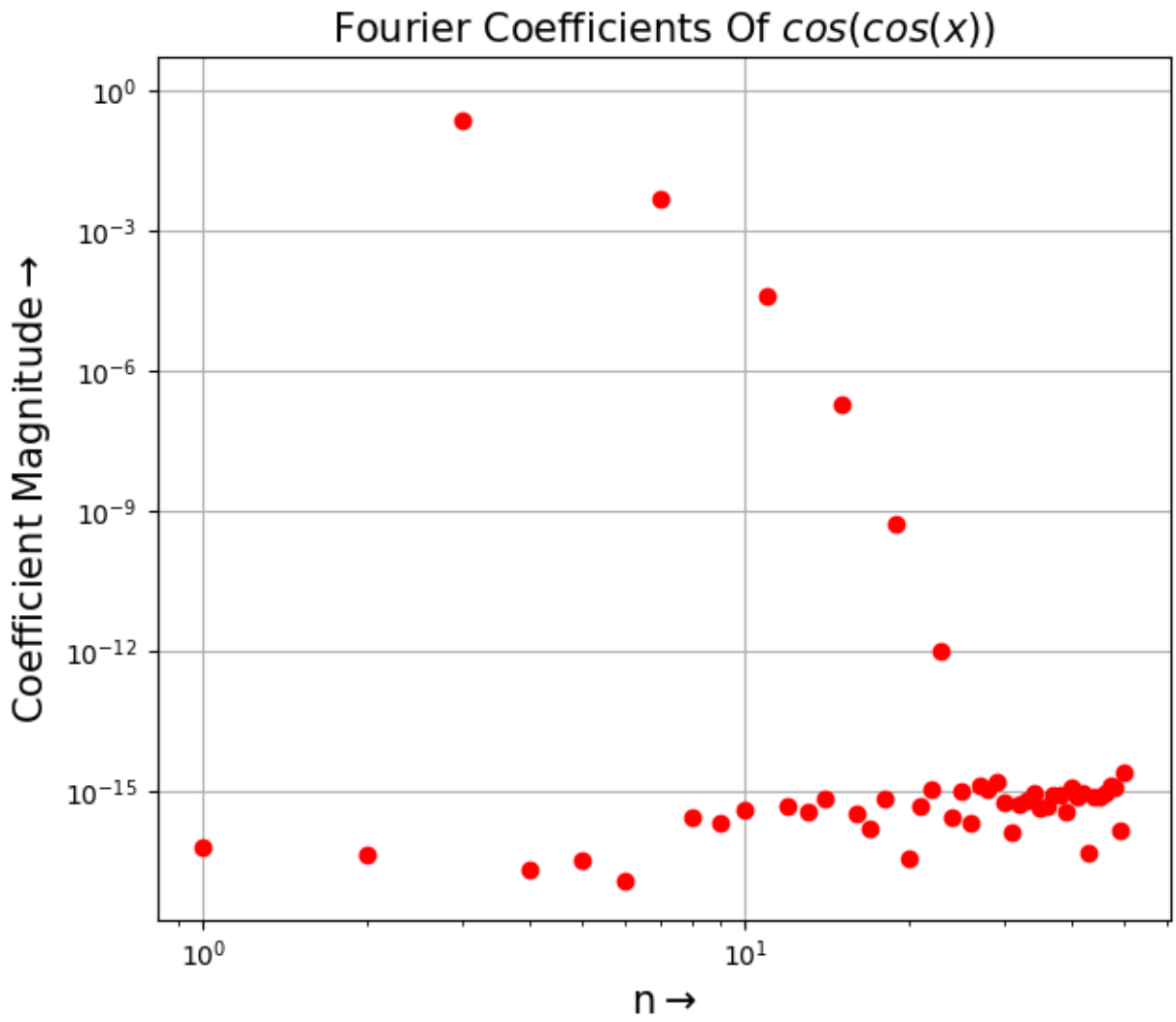

Figure 1.5: Fourier Coefficients of $g(x)$ by direct integration in semilog scale

The plot in Figure-1.5 is generated by:

```
#Fourier Coefficients of g(x) by direct integration in semilog scale
pylab.figure(figsize=(7,6))                                    # Creating A New Figure
pylab.semilogy(range(51),np.abs(coefficientCosCos),'ro')       # Making A Plot With Log Scaling On Both The Axis
pylab.xlabel(r'n$\rightarrow$',fontsize=15)                    # Setting The Label For The x-axis
pylab.ylabel(r'Coefficient Magnitude$\rightarrow$',fontsize=15)  # Setting The Label For The y-axis
```

```
pylab.title('Fourier Coefficients Of $cos(cos(x))$',fontsize=15)     #
    Setting Title Of The Graph
pylab.grid(True)                                                    #
    Displaying The Grid
pylab.show()                                                        #
    Displaying The Figure
```

3.2.2.2 loglog plot of $g(x)$:

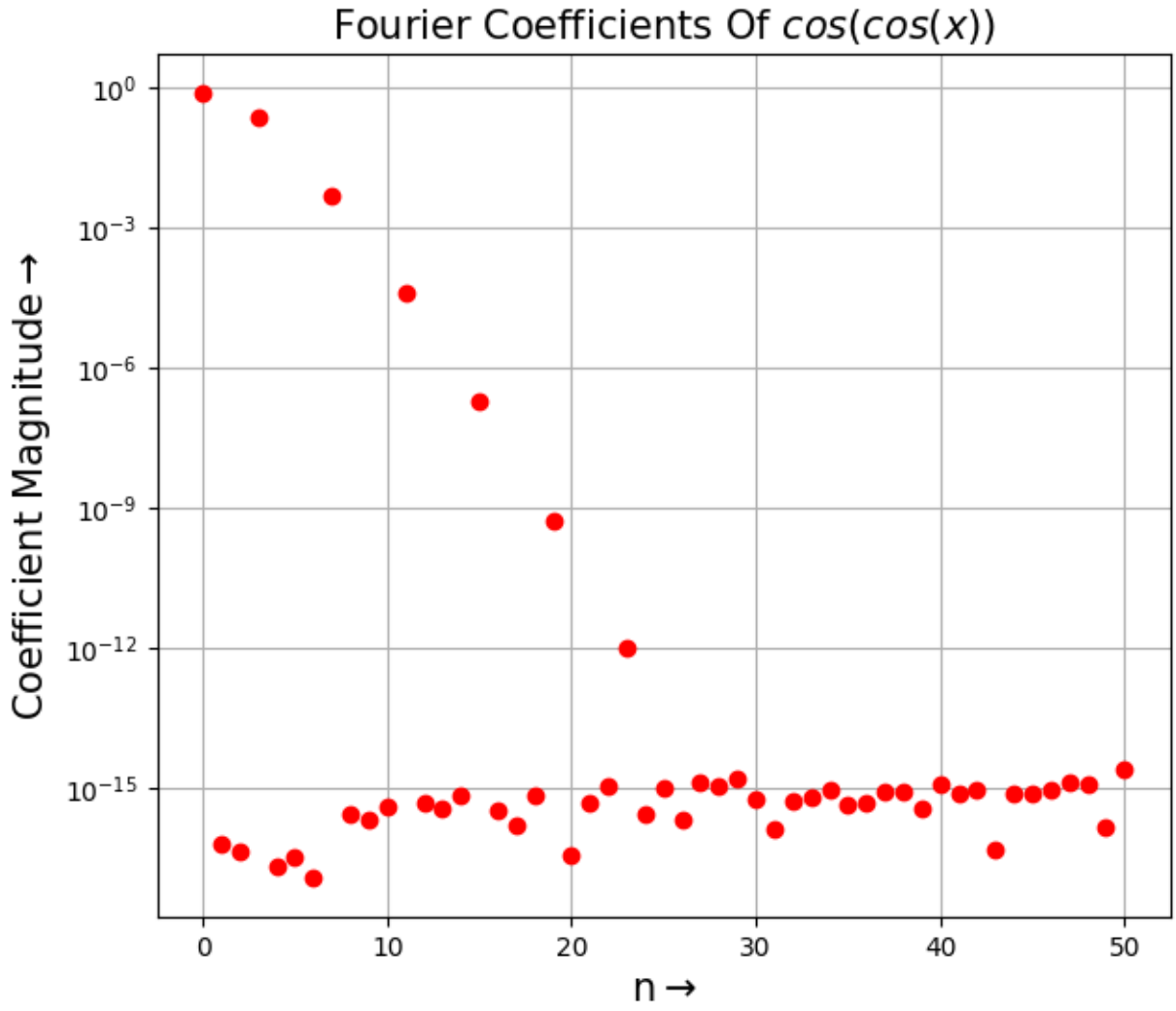

Figure 1.6: Fourier Coefficients of $g(x)$ by direct integration in loglog scale

The plot in Figure-1.6 is generated by:

```
#Fourier Coefficients of g(x) by direct integration in loglog scale
pylab.figure(figsize=(7,6))                                         #
    Creating A New Figure
```

```
3 pylab.loglog(range(51),np.abs(coefficientCosCos),'ro')                # Making
      A Plot With Log Scaling On Both The Axis
4 pylab.xlabel(r'n$\rightarrow$',fontsize=15)                           #
     Setting The Label For The x-axis
5 pylab.ylabel(r'Coefficient Magnitude$\rightarrow$',fontsize=15)       #
     Setting The Label For The y-axis
6 pylab.title('Fourier Coefficients Of $cos(cos(x))$',fontsize=15)      #
     Setting Title Of The Graph
7 pylab.grid(True)                                                      #
     Displaying The Grid
8 pylab.show()                                                          #
     Displaying The Figure
```

4 Least Square Approach:

4.1 Generating Fourier Coefficients:

The original functions are tried to be reconstructed using the Fourier expansion upto first 51 terms. The Fourier coefficients are calculated using `np.linalg.lstsq` function.

```
x=np.linspace(0,2*np.pi,400)
A=np.zeros((400,51))
A[:,0]=1
for k in range(1,26):
    A[:,2*k-1]=np.cos(k*x)
    A[:,2*k]=np.sin(k*x)
bExp=exp(x)
bCosCos=coscos(x)
cExp=np.linalg.lstsq(A,bExp,rcond=-1)[0]
cCosCos=np.linalg.lstsq(A,bCosCos,rcond=-1)[0]
```

4.2 Plots of Fourier Coefficients:

4.2.1 Plots of $f(x)$:

4.2.1.1 Semilog Plot of $f(x)$:

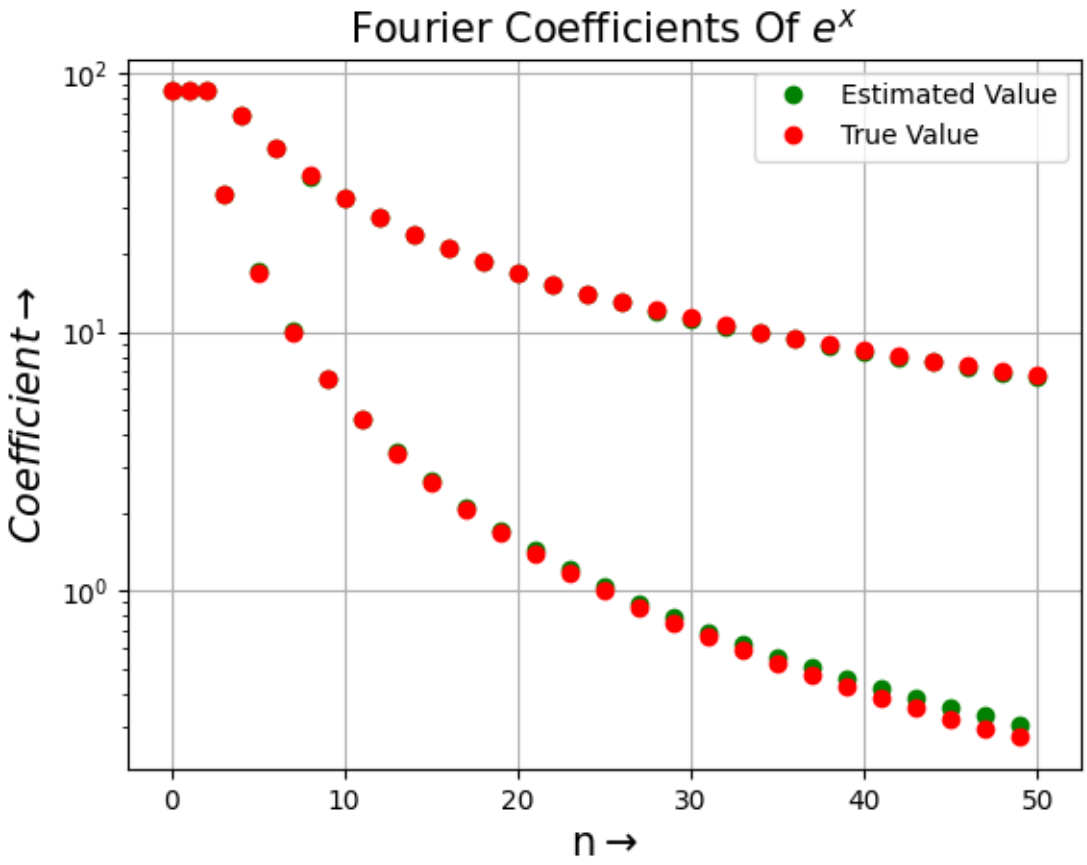

Figure 1.7: Fourier Coefficients of $f(x)$ in semilog scale

The plot in Figure-1.7 is generated by:

```
#semilog plot of e^x
pylab.semilogy(range(51),np.abs(cExp),'go',label='Estimated Value')
    # Making A Plot With Log Scaling On The y-axis
pylab.semilogy(range(51),np.abs(coefficientExp),'ro',label='True Value')
    # Making A Plot With Log Scaling On The y-axis
pylab.xlabel(r'n$\rightarrow$',fontsize=15)
    # Setting The Label For The x-axis
pylab.ylabel(r'$Coefficient\rightarrow$',fontsize=15)
    # Setting The Label For The y-axis
pylab.legend(loc='upper right')
    # Placing A Legend On The Top Right Corner Of The Graph
pylab.title('Fourier Coefficients Of $e^{x}$',fontsize=15)
    # Setting Title Of The Graph
```

```
pylab.grid(True)
    # Displaying The Grid
pylab.show()
    # Displaying The Figure
```

4.2.1.2 loglog Plot of $f(x)$:

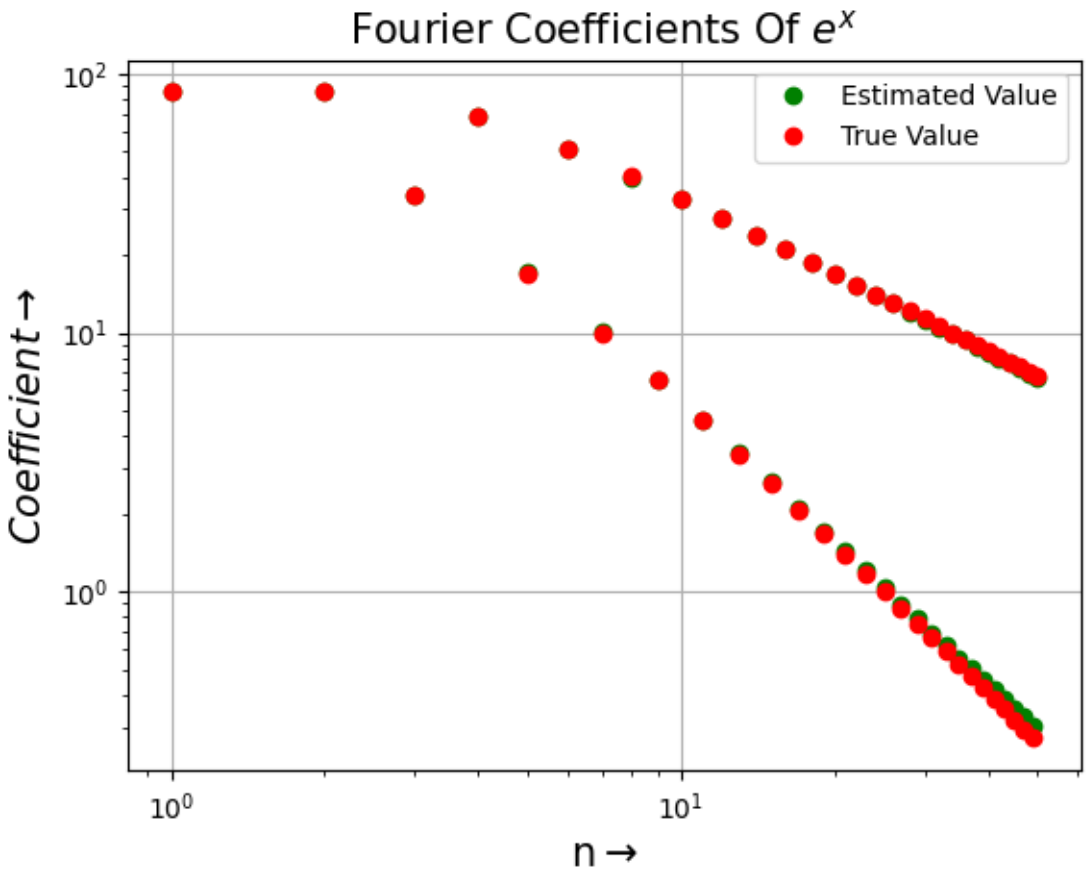

Figure 1.8: Fourier Coefficients of $f(x)$ in loglog scale

The plot in Figure- 1.8 is generated by:

```
#loglog plot of e^x
pylab.loglog(range(51),np.abs(cExp),'go',label='Estimated Value')
    # Making A Plot With Log Scaling On Both The Axis
pylab.loglog(range(51),np.abs(coefficientExp),'ro',label='True Value')
    # Making A Plot With Log Scaling On Both The Axis
pylab.xlabel(r'n$\rightarrow$',fontsize=15)
    # Setting The Label For The x-axis
pylab.ylabel(r'$Coefficient\rightarrow$',fontsize=15)
    # Setting The Label For The y-axis
pylab.legend(loc='upper right')
    # Placing A Legend On The Top Right Corner Of The Graph
pylab.title('Fourier Coefficients Of $e^{x}$',fontsize=15)
    # Setting Title Of The Graph
```

```
pylab.grid(True)
    # Displaying The Grid
pylab.show()
    # Displaying The Figure
```

4.2.2 Plots of $g(x)$:

4.2.2.1 Semilog Plot of $g(x)$:

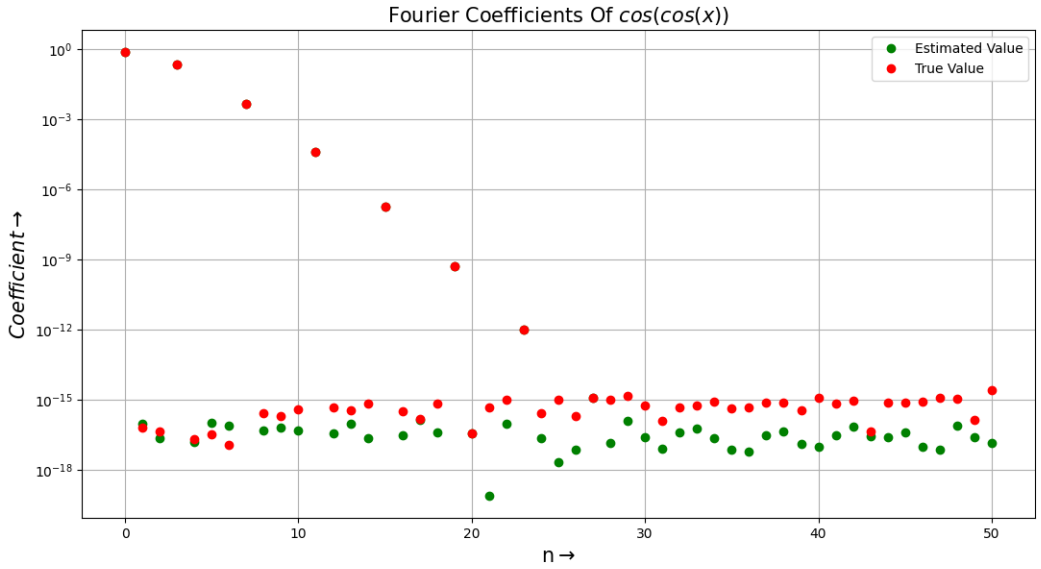

Figure 1.9: Fourier Coefficients of $g(x)$ in semilog scale

The plot in Figure-1.9 is generated by:

```
#semilog plot of cos(cos(x))
pylab.semilogy(range(51),np.abs(cCosCos),'go',label='Estimated Value')
    # Making A Plot With Log Scaling On The y-axis
pylab.semilogy(range(51),np.abs(coefficientCosCos),'ro',label='True Value')
    # Making A Plot With Log Scaling On The y-axis
pylab.xlabel(r'n$\rightarrow$',fontsize=15)
    # Setting The Label For The x-axis
pylab.ylabel(r'$Coefficient\rightarrow$',fontsize=15)
    # Setting The Label For The y-axis
pylab.legend(loc='upper right')
    # Placing A Legend On The Top Right Corner Of The Graph
pylab.title('Fourier Coefficients Of $cos(cos(x))$',fontsize=15)
    # Setting Title Of The Graph
pylab.grid(True)
    # Displaying The Grid
pylab.show()
    # Displaying The Figure
```

4.2.2.2 loglog Plot of $g(x)$:

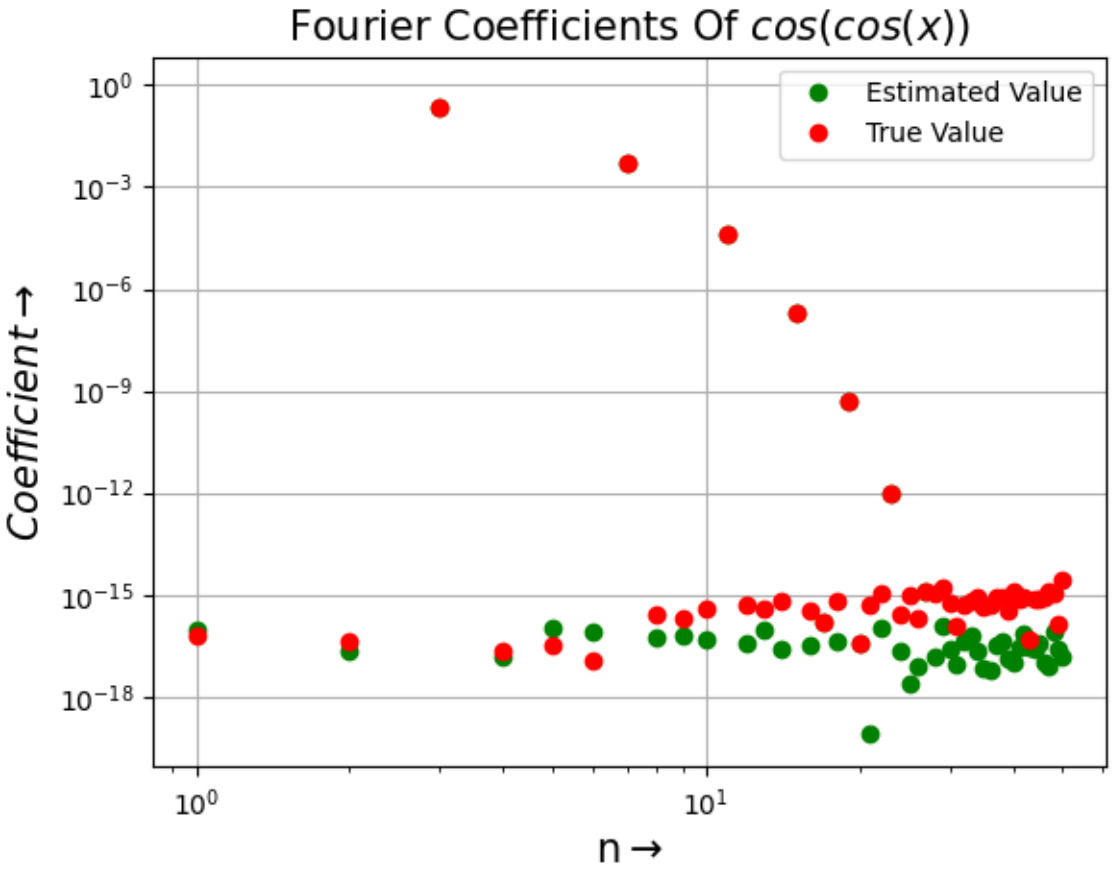

Figure 1.10: Fourier Coefficients of $g(x)$ in loglog scale

The plot in Figure-1.10 is generated by:

```
#loglog Plot of cos(cos(x))
pylab.loglog(range(51),np.abs(cCosCos),'go',label='Estimated Value')
    # Making A Plot With Log Scaling On Both The Axis
pylab.loglog(range(51),np.abs(coefficientCosCos),'ro',label='True Value')
    # Making A Plot With Log Scaling On Both The Axis
pylab.xlabel(r'n$\rightarrow$',fontsize=15)
    # Setting The Label For The x-axis
pylab.ylabel(r'$Coefficient\rightarrow$',fontsize=15)
    # Setting The Label For The y-axis
pylab.legend(loc='upper right')
    # Placing A Legend On The Top Right Corner Of The Graph
pylab.title('Fourier Coefficients Of $cos(cos(x))$',fontsize=15)
    # Setting Title Of The Graph
pylab.grid(True)
    # Displaying The Grid
pylab.show()
    # Displaying The Figure
```

4.3 Plots of Reconstructed Functions:

4.3.1 Plots of $f(x)$:

4.3.1.1 Semilog plot of $f(x)$:

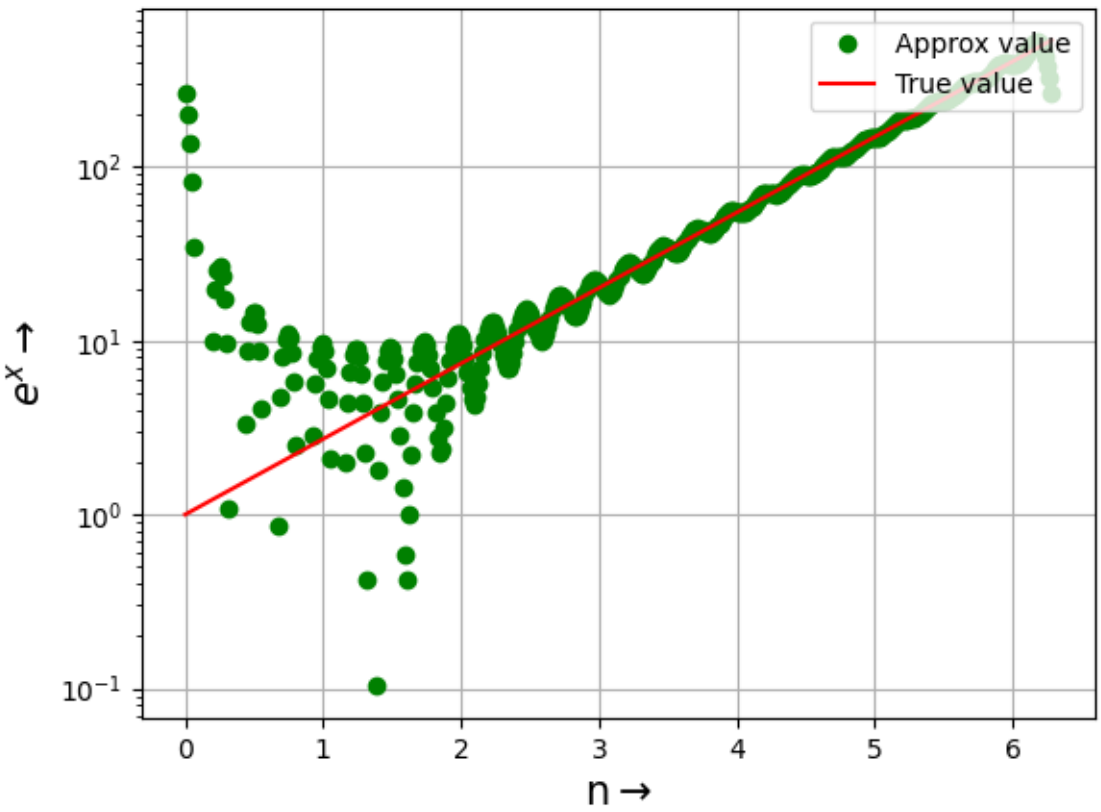

Figure 1.11: $f(x) = e^x$ obtained from reconstructing the Fourier coefficients verses Real $f(x)$

The plot in Figure-1.11 is generated by:

```
#original and reconstructed function from fourier coefficients for e^x
approxExp=np.matmul(A,cExp)                                    # Matrix
    Product Of Two Arrays
pylab.semilogy(x,approxExp,'go',label="Approx value")          # Making A
    Plot With Log Scaling On The y-axis
pylab.semilogy(x,exp(x),'-r',label='True value')               # Making A
    Plot With Log Scaling On The y-axis
pylab.xlabel(r'n$\rightarrow$',fontsize=15)                    # Setting
    The Label For The x-axis
pylab.ylabel(r'$e^{x}\rightarrow$',fontsize=15)                # Setting
    The Label For The y-axis
pylab.legend(loc='upper right')                                # Placing A
    Legend On The Top Right Corner Of The Graph
```

```
pylab.grid(True)                                                    # Displaying
    The Grid
pylab.show()                                                        # Displaying
    The Figure
```

4.3.2 Plots of $g(x)$:

4.3.2.1 Loglog plot of $g(x)$:

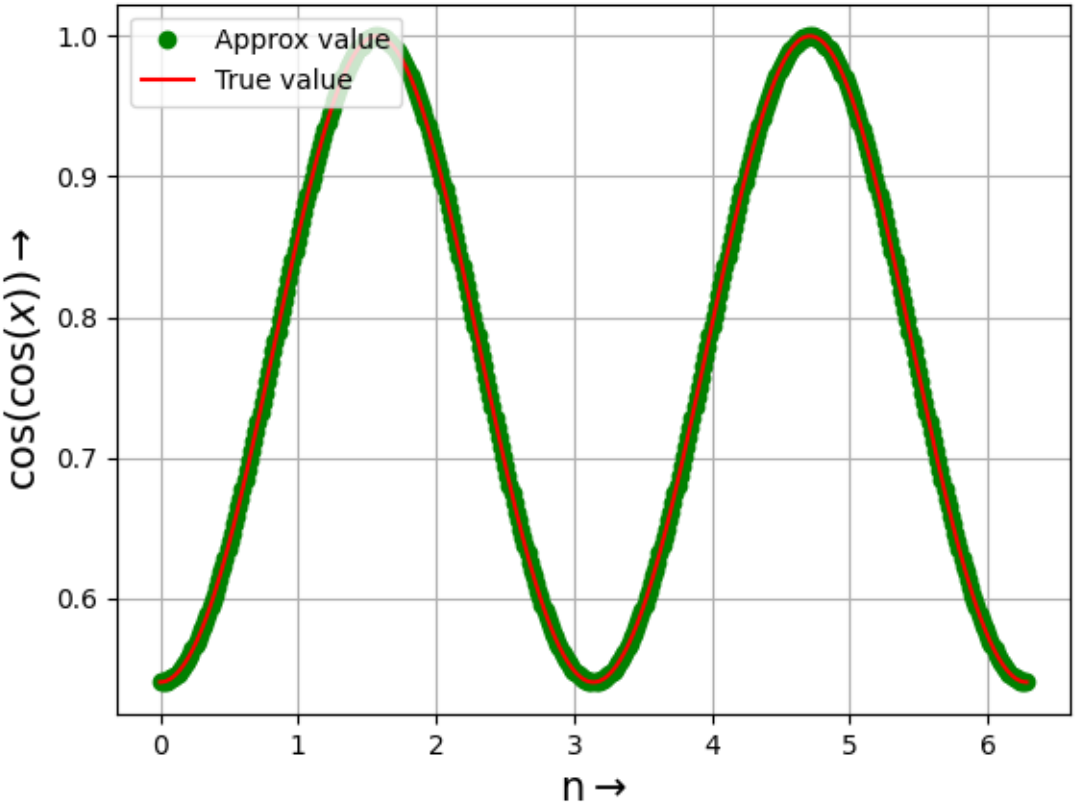

Figure 1.12: $g(x) = cos(cos(x))$ obtained from reconstructing the Fourier coefficients verses Real $g(x)$

The plot in Figure-1.12 is generated by:

```
#original and reconstructed function from fourier coefficients for cos(cos(x
    ))
approxCosCos=np.matmul(A,cCosCos)                                   # Matrix
    Product Of Two Arrays
pylab.plot(x,approxCosCos,'go',label="Approx value")                # Ploting y
    vs x As Lines And Markers
pylab.plot(x,coscos(x),'-r',label='True value')                     # Ploting y
    vs x As Lines And Markers
```

```
5 pylab.xlabel(r'n$\rightarrow$',fontsize=15)                    # Setting
    The Label For The x-axis
6 pylab.ylabel(r'$\cos(\cos(x))\rightarrow$',fontsize=15)        # Setting
    The Label For The y-axis
7 pylab.legend(loc='upper left')                                 # Placing A
    Legend On The Top Right Corner Of The Graph
8 pylab.grid(True)                                               # Displaying
     The Grid
9 pylab.show()                                                   # Displaying
     The Figure
```

5 Error Analysis:

The error is calculated as follows:

```
deviationExp=abs(coefficientExp-cExp)
deviationCosCos=abs(coefficientCosCos-cCosCos)
maxDeviationExp=np.max(deviationExp)
maxDeviationCosCos=np.max(deviationCosCos)
print(maxDeviationExp)
print(maxDeviationCosCos)
```

5.1 Absolute Error:

The absolute error in $f(x) = e^x$ in between the coefficients estimated through LSTSQ method and Direct Integration Method is 35.12283611763861

The absolute error in $g(x) = cos(cos(x))$ in between the coefficients estimated through LSTSQ method and Direct Integration Method is 2.8114348359621343e-14

5.2 Mean Error:

The Mean Error in $f(x) = e^x$ in between the coefficients estimated through LSTSQ method and Direct Integration Method is 0.6886830611301689

The Mean Error in $g(x) = cos(cos(x))$ in between the coefficients estimated through LSTSQ method and Direct Integration Method is 5.51261732541595e-16

5.3 Maximum Deviation:

The Maximum deviation in $f(x) = e^x$ in between the coefficients estimated through LSTSQ method and Direct Integration Method is 1.3327308703354106 , which occurs at a_{25}

The Maximum deviation in $g(x) = cos(cos(x))$ in between the coefficients estimated through LSTSQ method and Direct Integration Method is 2.656286271711572e-15 , which occurs at a_{25}

6 Result:

- The function $cos(cos(x))$ is periodic while e^x is not periodic but for e^x we get a periodic extension using Fourier analysis.
- Coefficients b_n of $cos(cos(x))$ are expected to be 0 as the function is even and b_n are coefficients of odd sinusoidal components but the small values of b_n are due to error in integration.
- The coefficients of e^x decay slowly compared to $cos(cos(x))$ as the function is not periodic in the given interval of $[0,2\pi)$.

7 Conclusion:

We calculated the Fourier Expansion of two functions $f(x) = e^x$ and $g(x) = cos(cos(x))$ and calculated the deviation the reconstructed function from the actual function.