Fourier Approximations

EE16B014
G.Harsha Kanaka Eswar

6th April,2018

Abstract

In this report we are going to derive Fourier series of two functions $cos(cos(x))$,e^x in two methods one is by integration and other by least square approach,By integration we can find Fourier coefficients just by the known formula, We can compute the functions back by putting the coefficients in the series and see the accuracy of the approximations.After computing approximation through integration ,we will compute coefficients using Least square approximation and again see the accuracy of approximation.Thus we can know that some functions can't be approximated using Fourier coefficients.

1 Fourier series:

Its an approximation of functions as sum of sinusoids having frequencies as multiples of fundamental frequency of the given function with different amplitudes.
More accurate definition:

An infinite series of trigonometric functions which represents an expansion or approximation of a periodic function, used in Fourier analysis.
Here is the Fourier series expansion:

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

The above expansion is the Fourier expansion,The coefficients a_n, b_n are computed from the below formula:

$$a_0 = \frac{1}{2\pi}\int_0^{2\pi} f(x)dx$$
$$a_n = \frac{1}{\pi}\int_0^{2\pi} f(x)cos(nx)dx$$
$$b_n = \frac{1}{\pi}\int_0^{2\pi} f(x)sin(nx)dx$$

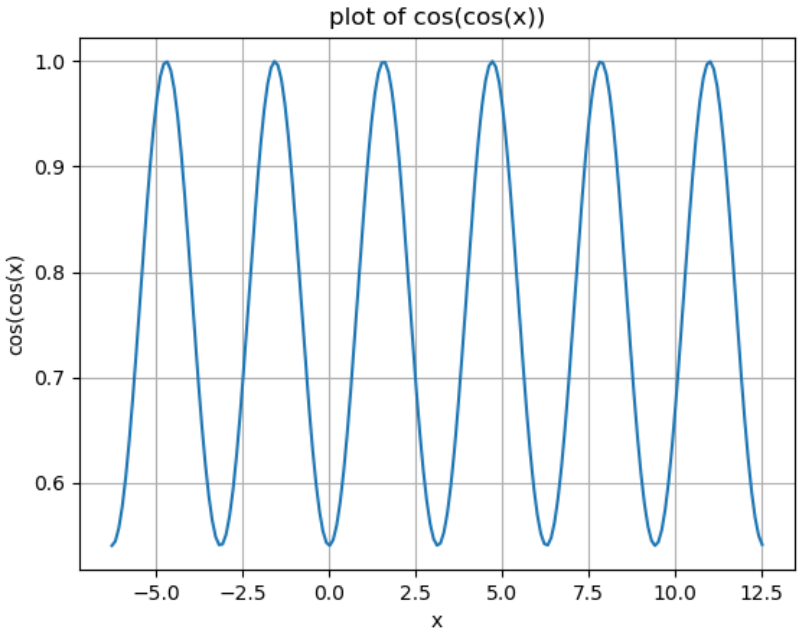
Thus we can get the Fourier approximations by getting those coefficients.

2 Determining Fourier coefficients by Integration:

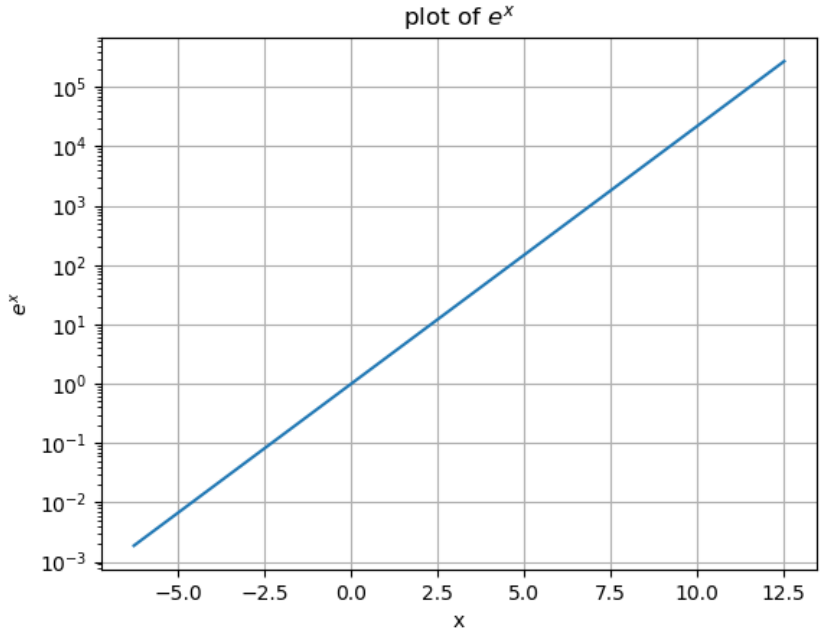
First let us check the periodic nature of functions by plotting the graph from $[-2\pi, 4\pi)$

```
from scipy import *
from pylab import *
t=arange(-2*pi,4*pi,0.1)
a=cos(cos(t))
b=exp(t)
plot(t,a)
title('plot of cos(cos(x))')
xlabel('x')
ylabel('cos(cos(x)')
grid(True)
show()
semilogy(t,b)
title('plot of $e^x$')
xlabel('x')
ylabel('$e^x$')
grid(True)
show()
```

Graph of$cos(cos(x))$:


plot of cos(cos(x))
cos(cos(x))
1.0
0.9
0.8
0.7
0.6
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0
12.5
x

Graph of e^x:


plot of e^x
e^x
10^5
10^4
10^3
10^2
10^1
10^0
10^{-1}
10^{-2}
10^{-3}
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0
12.5
x

We can clearly observe that the $cos(cos(x))$ is periodic but e^x is not periodic.Later we will discuss about the effect of periodic nature on Fourier series.

Now lets define function such that it take two arguments k, x as inputs and gives out $f(x)sin(kx)$ and $f(x)cos(kx)$ as the outputs don't forget to give x as the first argument which will be used by *quad* function for integration.The usage of k is to get all the values of $a'_k s, b'_k s$ just by avoiding to write a new function for every $sin(kx), cos(kx)$.
lets define a function $u(x, k1)$ where $k1$ is an array where the value $k1[0]$ defines the value of k ,the value of $k1[1]$ defines whether $f(x)$ is either $cos(cos(x)$ore^x,the value of $k1[2]$ defines weather we want $sin(kx)$or $cos(kx)$ and returns corresponding output.

```
def u(x,k1):
        a=(1/(pi))
        k,l,m=k1
        if(l==0):
                a*=cos(cos(x))
        if(l==1):
                a*=(exp(x))
        if(m==0):
                a*=cos(k*x)
        if(m==1):
                a*=sin(k*x)
        return a
```

Now we have the function which is to be integrated to get the coefficients, so with this lets compute the coefficients and a vector which will be in the form:

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

```
for i in x:
        a,b=quad(u,0,1,args=([i,0,0]))
```

```
            if(i==0):
                    a=a/2
            d.append(a)
            a1.append(a)
            a,b=quad(u,0,l,args=([i,0,1]))
            if(i>0):
                    d.append(a)
                    b1.append(a)
            a,b=quad(u,0,l,args=([i,1,0]))
            if(i==0):
                    a=a/2
            e.append(a)
            a2.append(a)
            a,b=quad(u,0,l,args=([i,1,1]))
            if(i>0):
                    e.append(a)
                    b2.append(a)
t1=arange(-2*pi,4*pi,0.1)
q=0
for k1 in t1:


            for i in range(0,25):
                    q=q+b1[i]*sin((i+1)*k1)+a1[i+1]*cos((i+1)*k1)
            q=q+a1[0]
            cg.append(q)
            q=0

j=array(d)
l=array(e)
show()
```

By the above code we have also computed the function values from coefficients.

3 Least Square approach:

Its is a way of solving equations where the coefficients are unknown but the function values are known at different points of input variables then we can get the best fitting coefficients by using this approach.

4 Computation of coefficients using least square approach:

We have the equation related to the sinusoid,which may not be exact(because its an infinite series) but we can get best fit for the approximation.
Approximation:

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

As we already understood that the least square approach is going to give the best values of coefficients,since we know the sinusoid values and function output at different values of x so we will form a matrix as shown below:

$$\begin{pmatrix} 1 & cosx_1 & sinx_1 & \dots & cos25x_1 & sin25x_1 \\ 1 & cosx_2 & sinx_2 & \dots & cos25x_2 & sin25x_2 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & cosx_{400} & sinx_{400} & \dots & cos25x_{400} & sin25x_{400} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ b_1 \\ ... \\ a_{25} \\ b_{25} \end{pmatrix} = \begin{pmatrix} f(x_1) \\ f(x_2) \\ \dots \\ f(x_{400}) \end{pmatrix}$$

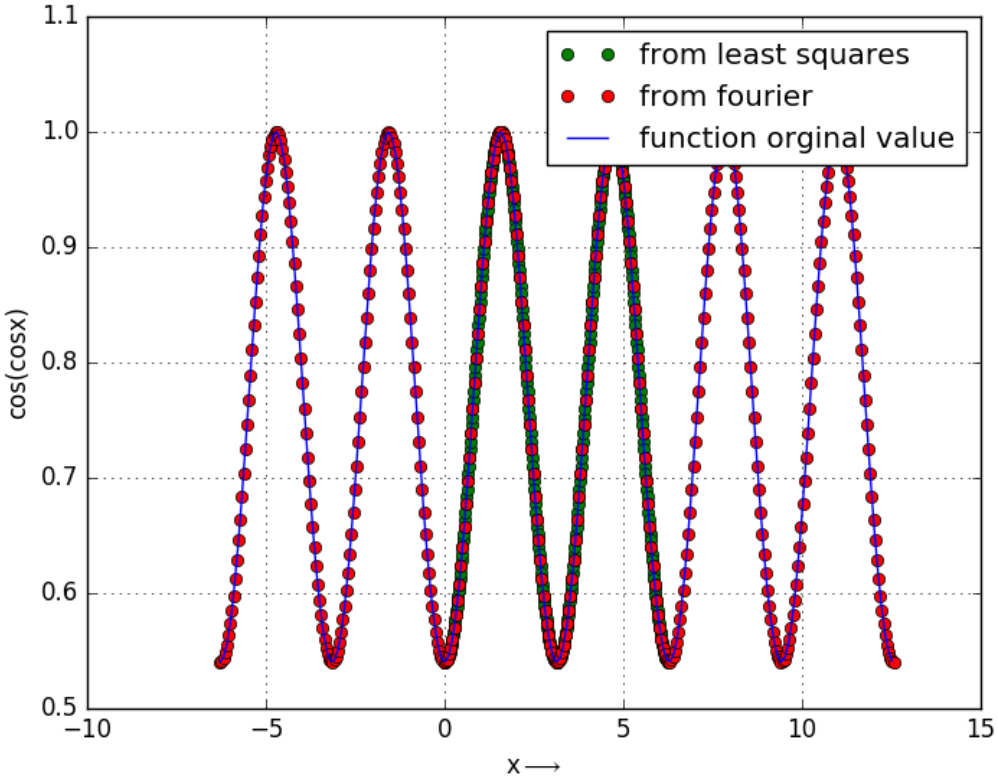
where we will define the matrices as Ac=b,where c is the coefficient vector we want ,so lets use lstsq() for computation:

```
x=linspace(0,2*pi,401)
x=x[:-1]
b=cos(cos(x))
A=zeros((400,51))
A[:,0]=1
for k in range (1,26):
        A[:,2*k-1]=cos(k*x)
        A[:,2*k]=sin(k*x)
c1=lstsq(A,b)[0]
cl=abs(lstsq(A,b)[0])


fig1=figure()
semilogy(cl,'go',label="least squares")
semilogy(j,'ro',label="from fourier")
```

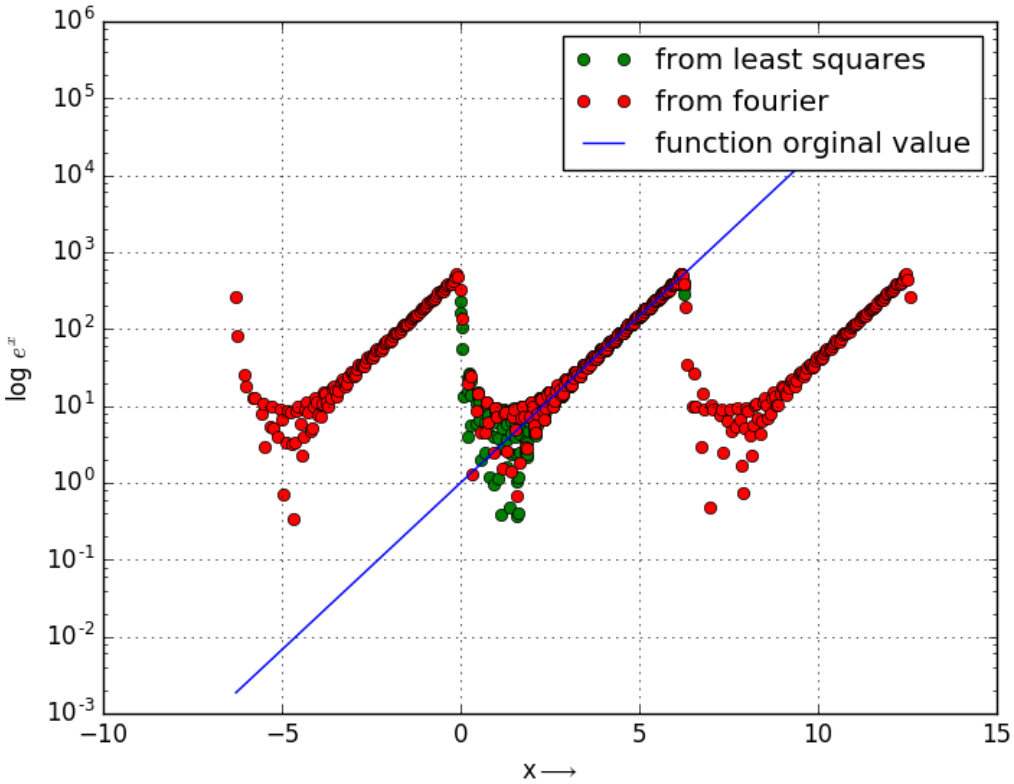
```
legend(loc="upper right")
plt.ylabel("log$\mid$fourier cofficients$\mid$")
plt.xlabel("n$\longrightarrow$")
plt.suptitle("Comparision of fourier cofficients of cos(cosx) \nfrom
grid()

fig2=figure()
loglog(c1,'go',label="least squares")
loglog(j,'ro',label="from fourier")
legend(loc="upper right")
plt.ylabel("log$\mid$fourier cofficients$\mid$")
plt.xlabel("logn$\longrightarrow$")


plt.suptitle("Comparision of fourier cofficients of cos(cosx) \nfrom
#e**(x):
b1=exp(x)
c11=abs(lstsq(A,b1)[0])
c2=lstsq(A,b1)[0]
grid()

fig3=figure()
semilogy(l,'r.',label="from fourier")
semilogy(c11,'go',label="least squares")
plt.suptitle("Comparision of fourier cofficients of $e^x$ \nfrom lea
legend(loc="upper right")
plt.ylabel("log$\mid$fourier cofficients$\mid$")
plt.xlabel("n$\longrightarrow$")
grid()

fig4=figure()
loglog(l,'r.',label="from fourier")
loglog(c11,'go',label="least squares")
plt.suptitle("Comparision of fourier cofficients of $e^x$ \nfrom lea
legend(loc="upper right")
plt.ylabel("log$\mid$fourier cofficients$\mid$")
plt.xlabel("logn$\longrightarrow$")
grid()

#largest deviation in cos(cos(x))
```

```
d1=abs(j1-c1)
e1=max(d1)
#largest deviation in e**(x)
d2=abs(l1-c2)
e2=max(d2)
print("the largest deviation between values of cofficients of cos(co
print (e1)
print("the largest deviation between values of cofficients of exp(x)
print (e2)

C=dot(A,c1)
E=dot(A,c2)
fig5=figure()
plot(x,C,'go',label="from least squares")
plot(t1,cg,'ro',label="from fourier")
plot(t,co,label="function orginal value")
legend(loc="upper right")
plt.xlabel("x$\longrightarrow$")
plt.ylabel("cos(cosx)")
grid()

fig5=figure()
semilogy(x,E,'go',label="from least squares")
semilogy(t1,eg,'ro',label="from fourier")
semilogy(t,ex,label="function orginal value")
plt.ylabel("log $e^x$")
plt.xlabel("x$\longrightarrow$")
legend(loc="upper right")
grid()
show()
```

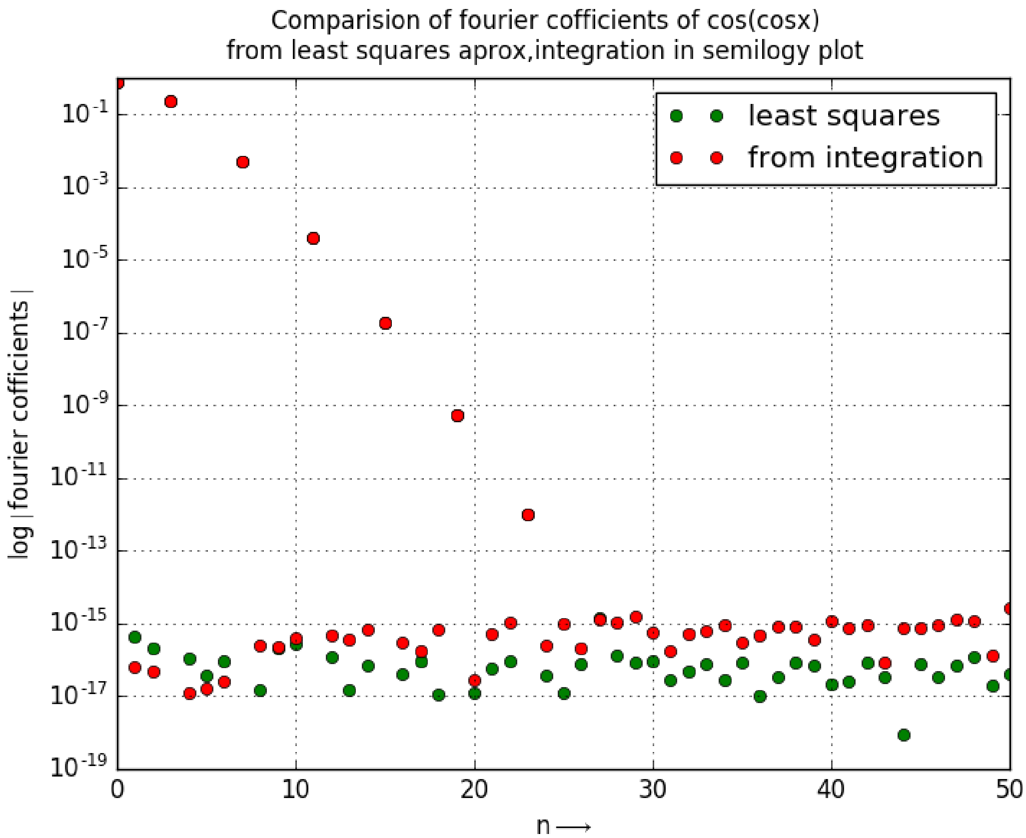
Thus we got the coefficients,so lets move on and plot the graphs:
Graph of original function and its estimations:
$cos(cos(x))$:


from least squares
from fourier
function orginal value
cos(cosx)
x ⟶
1.1
1.0
0.9
0.8
0.7
0.6
0.5
−10
−5
0
5
10
15

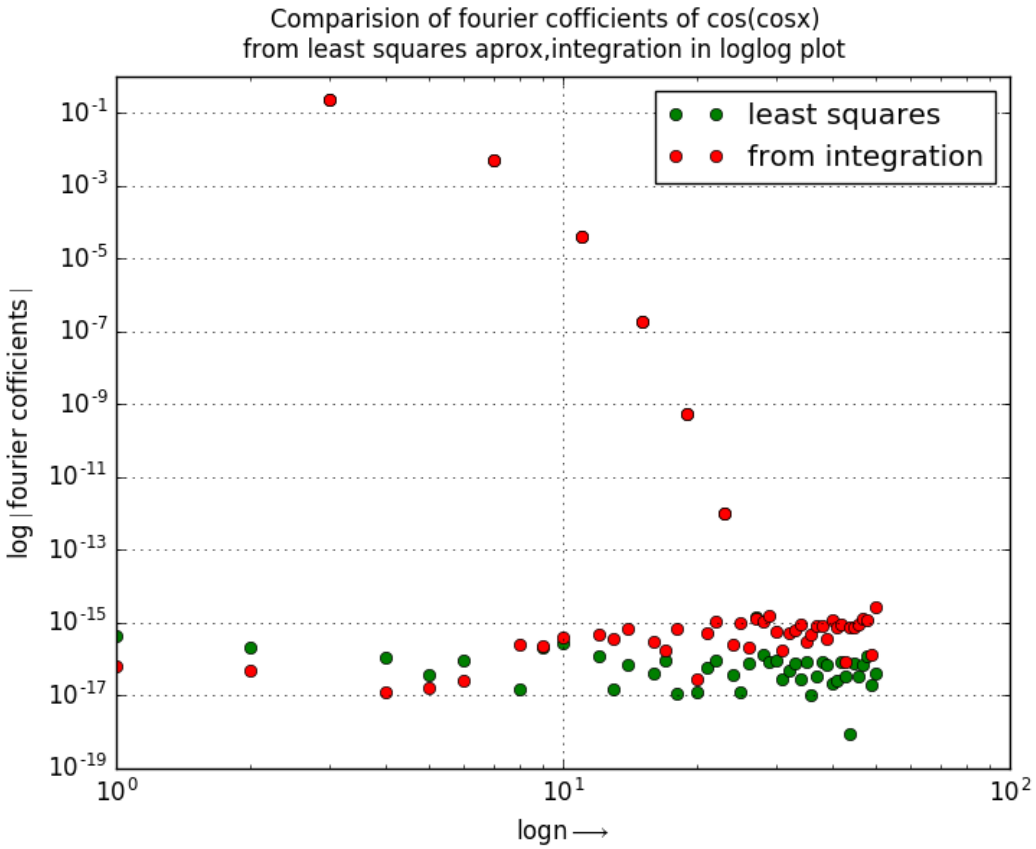
e^x:



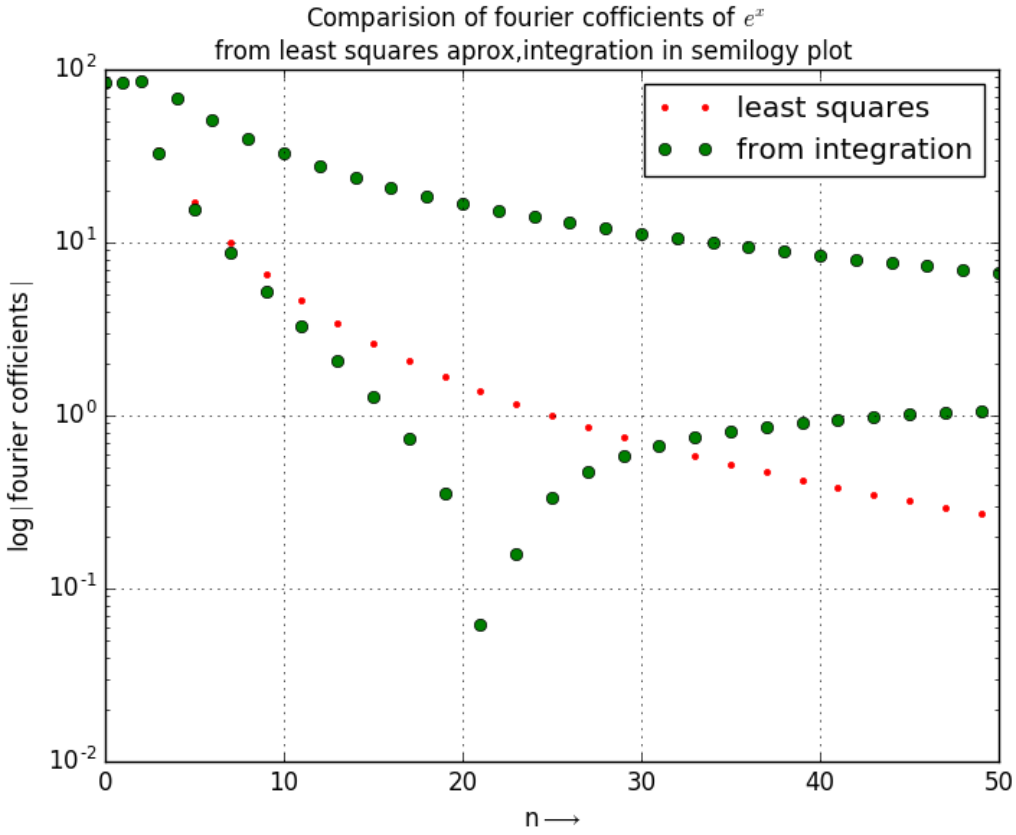
From above two graphs we can clearly see that the $cos(cos(x))$ is almost has a good approximation,where as the e^xdoesn't because the Fourier approximation is good approximation for periodic function and its is taken that the function should be of period 2π so its a giving an output repeating at the same periodic interval.Similarly least squares is giving a different graph though it is the best fit because we are computing the coefficient approximation in period 0 to 2π.where it is near to the function value as seen in graph.
Now lets see the error in approximation for the computation:
$cos(cos(x))$ semilogy:


Comparision of fourier cofficients of cos(cosx)
from least squares aprox,integration in semilogy plot
least squares
from integration
log |fourier cofficients|
n ⟶
10^-1
10^-3
10^-5
10^-7
10^-9
10^-11
10^-13
10^-15
10^-17
10^-19
0
10
20
30
40
50

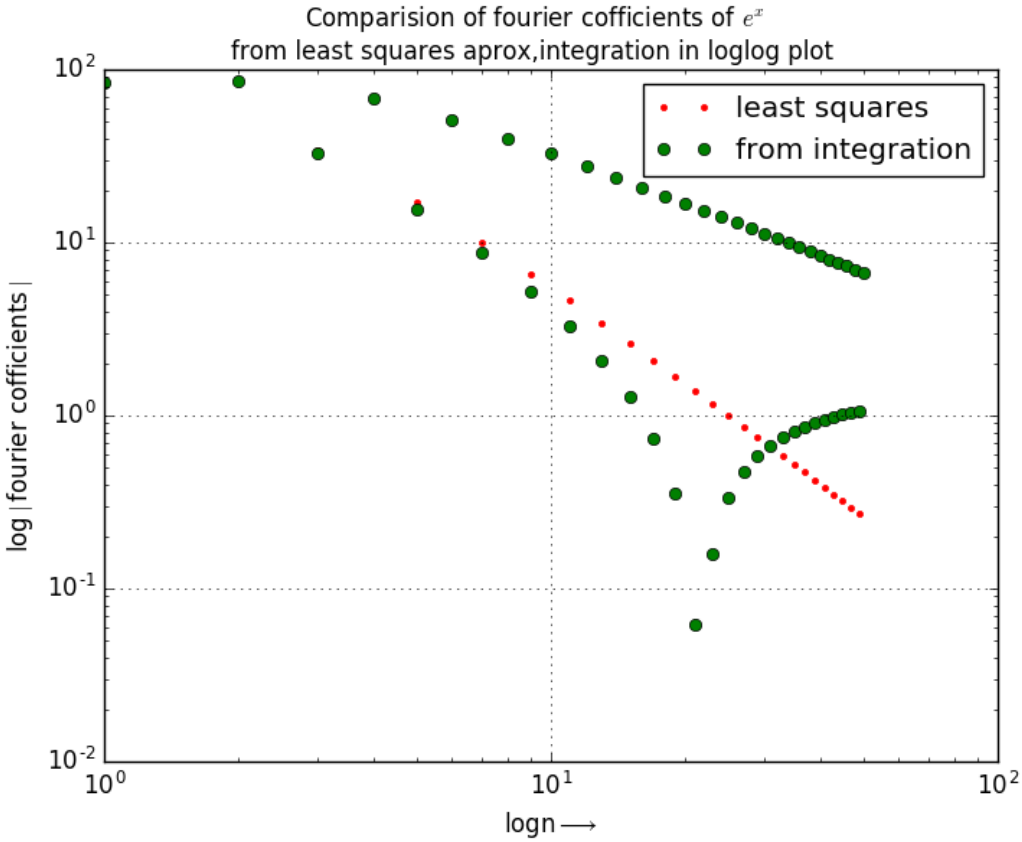
loglog plot:


Comparision of fourier cofficients of cos(cosx)
from least squares aprox,integration in loglog plot
least squares
from integration
log |fourier cofficients|
logn ⟶
10^{-1}
10^{-3}
10^{-5}
10^{-7}
10^{-9}
10^{-11}
10^{-13}
10^{-15}
10^{-17}
10^{-19}
10^{0}
10^{1}
10^{2}

e^x semilogy:


Comparision of fourier cofficients of e^x
from least squares aprox,integration in semilogy plot
least squares
from integration
log |fourier cofficients|
n ⟶
10^2
10^1
10^0
10^{-1}
10^{-2}
0
10
20
30
40
50

loglog:



5 Conclusions:

- 1) The b_n coefficients for cos(cos(x)) are nearly zero.Reason:Because cos(cos(x)) is even function whereas sin kx is an odd function so multiplication of both functions is an odd function with period of 2π.Since we integrate the function over a time period of 2π to get b_n's, which is nothing but the period of odd function ,we get zero(integration of an odd function over an interval which is multiple of it's period is zero)

- cos(cos(x)) got it's both of the Fourier and least square approach almost same as the function value whereas the e^x it is not so good.Reason:This is because the Fourier series approximation is best suited for periodic signals and it is also designed to approximate periodic signals like cos(cos(x)).where as the function e^x is not periodic so the approximation fails,and for such type of non periodic functions we use Fourier transform with some conditions.

- In plot4 the Fourier coefficients of e^x in loglog scale are nearly linear.Reason:Because the Fourier coefficients after calculating the integral turn out to be in order of k^{-2} when drawn in loglog scale with respect to k it looks linear.

6 The largest deviation in error:

.

The largest deviation between values of coefficients of cos(cos(x)), when calculated from least squares and Fourier: 2.65318862443e-15

The largest deviation between values of coefficients of exp(x), when calculated from least squares and Fourier: 1.33273087034