Assignment 4 - Jupyter Notebook

Yogesh Agarwala
EE19B130

March 10, 2021

0.1 Assignment 4: Fourier Approximations

[1]:
```
import numpy as np
import math
from scipy.integrate import quad
import matplotlib.pyplot as plt
import timeit
```

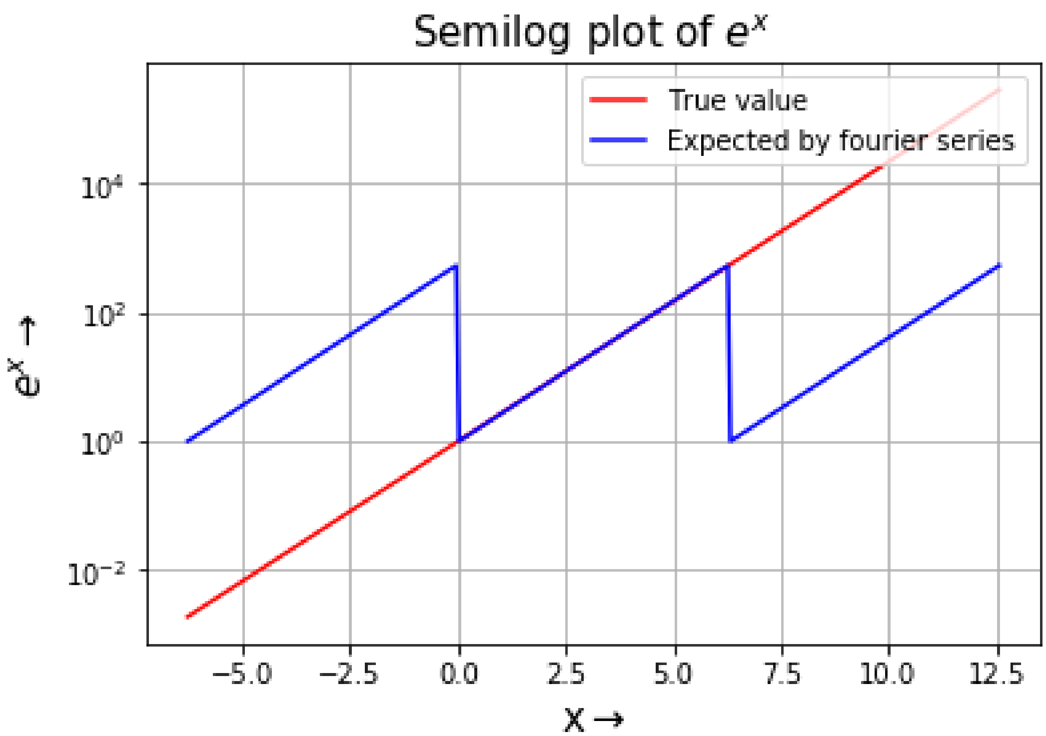
0.2 Q1. Defining Functions

[2]:
```
# The functions e^x and cos(cos(x)) are defined as:
def exp(x):
    return np.exp(x)
```

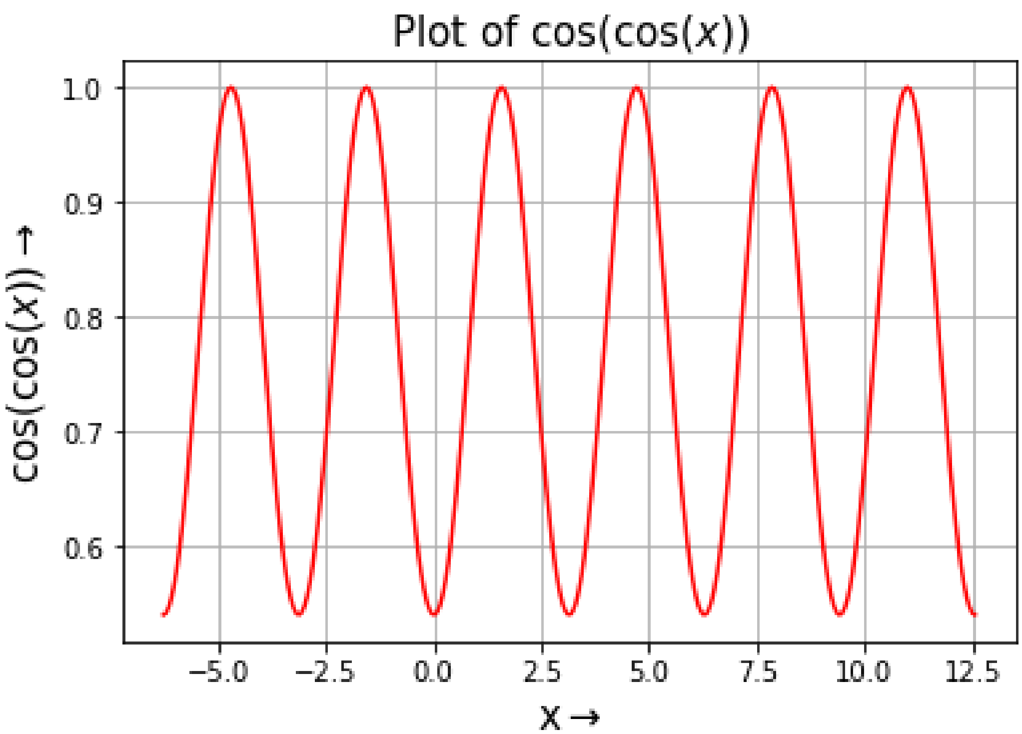
[3]:
```
def coscos(x):
    return np.cos(np.cos(x))
```

0.3 Function Plots

[4]:
```
# for e^x

x1 = np.linspace(-2*np.pi,4*np.pi,300)
x2 = np.linspace(0,2*np.pi,100)
tiled = np.tile(x2,3)
exp_x = exp(x1)

#Since exp(x) grows rapidly, we use semilogy for that plot.
plt.semilogy(x1,exp_x,'-r',label='True value')
plt.semilogy(x1,exp(tiled),'-b',label='Expected by fourier series')
plt.grid(True)
plt.ylabel(r'$e^{x}\rightarrow$',fontsize=15)
plt.xlabel(r'x$\rightarrow$',fontsize=15)
plt.title('Semilog plot of $e^{x}$',fontsize=15)
plt.legend(loc='upper right')
plt.show()
```



[5]:
```
# for cos(cos(x))

x1 = np.linspace(-2*np.pi,4*np.pi,300)
coscos_x = coscos(x1)

plt.plot(x1,coscos_x,'r')
plt.grid(True)
plt.xlabel(r'x$\rightarrow$',fontsize=15)
plt.ylabel(r'$\cos(\cos(x))\rightarrow$',fontsize=15)
plt.title('Plot of $\cos(\cos(x))$',fontsize=15)
plt.show()
```



0.4 Q2. Finding the Fourier series coefficients: Integration approach

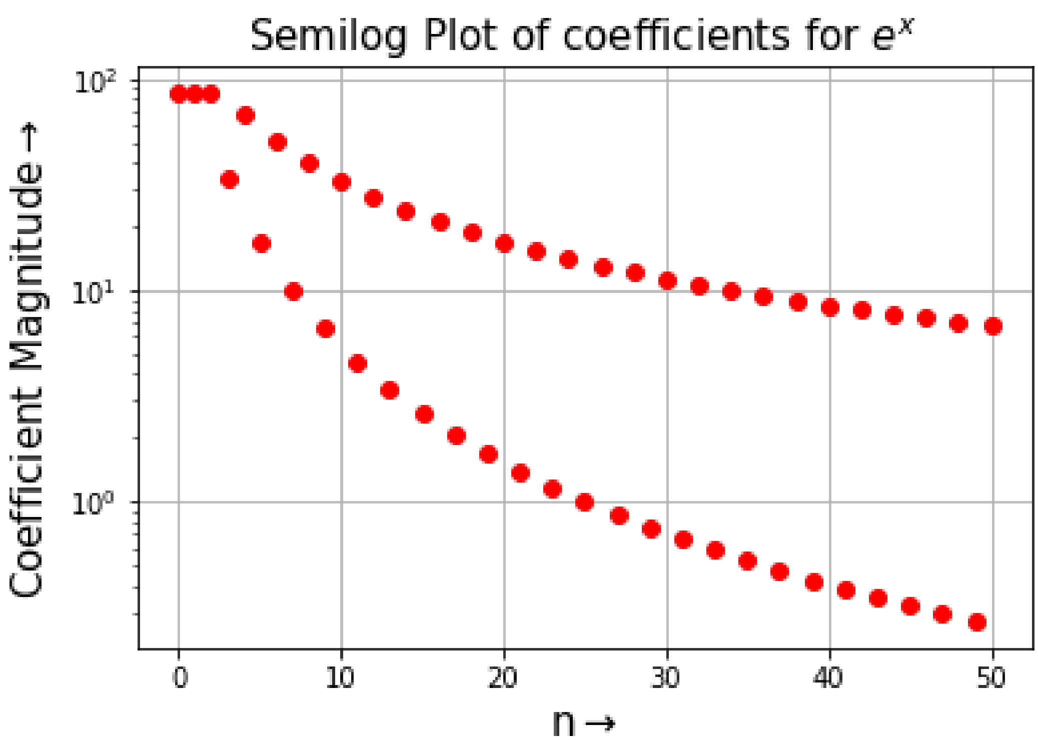
[6]:
```
#this function returns the first n fourier coefficients of the input function␣
↪using the integration method

func_dict = {'exp(x)':exp,'cos(cos(x))': coscos}
def find_coeff(n,label):
    coeff = np.zeros(n)
    func = func_dict[label]
    u = lambda x,k: func(x)*np.cos(k*x)
    v = lambda x,k: func(x)*np.sin(k*x)
    coeff[0]= quad(func,0,2*np.pi)[0]/(2*np.pi)
    for i in range(1,n,2):
        coeff[i] = quad(u,0,2*np.pi,args=((i+1)/2))[0]/np.pi
    for i in range(2,n,2):
        coeff[i] = quad(v,0,2*np.pi,args=(i/2))[0]/np.pi
    return coeff
```

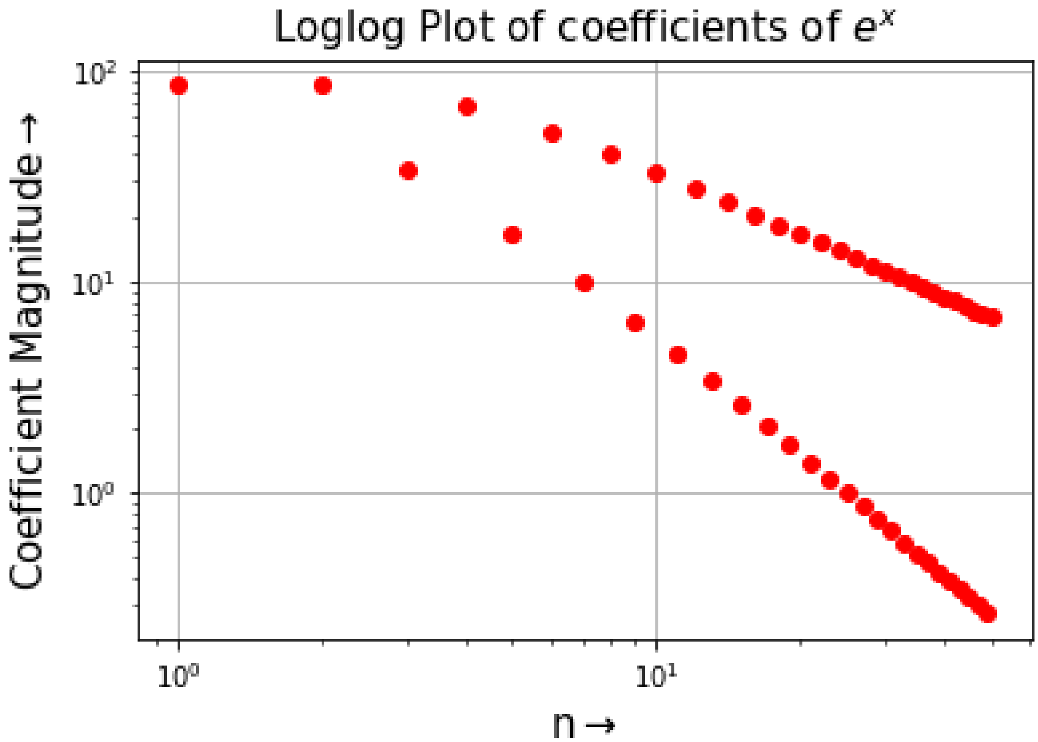
[7]:
```
#calculating first 51 fourier coefficients for the two functions

coeff_coscos = find_coeff(51,'cos(cos(x))')
coeff_exp = find_coeff(51,'exp(x)')
```

0.5 Q3. Semilog and Loglog plot for both functions

0.5.1 (a) For e^x

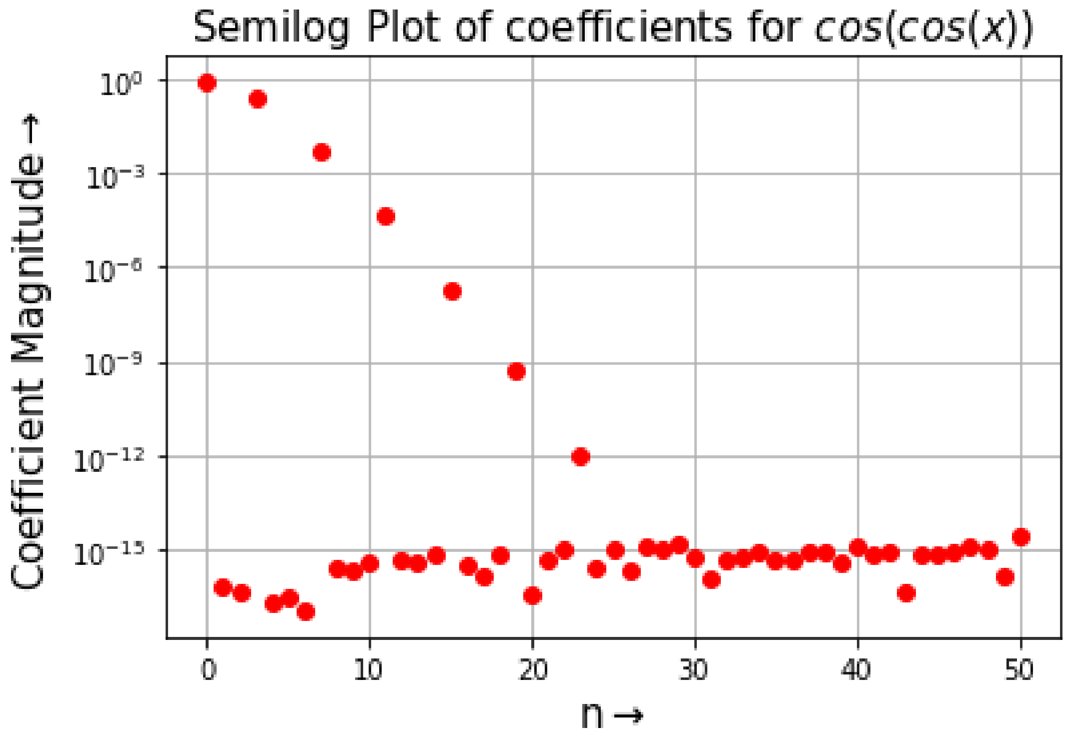
[8]:
```
# semilogy plot
plt.semilogy(range(51),np.abs(coeff_exp),'ro')
plt.grid(True)
plt.xlabel(r'n$\rightarrow$',fontsize=15)
plt.ylabel(r'Coefficient Magnitude$\rightarrow$',fontsize=15)
plt.title('Semilog Plot of coefficients for $e^{x}$',fontsize=15)
plt.show()

# loglog plot
plt.loglog(range(51),np.abs(coeff_exp),'ro')
plt.grid(True)
plt.xlabel(r'n$\rightarrow$',fontsize=15)
plt.ylabel(r'Coefficient Magnitude$\rightarrow$',fontsize=15)
plt.title('Loglog Plot of coefficients of $e^{x}$',fontsize=15)
plt.show()
```

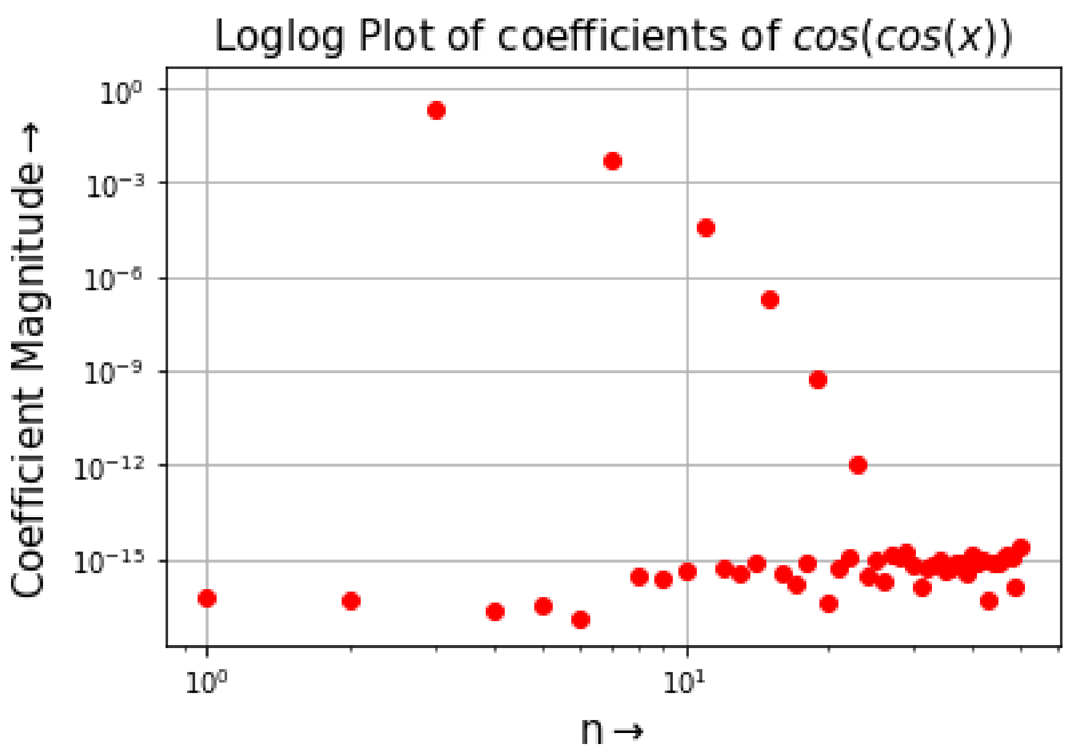




0.5.2 (b) For $cos(cos(x))$

[9]:
```
# semilogy plot
plt.semilogy(range(51),abs(coeff_coscos),'ro')
plt.grid(True)
plt.xlabel(r'n$\rightarrow$',fontsize=15)
plt.ylabel(r'Coefficient Magnitude$\rightarrow$',fontsize=15)
plt.title('Semilog Plot of coefficients for $cos(cos(x))$',fontsize=15)
plt.show()

# loglog plot
plt.loglog(range(51),abs(coeff_coscos),'ro')
plt.grid(True)
plt.xlabel(r'n$\rightarrow$',fontsize=15)
plt.ylabel(r'Coefficient Magnitude$\rightarrow$',fontsize=15)
plt.title('Loglog Plot of coefficients of $cos(cos(x))$',fontsize=15)
plt.show()
```


Semilog Plot of coefficients for $cos(cos(x))$
Coefficient Magnitude →
10^{0}
10^{-3}
10^{-6}
10^{-9}
10^{-12}
10^{-15}
0
10
20
30
40
50
n →


Loglog Plot of coefficients of $cos(cos(x))$
Coefficient Magnitude →
10^{0}
10^{-3}
10^{-6}
10^{-9}
10^{-12}
10^{-15}
10^{0}
10^{1}
n →

0.6 Q4. Finding the Fourier series coefficients: Least Squares approach

[10]:
```
x = np.linspace(0,2*np.pi,401)
x = x[:-1]
y = np.linspace(0,2*np.pi,400)
```

[11]:
```
# We want to solve the matrix equation "Ac = b" where c are the fourier␣
↪coefficients.

"""A"""
A = np.zeros((400,51))
A[:,0] = 1
for i in range(1,26):
    A[:,2*i-1] = np.cos(i*x)
    A[:,2*i] = np.sin(i*x)

"""b"""
b_exp = exp(x)
b_coscos = coscos(x)

"""c"""
c_exp = np.linalg.lstsq(A,b_exp, rcond=None)[0]
c_coscos = np.linalg.lstsq(A,b_coscos,rcond=None)[0]
```

[12]:
```
#Runtime is calculated for comparison with the least squares method.

"""integration approach"""
start1 = timeit.default_timer()
coeff_exp = find_coeff(51,'exp(x)')
elapsed1 = timeit.default_timer() - start1
print('Runtime with integration approach = ',elapsed1)


"""linear square approach"""
start2 = timeit.default_timer()
for i in range(1,26):
    A[:,2*i-1] = np.cos(i*x)
    A[:,2*i] = np.sin(i*x)
b_exp = exp(x)
c_exp = np.linalg.lstsq(A,b_exp, rcond=None)[0]
elapsed2 = timeit.default_timer() - start2
print('Runtime with linear square approach = ', elapsed2)
```

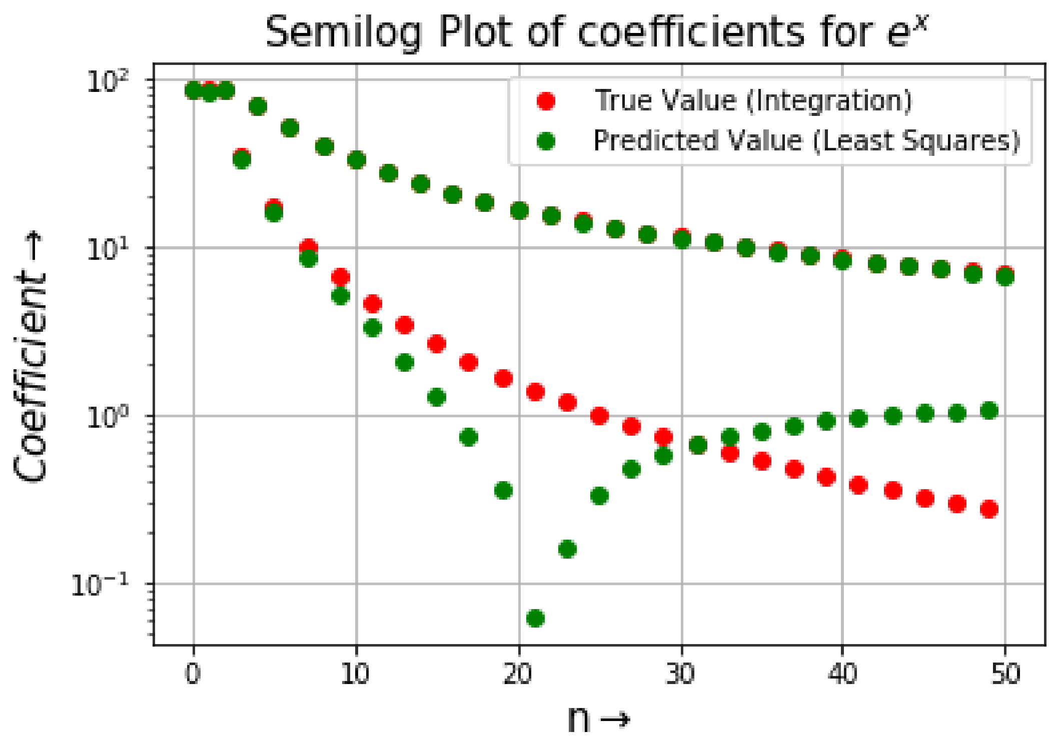
```
Runtime with integration approach =  0.16507640000000023
Runtime with linear square approach =  0.003682799999999986
```

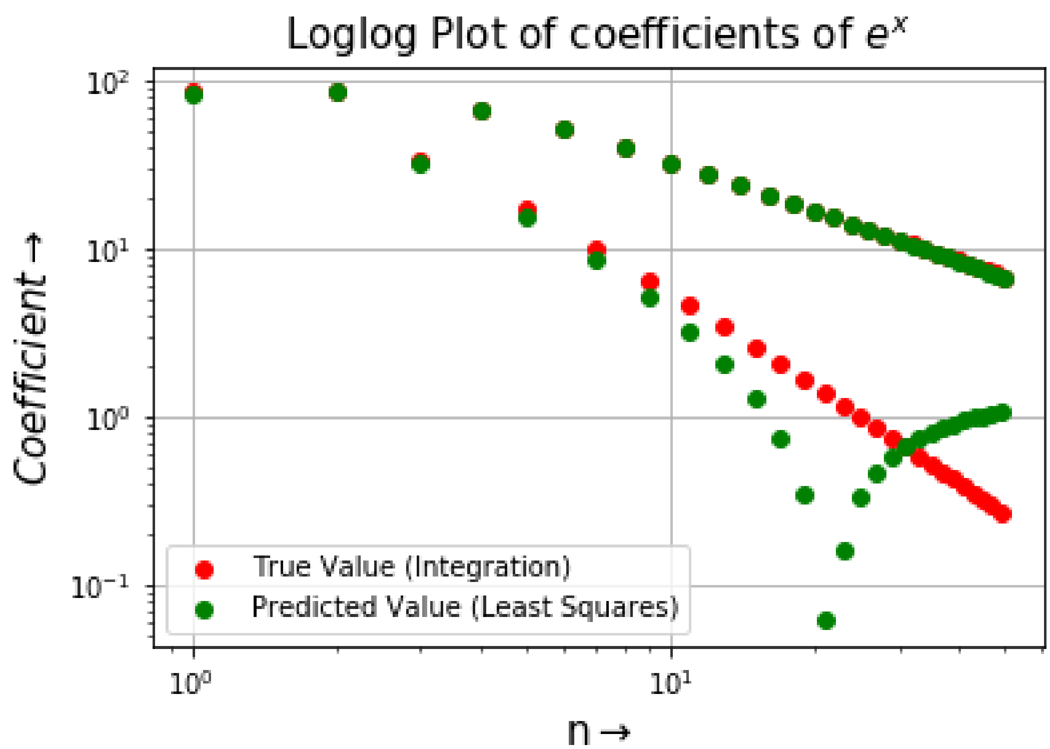
[13]:
```python
# CLearly, the runtime in lstsq approach is found to be 50 times less than the␣
↪direct integration approach
```

0.6.1 Q5. Plots comparing the coefficients by "Integration" and "Least Square" approaches

0.6.2 (a) For e^x

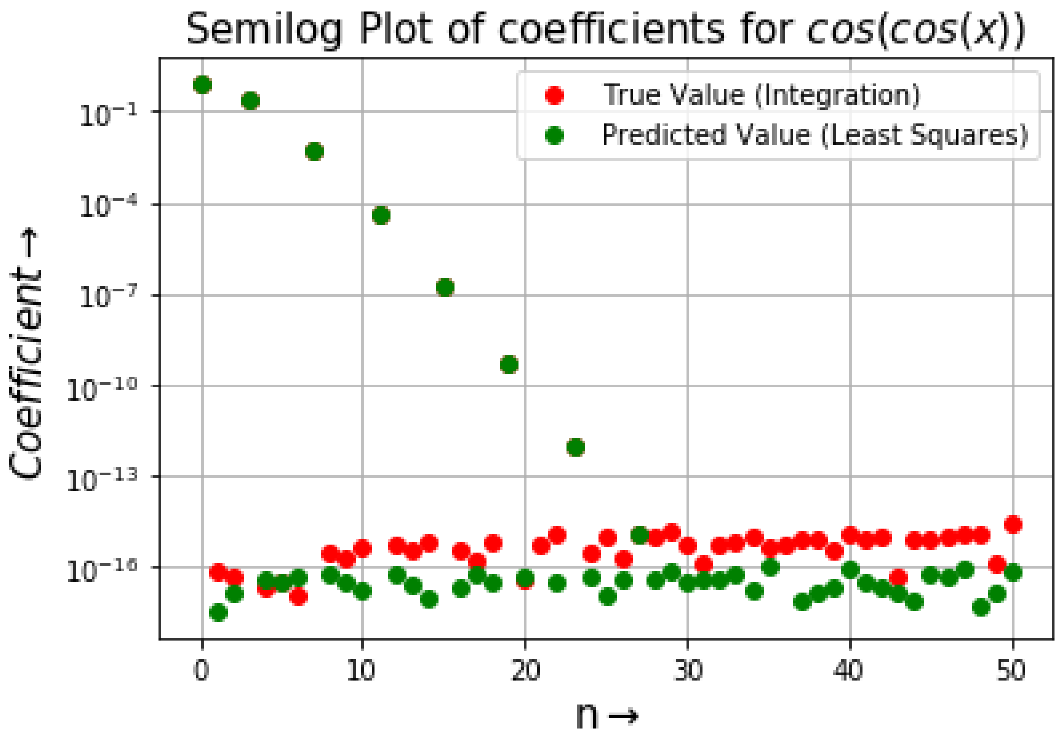
[14]:
```python
# semilogy plot
plt.semilogy(range(51),np.abs(coeff_exp),'ro',label='True Value (Integration)')
plt.semilogy(range(51),np.abs(c_exp),'go',label='Predicted Value (Least␣
↪Squares)')
plt.grid(True)
plt.xlabel(r'n$\rightarrow$',fontsize=15)
plt.ylabel(r'$Coefficient\rightarrow$',fontsize=15)
plt.title('Semilog Plot of coefficients for $e^{x}$',fontsize=15)
plt.legend(loc='upper right')
plt.show()

# loglog plot
plt.loglog(range(51),np.abs(coeff_exp),'ro',label = 'True Value (Integration)')
plt.loglog(range(51),np.abs(c_exp),'go',label='Predicted Value (Least Squares)')
plt.grid(True)
plt.xlabel(r'n$\rightarrow$',fontsize=15)
plt.ylabel(r'$Coefficient\rightarrow$',fontsize=15)
plt.title('Loglog Plot of coefficients of $e^{x}$',fontsize=15)
plt.legend(loc='lower left')
plt.show()
```

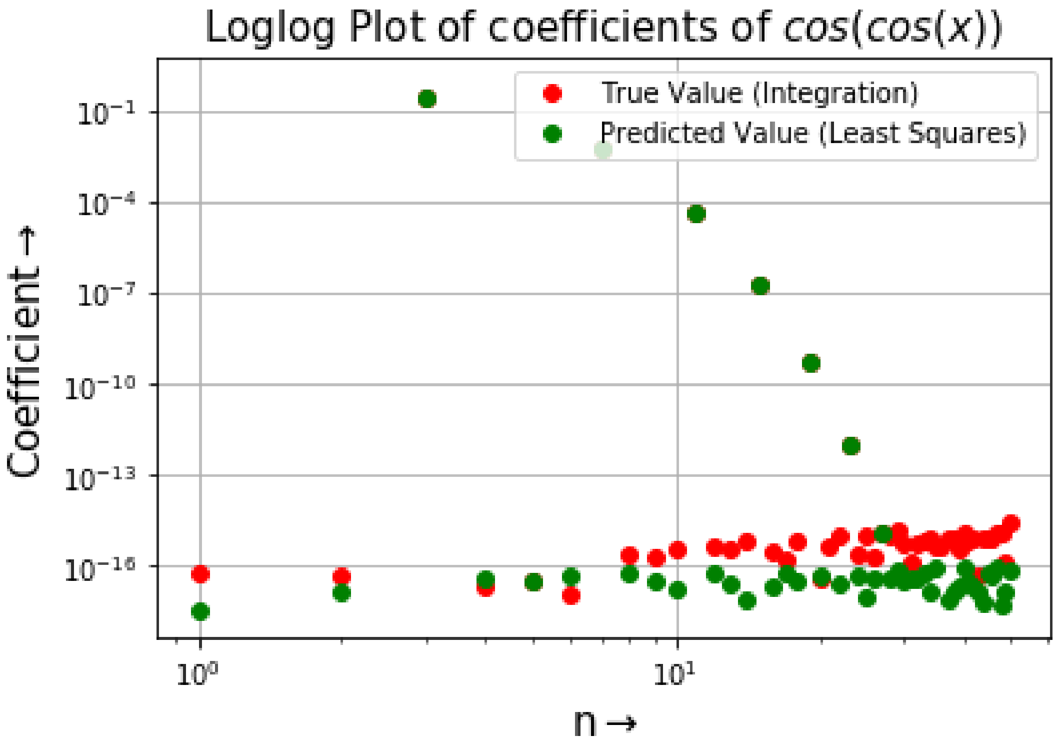

Semilog Plot of coefficients for e^x
True Value (Integration)
Predicted Value (Least Squares)
10^2
10^1
10^0
10^{-1}
0
10
20
30
40
50
Coefficient →
n →


Loglog Plot of coefficients of e^x
True Value (Integration)
Predicted Value (Least Squares)
10^2
10^1
10^0
10^{-1}
10^0
10^1
Coefficient →
n →

0.6.3 (b) For $cos(cos(x))$

[15]:
```
# semilogy plot
plt.semilogy(range(51),abs(coeff_coscos),'ro',label='True Value (Integration)')
plt.semilogy(range(51),abs(c_coscos),'go',label="Predicted Value (Least␣
↪Squares)")
plt.grid(True)
plt.xlabel(r'n$\rightarrow$',fontsize=15)
plt.ylabel(r'$Coefficient\rightarrow$',fontsize=15)
plt.title('Semilog Plot of coefficients for $cos(cos(x))$',fontsize=15)
plt.legend(loc='upper right')
plt.show()

plt.loglog(range(51),abs(coeff_coscos),'ro',label='True Value (Integration)')
plt.loglog(range(51),abs(c_coscos),'go',label="Predicted Value (Least Squares)")
plt.grid(True)
plt.xlabel(r'n$\rightarrow$',fontsize=15)
plt.ylabel(r'Coefficient$\rightarrow$',fontsize=15)
plt.title('Loglog Plot of coefficients of $cos(cos(x))$',fontsize=15)
plt.legend(loc='upper right')
plt.show()
```





0.7 Q6. Calculating the deviation b/w Least square and direct integration cofficients

[16]:
```
deviation_exp = abs(coeff_exp - c_exp)
deviation_coscos = abs(coeff_coscos - c_coscos)
```

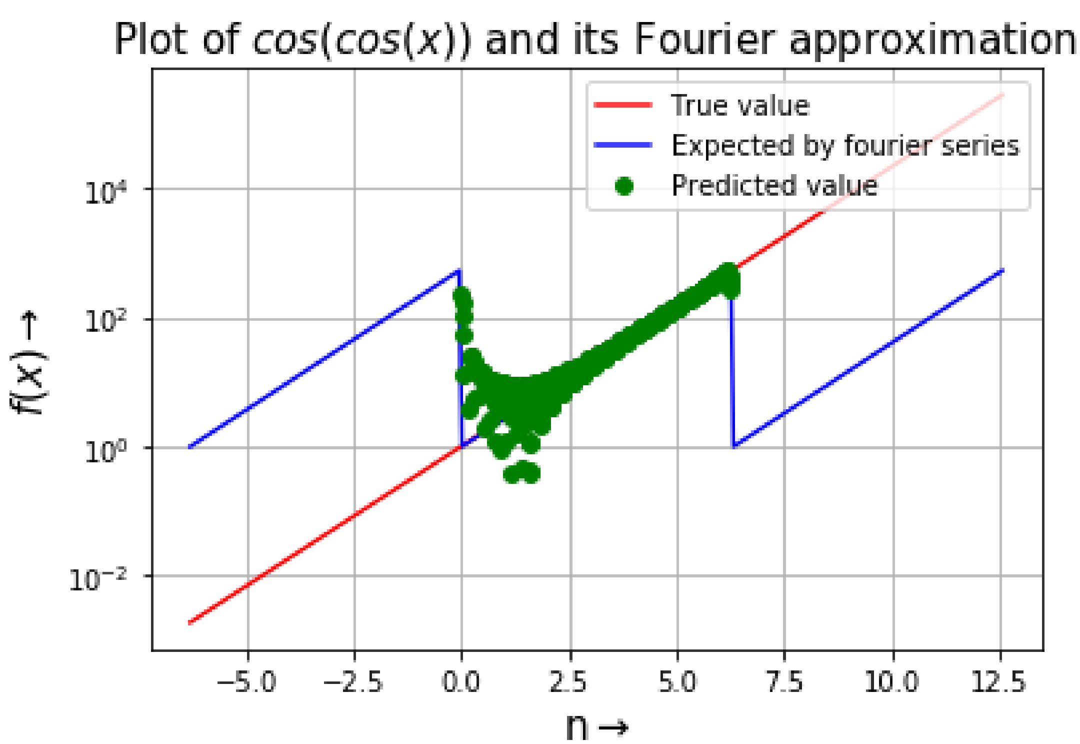
[17]:
```
max_dev_exp = np.max(deviation_exp)
max_dev_coscos = np.max(deviation_coscos)
print("Largest deviation b/w the two sets of coefficients of e^x =", max_dev_exp)
print("Largest deviation b/w the two sets of coefficients of cos(cos(x)) =",
    max_dev_coscos)
```

```
Largest deviation b/w the two sets of coefficients of e^x = 1.332730870335439
Largest deviation b/w the two sets of coefficients of cos(cos(x)) =
2.674677035413032e-15
```

0.7.1 Q7. Plots compare the function values obtained by the "least squares" method with the "true" value

0.7.2 (a) For e^x

[18]:
```
# Ac from the estimated values of c represents the function values

"""true"""
x1 = np.linspace(-2*np.pi,4*np.pi,300)
x2 = np.linspace(0,2*np.pi,100)
tiled = np.tile(x2,3)

"""predicted"""
x3 = np.linspace(0,2*np.pi,401)
x3 = x3[:-1]
predicted_exp = np.matmul(A,c_exp)

plt.plot(x1,exp_x,'-r',label='True value')
plt.semilogy(x1,exp(tiled),'-b',label='Expected by fourier series')
plt.plot(x3,predicted_exp,'go',label="Predicted value")
plt.grid(True)
plt.xlabel(r'n$\rightarrow$',fontsize=15)
plt.ylabel(r'$f(x)\rightarrow$',fontsize=15)
plt.title('Plot of $cos(cos(x))$ and its Fourier approximation',fontsize=15)
plt.legend(loc='upper right')
plt.show()
```



0.7.3 (b) For $cos(cos(x))$

[19]:
```
# Ac from the estimated values of c represents the function values

"""true"""
x1 = np.linspace(-2*np.pi,4*np.pi,300)
coscos_x = coscos(x1)

"""predicted"""
x2 = np.linspace(0,2*np.pi,401)
x2 = x2[:-1]
predicted_coscos = np.matmul(A,c_coscos)

plt.plot(x1,coscos_x,'-r',label='True value')
plt.plot(x2,predicted_coscos,'go',label="Predicted value")
plt.grid(True)
plt.xlabel(r'n$\rightarrow$',fontsize=15)
plt.ylabel(r'$f(x)\rightarrow$',fontsize=15)
plt.title('Plot of $cos(cos(x))$ and its Fourier approximation',fontsize=15)
plt.legend(loc='upper right')
plt.show()
```