EE2703 Assignment 9
Spectra of non-periodic signals

Pranav Phatak, EE19B105

3rd June 2021

1 Introduction

In this assignment, we perform FFT on aperiodic functions by periodically extending them. In particular, we look at functions which are normally continuous but become discontinuous when periodically extended, hence leading to Gibbs Phenomenon in their Fourier Transforms. We also look at Windowing while computing the FFT.

2 Examples

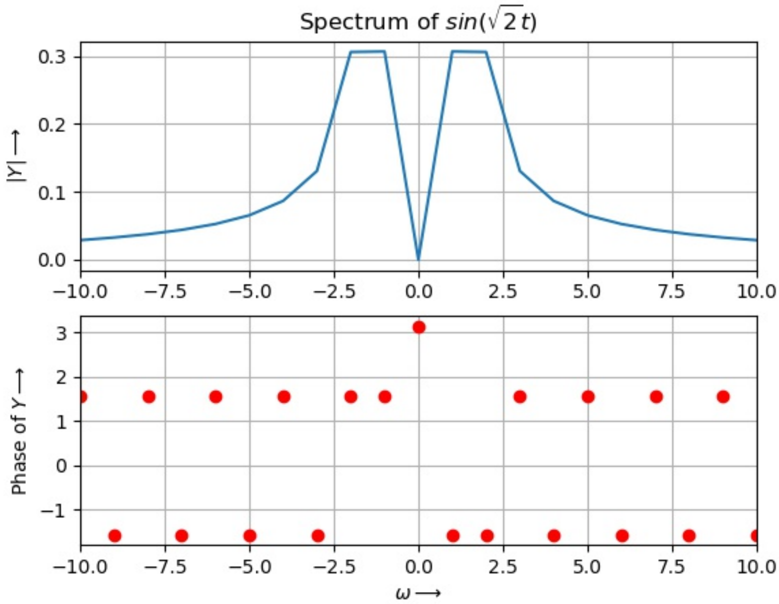
We plot the spectra of the examples given in the assignment. This is easily done by the fft() and fftshift() functions, as we did in the last assignment.

The examples discussed are $sin(\sqrt{2}t)$ and $(\sqrt{2}t)\times w(t)$ where $w(t)$ is essentially a modified rect() function in the middle of the chosen window with half of the time having positive value. Mathematically,

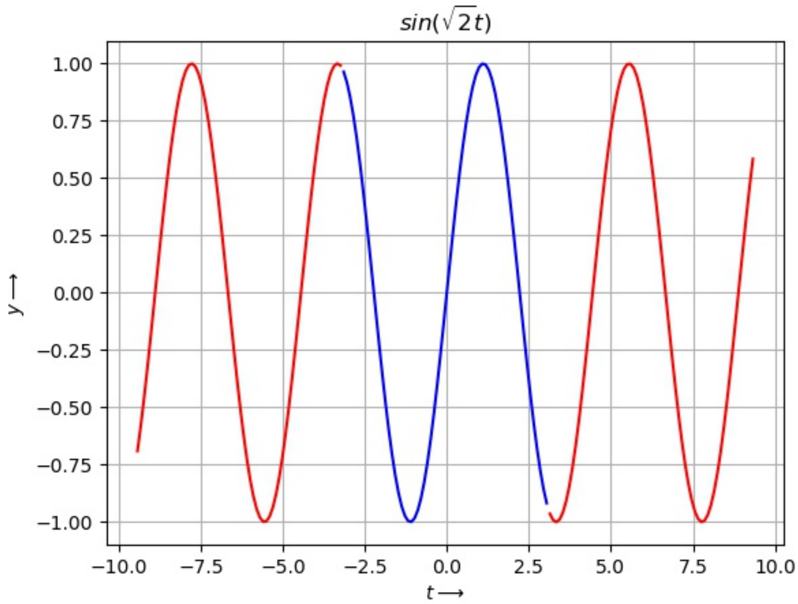
$w(t) = (0.54 + 0.46cos(\frac{2\pi N}{N-1}))\times rect(\frac{t}{(N-1)/2})$

Here, N is number of points taken, which are taken as 64 and 256 and analyzed.

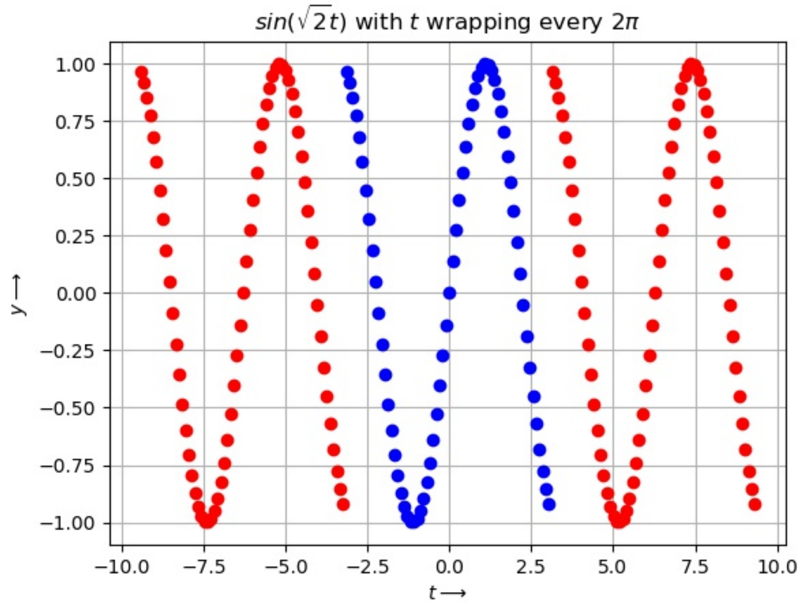
Following is the plot of FFT of $sin(\sqrt{2}t)$ for ω from [-π, +π]



Following is the plot of the pure function :

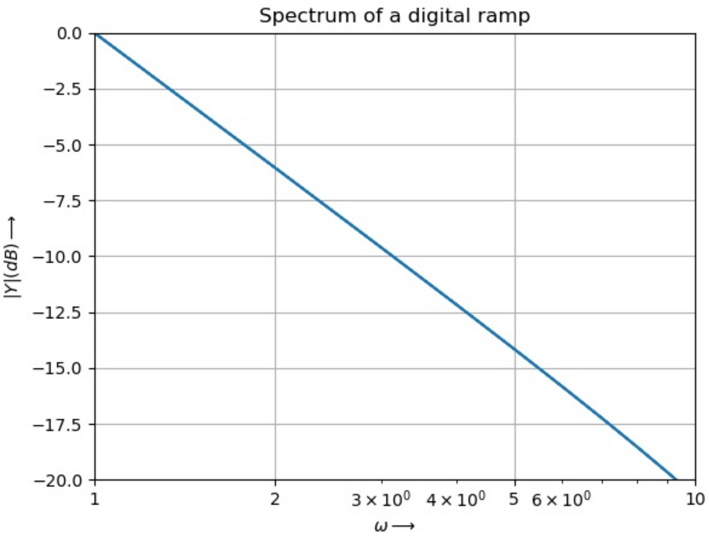


Following is the discretized plot of the function, periodically extended from [-π, +π] to [-∞, +∞] with 64 samples in each period.



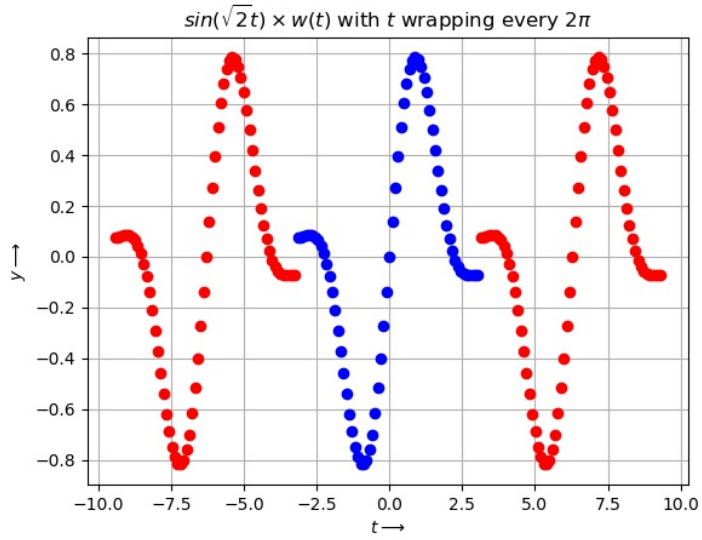
In the case of this function, since the period is **made** to be 2π, while the function is aperiodic in discrete-time (periodic in continuous-time with non-integral period of $\sqrt{2}\pi$) hence there arise discontinuities at the end points of each period, which manifests as wiggles in Fourier Domain (also known as Gibbs Phenomenon).

The magnitude of these non harmonic components of the FFT decay proportional to $\frac{1}{\omega}$. To illustrate this fact, we plot the spectrum of the periodically extended ramp function.



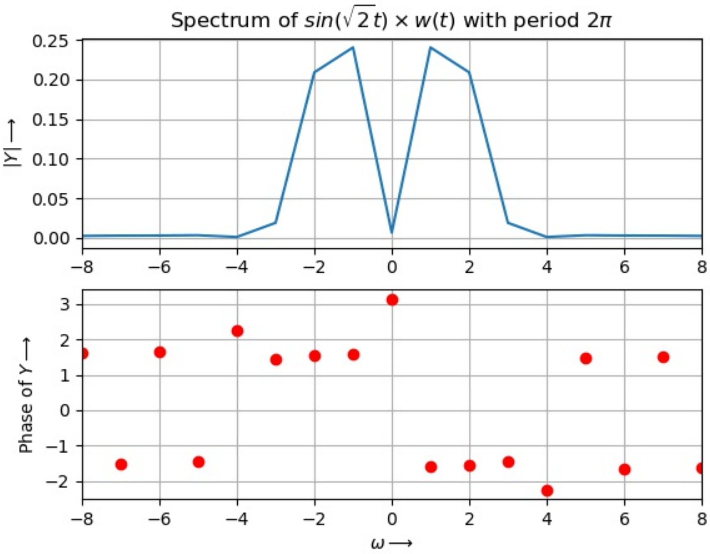
Since the log plot of the magnitude of the FFT coefficients of the ramp function decreases linearly with ω with slope approximately -20dB, we can say that the magnitude is inversely proportional to ω.

Now we plot the function $sin(\sqrt{2}t)\times w(t)$ against t, periodically extended with period 2π.

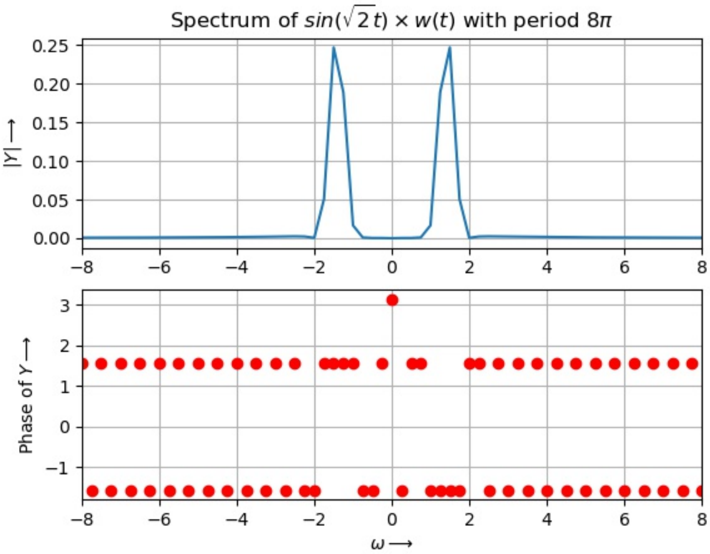


Now we plot the spectrum of this function.

```
# Spectrum of sin(sqrt(2)t)*w(t) with period 2pi


t=linspace(-np.pi,np.pi,65)[:-1]
dt=t[1]-t[0];
fmax=1/dt
n=np.arange(64)
wnd=fftshift(0.54+0.46*np.cos(2*np.pi*n/63))
y=np.sin(np.sqrt(2)*t)*wnd
y[0]=0
y=fftshift(y)
Y=fftshift(fft(y))/64.0
w=linspace(-np.pi*fmax,np.pi*fmax,65)[:-1]
figure(6)
subplot(2,1,1)
plot(w,abs(Y))
xlim([-8,8])
ylabel(r"$|Y|\longrightarrow$")
title(r"Spectrum of $sin(\sqrt{2}t)\times w(t)$ with period $2\pi$")
grid(True)
subplot(2,1,2)
plot(w,np.angle(Y),'ro')
xlim([-8,8])
ylabel(r"Phase of $Y\longrightarrow$")
xlabel(r"$\omega\longrightarrow$")
grid(True)
savefig("Spectrum of sin(sqrt(2)t)w(t) with period 2pi.jpg")
```



Now we plot the spectrum of the same function, using the same method but now we extend it periodically with period 8π, keeping accuracy same (that is, total 256 points sampled between [-4π, +4π])



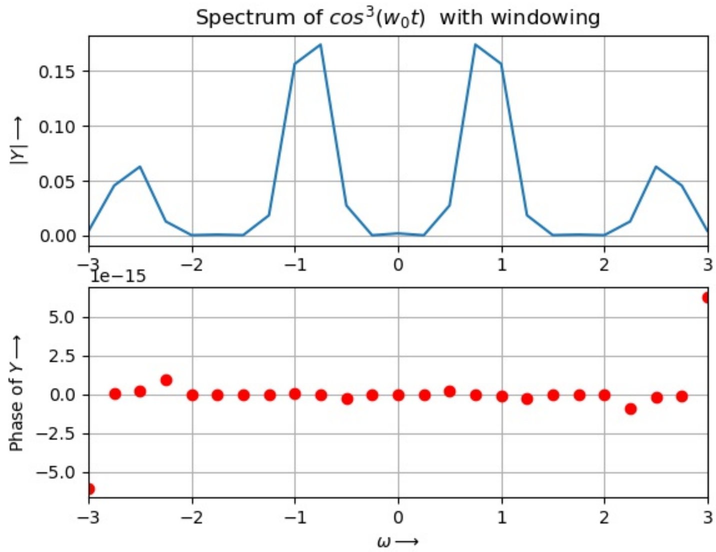
3 Assignment Questions

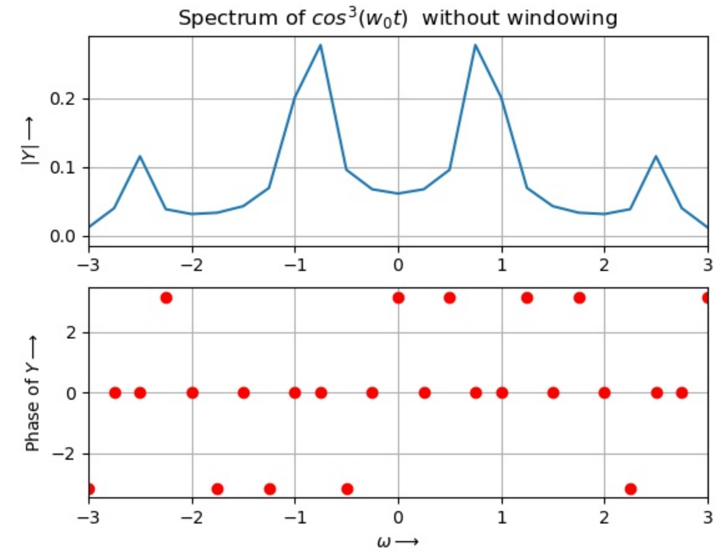
3.1 General function for finding and plotting FFT

```
# General function for finding spectrum and plotting it :
def FFT_and_plotting_spectrum(lim,n,f,t_=0,t_lims = False,
    windowing= False,xlim1=10,title1 = r"Spectrum of
    $sin(\sqrt{2}t)$",xlabel1 = r"$\omega\longrightarrow$",ylabel1=
    r"$|Y|\longrightarrow$", ylabel2 = r"Phase of
    $Y\longrightarrow$", display = True,savename=None):
    if(t_lims):
        t = t_
    else:
        t=linspace(-lim,lim,n+1)[:-1]
    dt=t[1]-t[0];
    fmax=1/dt
    y = f(t)
    if (windowing):
        m=np.arange(n)
        wnd=fftshift(0.54+0.46*np.cos(2*np.pi*m/n))
        y = y*wnd
    y[0]=0
    y=fftshift(y)
    Y=fftshift(fft(y))/float(n)
    w=linspace(-np.pi*fmax,np.pi*fmax,n+1)[:-1]

    mag = abs(Y)
    ph = np.angle(Y)

    if (display):

        figure()
        subplot(2,1,1)
        plot(w,mag)
        xlim([-xlim1,xlim1])
        ylabel(ylabel1)
        title(title1)
        grid(True)
        subplot(2,1,2)
        ph[np.where(mag<3e-3)] = 0
        plot(w,ph,'ro')
        xlim([-xlim1,xlim1])
        ylabel(ylabel2)
        xlabel(xlabel1)
        grid(True)
        savefig(savename)

    return w,Y
```

3.2 Spectrum of $cos^3(\omega_0 t)$

We plot the magnitude and phase spectra of $f(t) = cos^3(\omega_0 t)$ with $\omega_0 = 0.86$ with and without Hamming window. We expect peaks at $\omega_0 = \pm 0.86$ and $\omega_0 = \pm 2.58$.





```
# Question 2
a,b = FFT_and_plotting_spectrum(4*np.pi,64*4,cos3,xlim1=
    3,windowing=False, title1 = r"Spectrum of $cos^3(w_0t)$ without
```

```
    windowing", display = True,savename='Spectrum of cos^3(wt)
    without windowing.jpg')
a,b = FFT_and_plotting_spectrum(4*np.pi,64*4,cos3,xlim1=
    3,windowing=True, title1 = r"Spectrum of $cos^3(w_0t)$ with
    windowing",display = True,savename='Spectrum of cos^3(wt) with
    windowing.jpg')
```

3.3 $f(t) = cos(\omega_0 t + \delta)$

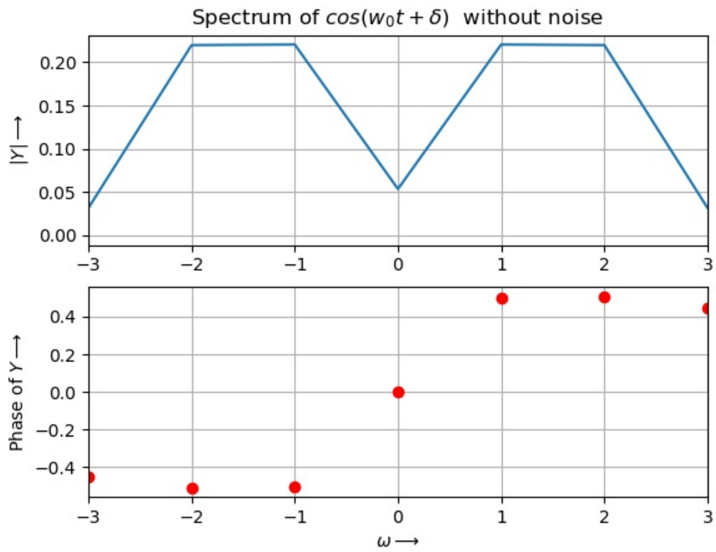
We plot the spectrum of $cos(\omega_0 t + \delta)$ and estimate ω_0 and δ by weighted mean. Estimation of ω_0 is done with respect to the energy present in the frequency components. True values are $\omega_0 = 1.5$ and $\delta = 0.5$.

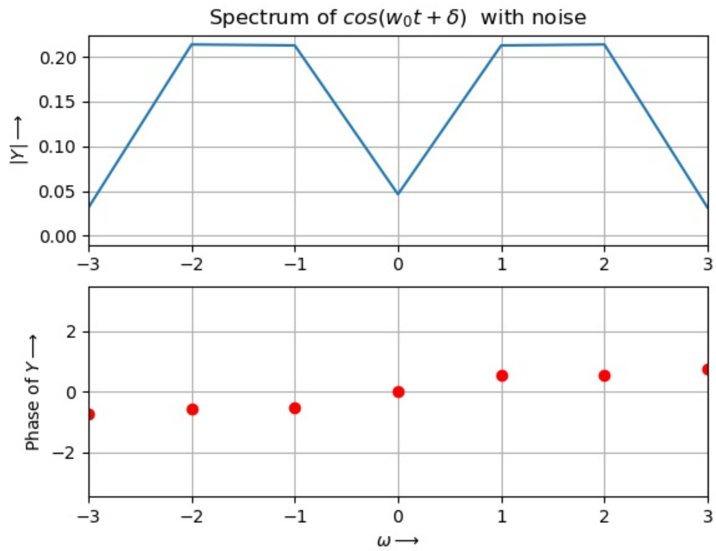
```
# Question 3
#FFT_and_plotting_spectrum of cos(wt+delta) windowed windowing to
    estimate w, delta
w,Y = FFT_and_plotting_spectrum(np.pi,128,cosine,xlim1=
    3,windowing=True, title1 = r"Spectrum of $cos(w_0t + \delta)$
    without noise",display = True,savename='cos(wt+d) without
    noise.jpg')
print("Without noise : ")

def estimate_omega_delta(w,Y):
    # by finding the weighted average of all w>0, we find w0. Delta
        is found by calculating the phase at w closest to w0.
    ii = where(w>=0)
    w_from_spectrum = sum(abs(Y[ii])**2*w[ii])/sum(abs(Y[ii])**2)
    i = abs(w-w_from_spectrum).argmin()
    delta = angle(Y[i])
    print("Value of w0 without noise from the spectrum:
        ",w_from_spectrum)
    print("Value of delta without noise from the spectrum: ",delta)

estimate_omega_delta(w,Y)
```

3.4 Adding Gaussian Noise in the cosine

After adding random Gaussian Noise to this cosine, we observe the spectrum and also compare the estimated values of ω_0 and δ.

```
Without noise :
Value of w0 without noise from the spectrum: 1.4730276250507859
Value of delta without noise from the spectrum: 0.5018760117245951
```

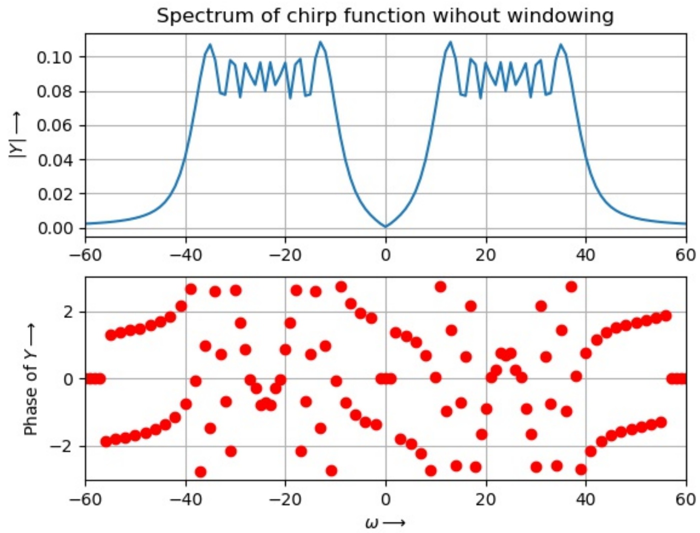




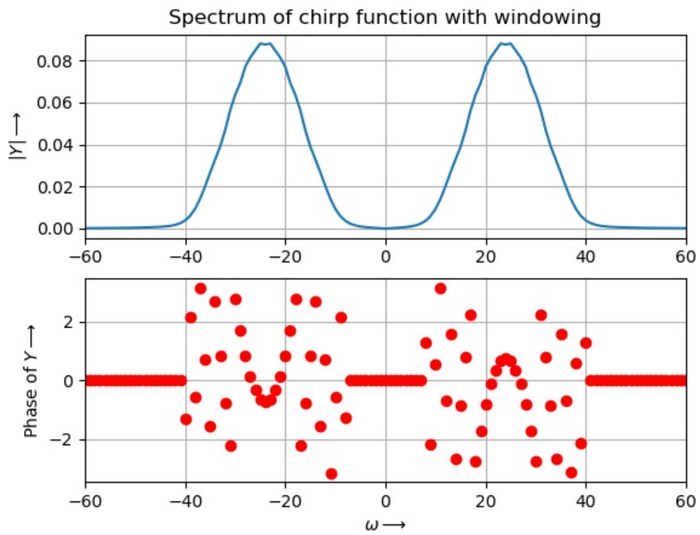

```
With added Gaussian Noise :
Value of w0 without noise from the spectrum: 2.0780679522276984
Value of delta without noise from the spectrum: 0.5457299182257701
```

3.5 FM Modulated Signal (Chirp Signal)

The Chirp signal is one such that it's frequency is proportional to time, a standard feature of FM Modulated signals. In particular, we consider $f(t) = cos(24t + \frac{8}{\pi}t^2)$. Without windowing, the spectrum appears like this :



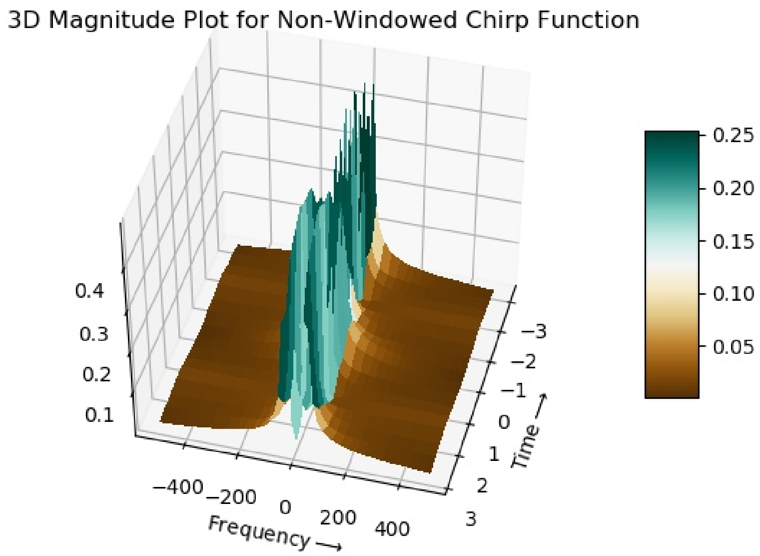
With windowing, the spectrum appears like this :

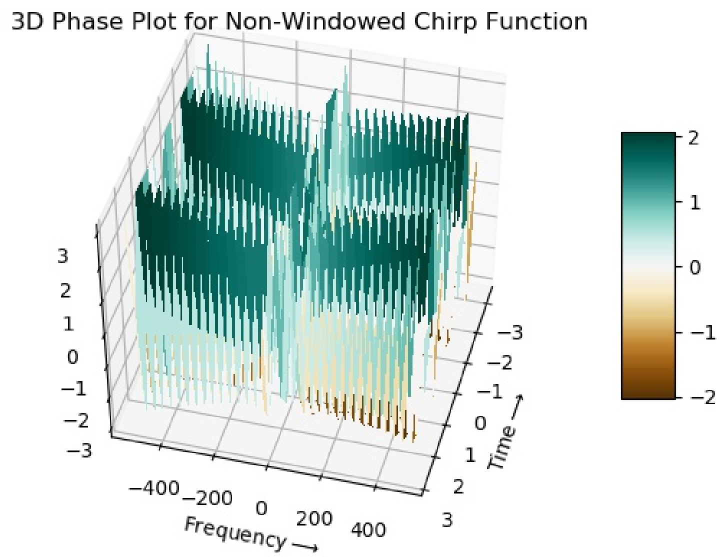


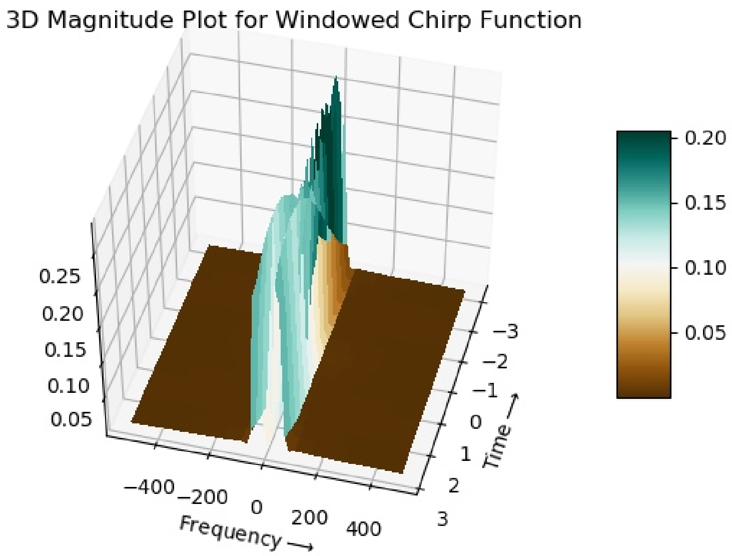
3.6 3D Plot

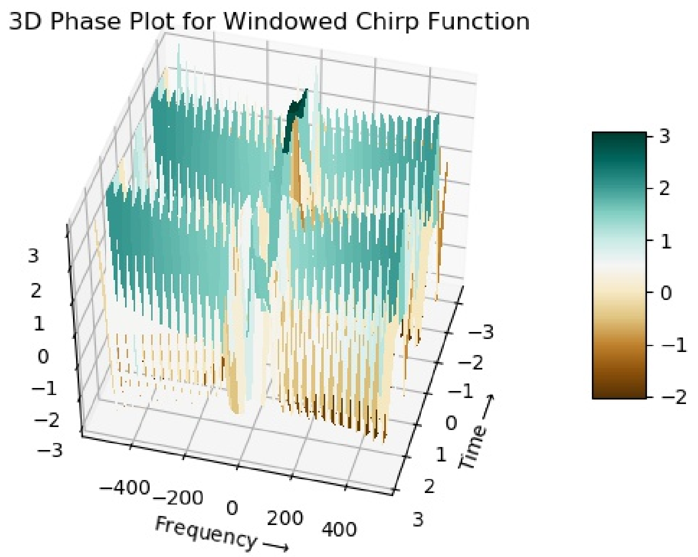
```
#  Question 6
t=np.linspace(-np.pi,np.pi,1025);t=t[:-1]
t_arrays=np.split(t,16)

Y1_mags=np.zeros((16,64))
Y1_angles=np.zeros((16,64))
Y2_mags=np.zeros((16,64))
Y2_angles=np.zeros((16,64))

for i in range(len(t_arrays)):
    w1,Y1 = FFT_and_plotting_spectrum(lim = 10,t_ =
        t_arrays[i],t_lims=True,n = 64,f = chirp,xlim1=
        60,windowing=False, title1 = r"Spectrum of chirp
        function",display = False)
    Y1_mags[i] = abs(Y1)
    Y1_angles[i] = np.angle(Y1)

for i in range(len(t_arrays)):
    w2,Y2 = FFT_and_plotting_spectrum(lim = 10,t_ =
        t_arrays[i],t_lims=True,n = 64,f = chirp,xlim1=
        60,windowing=True, title1 = r"Spectrum of chirp
        function",display = False)
    Y2_mags[i] = abs(Y2)
    Y2_angles[i] = np.angle(Y2)


# 3D Plot of frequency, magnitude and time for non-windowed Chirp
    function :
fig = figure()
ax = fig.add_subplot(111, projection='3d')
ax.view_init(45,15)
t=np.linspace(-np.pi,np.pi,1025);t=t[:-1]
fmax = 1/(t[1]-t[0])
t=t[::64]
w1=np.linspace(-fmax*np.pi,fmax*np.pi,64+1);w1=w1[:-1]
t,w1=np.meshgrid(t,w1)

surf=ax.plot_surface(t,w1,Y1_mags.T,cmap=cm.BrBG,linewidth=0,
    antialiased=False)
fig.colorbar(surf, shrink=0.5, aspect=5)
ylabel("Frequency$\longrightarrow$")
xlabel("Time$\longrightarrow$")
title("3D Magnitude Plot for Non-Windowed Chirp Function")
savefig("Magnitude 3D plot non-windowed chirp.jpg")

fig = figure()
ax = fig.add_subplot(111, projection='3d')
```

```
ax.view_init(45,15)
surf=ax.plot_surface(t,w1,Y1_angles.T,cmap=cm.BrBG,linewidth=0,
    antialiased=False)
fig.colorbar(surf, shrink=0.5, aspect=5)
ylabel("Frequency$\longrightarrow$")
xlabel("Time$\longrightarrow$")
title("3D Phase Plot for Non-Windowed Chirp Function")
savefig("Phase 3D plot non-windowed chirp.jpg")


# 3D Plot of frequency, magnitude and time for Windowed Chirp
    function :
fig = figure()
ax = fig.add_subplot(111, projection='3d')
ax.view_init(45,15)
t=np.linspace(-np.pi,np.pi,1025);t=t[:-1]
fmax = 1/(t[1]-t[0])
t=t[::64]
w2=np.linspace(-fmax*np.pi,fmax*np.pi,64+1);w2=w2[:-1]
t,w2=np.meshgrid(t,w2)

surf=ax.plot_surface(t,w2,Y2_mags.T,cmap=cm.BrBG,linewidth=0,
    antialiased=False)
fig.colorbar(surf, shrink=0.5, aspect=5)
ylabel("Frequency$\longrightarrow$")
xlabel("Time$\longrightarrow$")
title("3D Magnitude Plot for Windowed Chirp Function")
savefig("Magnitude 3D plot windowed chirp.jpg")

fig = figure()
ax = fig.add_subplot(111, projection='3d')
ax.view_init(45,15)
surf=ax.plot_surface(t,w2,Y2_angles.T,cmap=cm.BrBG,linewidth=0,
    antialiased=False)
fig.colorbar(surf, shrink=0.5, aspect=5)
ylabel("Frequency$\longrightarrow$")
xlabel("Time$\longrightarrow$")
title("3D Phase Plot for Windowed Chirp Function")
savefig("Phase 3D plot windowed chirp.jpg")
```


3D Magnitude Plot for Non-Windowed Chirp Function
0.25
0.20
0.15
0.10
0.05
0.4
0.3
0.2
0.1
−400
−200
0
200
400
Frequency ⟶
−3
−2
−1
0
1
2
3
Time ⟶


3D Phase Plot for Non-Windowed Chirp Function
2
1
0
−1
−2
3
2
1
0
−1
−2
−3
−400
−200
0
200
400
Frequency ⟶
−3
−2
−1
0
1
2
3
Time ⟶





4 Observations

- Using a Hamming Window made the spectra more sharper and accurate to the expected results.

- In the example of $sin(\sqrt{2}t)$, we saw that after windowing the peaks became sharper and thinner, implying less energy is possessed by the frequencies which do not belong to the signal.

- The same can be observed for $cos^3(\omega_0 t)$; after windowing, the energy of all other frequencies reduced significantly.

- We can estimate parameters of original function with help of sampled DFT.

- The anomalies due to Gibbs Phenomenon are mitigated by the help of Hamming Window because it smoothens the gap due to multiplication by a factor of less than 1.