LAB 4

HARSHA VARDHAN EE17B061

February 27, 2019

1 Report

In this lab we are trying to fit data computed using fourier series and lstsq to original functions e^x and $cos(cosx)$.

code

1.1 import statements

```
from pylab import *
from scipy.integrate import quad
from scipy.linalg import lstsq
```

1.2 functions defined

```
    fexp = np.exp
    fcos = lambda x:np.cos(np.cos(x))

def FourierCoeffs(func=fcos,N=25):# To N+1 a's and N b's for a given func in [0,2*pi]
    u,v = lambda x,k:func(x)*cos(k*x),lambda x,k:func(x)*sin(k*x)
    c = [quad(u,0,2*pi,0)[0]/(2*pi)]#constant or a0
    for i in range(1,N+1):   c.extend([quad(u,0,2*pi,i)[0]/pi,quad(v,0,2*pi,i)[0]/pi])
    return array(c).reshape(2*N+1,1)

def makeA(t,N=25):
    '''rtns a matrix with cos and sin vals for Fs'''
    A = np.ones(t.shape)
    for i in range(1,26):   A = hstack([A,cos(i*t),sin(i*t)])
    return A
```

1.3 computing values for plotting

```
    N = 400
t = np.linspace(0,2*pi,N+1)[:-1].reshape(N,1)#column vector #to find the fourier coeffs
```

```
t1 = np.linspace(-2*pi,4*pi,N,endpoint = False).reshape(N,1)# to plot curves

A,A1 = makeA(t),makeA(t1)
c1,c2 = FourierCoeffs(fexp), FourierCoeffs(fcos)# fourier coeffs
est1,est2 = lstsq(A,fexp(t))[0],lstsq(A,fcos(t))[0]# lstsq coeffs
```

first we plotted e^x in semilog and cos(cosx) in semilogy and linear scales.These are shown in fig1 and fig2.The fourier series of e^x diverges from e^x since it is not periodic but it fits well for $cos(cos(x))$

code for figures 1 and 2

```
plt.figure(1)
aexp = fexp(t)
plt.semilogy(t1,fexp(t1),label='fexp_Normal')   #original curve
plt.semilogy(r_[t-2*np.pi,t,t+2*np.pi],r_[aexp,aexp,aexp],label='expected fourier series')
plt.semilogy(t1,dot(A1,c1),label='fexp_FS')     #fourier series fit
plt.semilogy(t1,dot(A1,est1),label='fexp_lstsq')#lstsq fit
plt.xticks(np.arange(-2*pi,4*pi+0.1,pi),[r'${}\pi$'.format(i) for i in range(-2,5)])

plt.figure(2)
ACOS = fcos(t)
plt.plot(t1,fcos(t1),label='fcos_Normal')
plt.plot(r_[t-2*np.pi,t,t+2*np.pi],r_[ACOS,ACOS,ACOS],label='expected fourier series')
plt.plot(t1,dot(A1,c2),label='fcos_FS')
plt.plot(t1,dot(A1,est2),label='fcos_lstsq')
plt.xticks(np.arange(-2*pi,4*pi+0.1,pi),[r'${}\pi$'.format(i) for i in range(-2,5)])
```

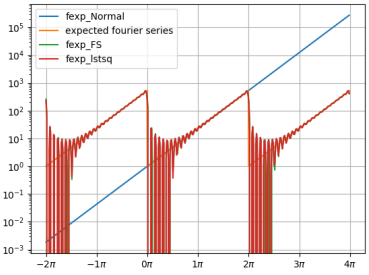

Figure 1: plot1

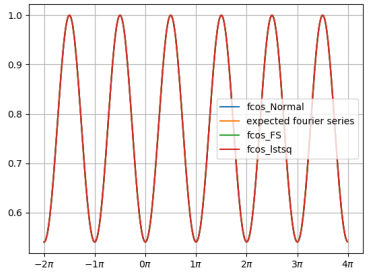

Figure 2: plot2

Then we calculated fourier series coefficients of the functions and plotted them in both semilog and loglog axes.The plots for these are shown from fig3

to fig6.

code for figures 3 to 6

```
XF = arange(1,52)
figure(3);semilogy();scatter(XF,abs(c1),c='r',label='fexpFcoeffs')
figure(4);loglog();scatter(XF,abs(c1),c='r',label='fexpFcoeffs')
figure(5);semilogy();scatter(XF,abs(c2),c='r',label='fcosFcoeffs')
figure(6);loglog();scatter(XF,abs(c2),c='r',label='fcosFcoeffs')

figure(3);semilogy();scatter(XF,abs(est1),c='g',label='fexpLstsq')
figure(4);loglog();scatter(XF,abs(est1),c='g',label='fexpLstsq')
figure(5);semilogy();scatter(XF,abs(est2),c='g',label='fcosLstsq')
figure(6);loglog();scatter(XF,abs(est2),c='g',label='fcosLstsq')
```

first we plotted e^x in semilog and cos(cosx) in semilogy and linear scales.These are shown in fig1 and fig2.The fourier series of e^x diverges from e^x since it is not periodic but it fits well for $cos(cos(x))$

Since cos(cosx) is an even function coefficients of sine are nearly zero(which should be zero but not,because of error in integration) as sin(x) is an odd function.

Also,coefficients decay faster for cos(cosx) because it is periodic.But for e^x, we need more frequencies to fit the function as it is not periodic.

Then we used least square method to find the coefficients(for best fit to the function) of sinusoids

by creating a matrix A containing sine and cos harmonics upto 25^{th} order and column vector containing original values of functions called b.
By solving Ac=b we can get c which contains the fourier coefficients.

linearity of coefficients can be obtained by writing cos and sin in terms of e^{jw},e^{-jw}.
we will get a relations for a and b as follows

$$a_k = \int_0^{2\pi} e^x cos(x)dx = constant * k/(1+k^2)$$

$$b_k = \int_0^{2\pi} e^x cos(x)dx = constant/(1+k^2)$$

$$\log(a_k) \approx \log(constant) - \log(k) \tag{1}$$

$$\log(b_k) \approx \log(constant) - 2\log(k) \tag{2}$$

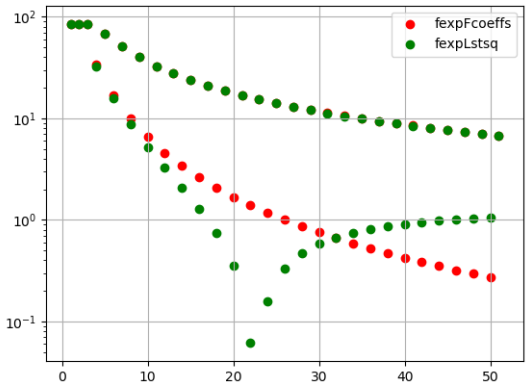

Figure 3: plot3

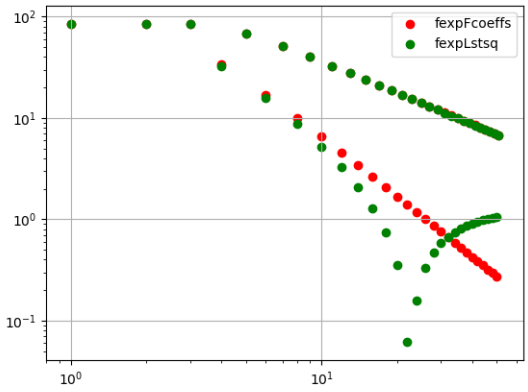

Figure 4: plot4

this is why b_k has a higher negative slope compared to a_k for exp.
In a similar way we can see that the coefficients for $cos(cos(x))$ are linear in semilog scale Then we plotted these values from lstsq method in the corresponding plots.And hence the above plots are obtained.

Then we got the maximum deviation between coefficients calculated in both these methods as 1.3327308 for e^x and 2.5816134*10^{-15} for cos(cosx)

Lastly,we plotted the functions using the fourier series (and least square estimate).we find that both the functions fit very well for cos(cosx) but not for

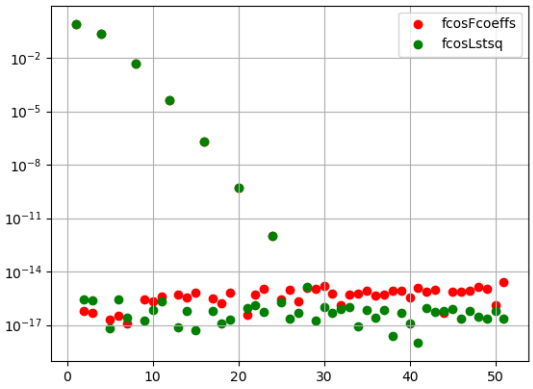

Figure 5: plot5

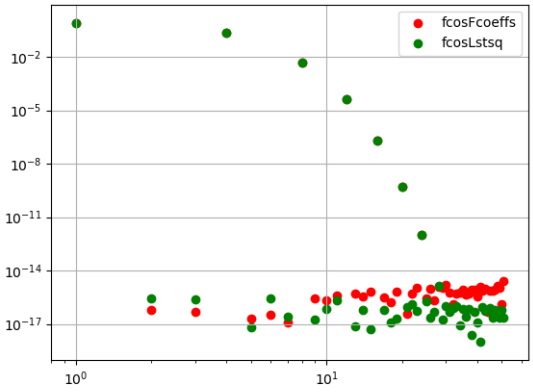

Figure 6: plot6

e^x because cos(cosx) is periodic(with period 2π) and e^x is not.The least square is more accurate compared to fourier series as it tries to fit using the available frequencies rather than assume the existence of higher frequencies.

```
from pylab import *
from scipy.integrate import quad
from scipy.linalg import lstsq

fexp = np.exp
fcos = lambda x:np.cos(np.cos(x))
```

```
def FourierCoeffs(func=fcos,N=25):# To N+1 a's and N b's for a given func in [0,
    u,v = lambda x,k:func(x)*cos(k*x),lambda x,k:func(x)*sin(k*x)
    c = [quad(u,0,2*pi,0)[0]/(2*pi)]#constant or a0
    for i in range(1,N+1):    c.extend([quad(u,0,2*pi,i)[0]/pi,quad(v,0,2*pi,i)[0
    return array(c).reshape(2*N+1,1)

def makeA(t,N=25):
    '''rtns a matrix with cos and sin vals for Fs'''
    A = np.ones(t.shape)
    for i in range(1,26):    A = hstack([A,cos(i*t),sin(i*t)])
    return A

N = 400
t = np.linspace(0,2*pi,N+1)[:-1].reshape(N,1)#column vector #to find the fourier
t1 = np.linspace(-2*pi,4*pi,N,endpoint = False).reshape(N,1)# to plot curves

A,A1 = makeA(t),makeA(t1)
c1,c2 = FourierCoeffs(fexp), FourierCoeffs(fcos)# fourier coeffs
est1,est2 = lstsq(A,fexp(t))[0],lstsq(A,fcos(t))[0]# lstsq coeffs

plt.figure(1)
aexp = fexp(t)
plt.semilogy(t1,fexp(t1),label='fexp_Normal')    #original curve
plt.semilogy(r_[t-2*np.pi,t,t+2*np.pi],r_[aexp,aexp,aexp],label='expected_fourie
plt.semilogy(t1,dot(A1,c1),label='fexp_FS')      #fourier series fit
plt.semilogy(t1,dot(A1,est1),label='fexp_lstsq')#lstsq fit
plt.xticks(np.arange(-2*pi,4*pi+0.1,pi),[r'${}\pi$'.format(i) for i in range(-2

plt.figure(2)
ACOS = fcos(t)
plt.plot(t1,fcos(t1),label='fcos_Normal')
plt.plot(r_[t-2*np.pi,t,t+2*np.pi],r_[ACOS,ACOS,ACOS],label='expected_fourier_se
plt.plot(t1,dot(A1,c2),label='fcos_FS')
plt.plot(t1,dot(A1,est2),label='fcos_lstsq')
plt.xticks(np.arange(-2*pi,4*pi+0.1,pi),[r'${}\pi$'.format(i) for i in range(-2

XF = arange(1,52)
figure(3);semilogy();scatter(XF,abs(c1),c='r',label='fexpFcoeffs')
figure(4);loglog();scatter(XF,abs(c1),c='r',label='fexpFcoeffs')
figure(5);semilogy();scatter(XF,abs(c2),c='r',label='fcosFcoeffs')
figure(6);loglog();scatter(XF,abs(c2),c='r',label='fcosFcoeffs')

figure(3);semilogy();scatter(XF,abs(est1),c='g',label='fexpLstsq')
figure(4);loglog();scatter(XF,abs(est1),c='g',label='fexpLstsq')
figure(5);semilogy();scatter(XF,abs(est2),c='g',label='fcosLstsq')
figure(6);loglog();scatter(XF,abs(est2),c='g',label='fcosLstsq')
```

```
print("max diff b/w Fcoeffs and lstsq in fexp is {[0]} \
and fcos is {[0]}".format(max(abs(c1-est1)),max(abs(c2-est2))))
print('Close' if allclose(A.dot(est1),fexp(t),atol=1e-8) else 'not Close')

for i in range(1,7):
    figure(i);legend();grid(True);#savefig('Figure_%s.png'%i)
show()
```