Assignment 9: Spectra of Non-Periodic Signals

J.Phani Jayanth - EE19B026

May 22, 2021

Abstract

In this weeks assignment, we look into extracting and plotting the Discrete Fourier Transform (DFT) Spectra of non-periodic functions, in the interval of ($-\pi$,π), using numpy.fft.fft( )function. We also observe and interpret these functions and try to obtain their accurate spectra. For this, we multiply the function by Hamming Window, to make it periodic in ($-\pi$,π).

Introduction

We begin by looking into the function $sin(\sqrt{2}t)$. Initially, we obtain the DFT and plot the spectrum of $sin(\sqrt{2}t)$, by considering 64 samples over the range $(0, 2\pi)$. We have used the function *fftshift()* for obtaining the DFT.

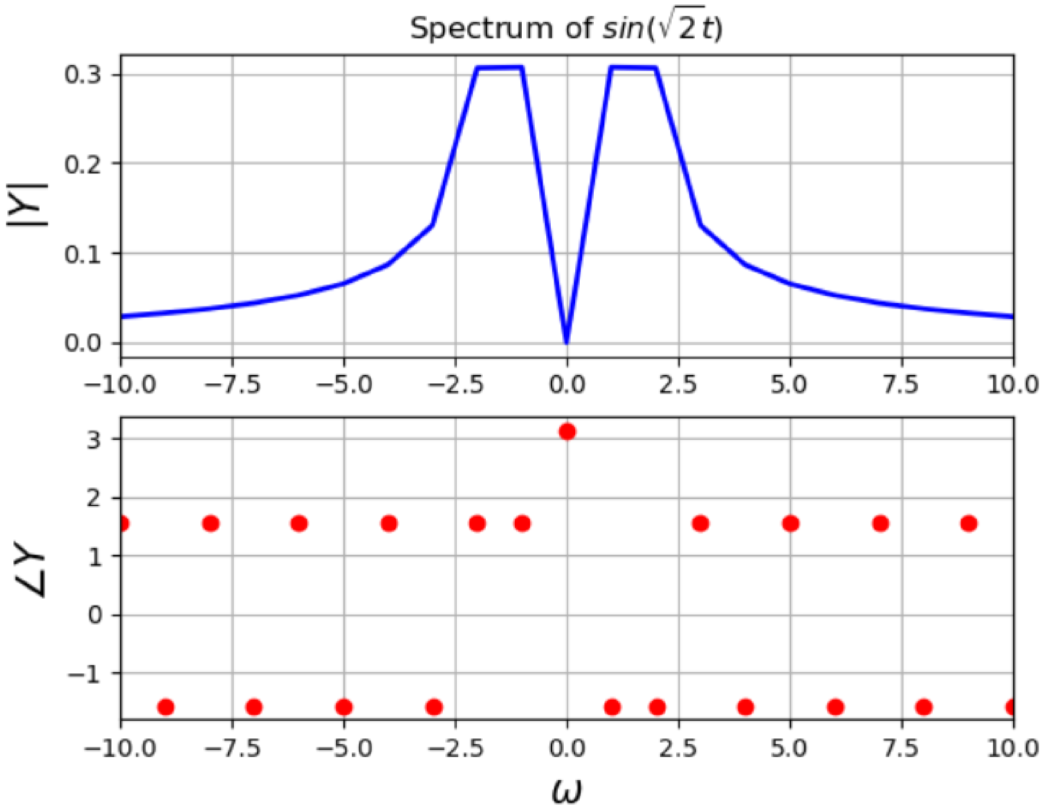


Figure 1: Spectrum of $sin(\sqrt{2}t)$

We observe that the peaks are not at the expected frequencies. We got two peaks each with two values and a gradually decaying magnitude. The phase plot obtained is correct though. This happened because the part of the signal $sin(\sqrt{2}t)$ in the range of $(-\pi, \pi)$ is not periodically repeated. We plot this function for several periods to observe this, as shown in Figure 2.

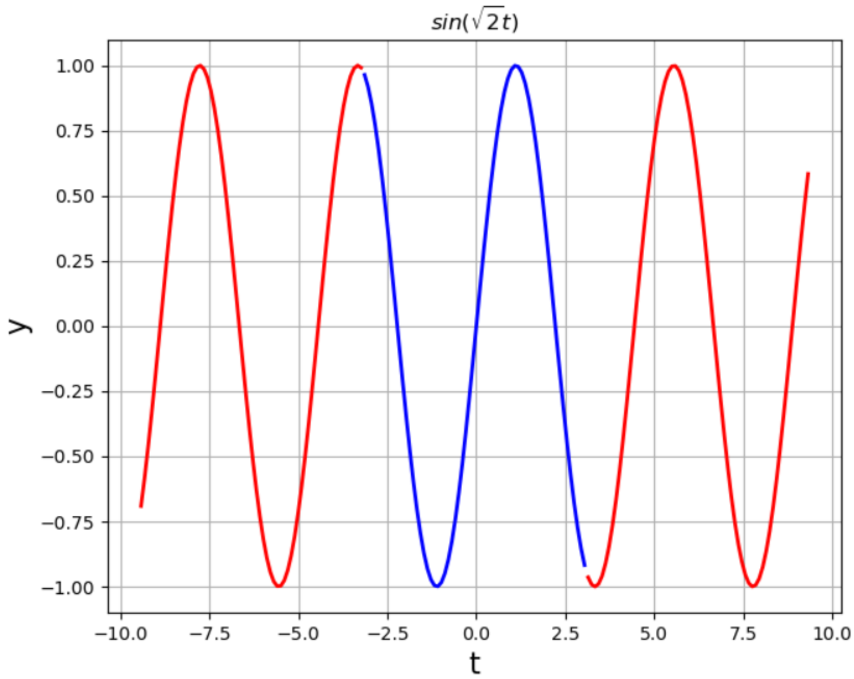


Figure 2: Time Domain Plot of $sin(\sqrt{2}t)$

Therefore, the DFT is not analysing the function $sin(\sqrt{2}t)$, but is rather analysing the function which is a periodic repetition of the blue part highlighted in the above figure.

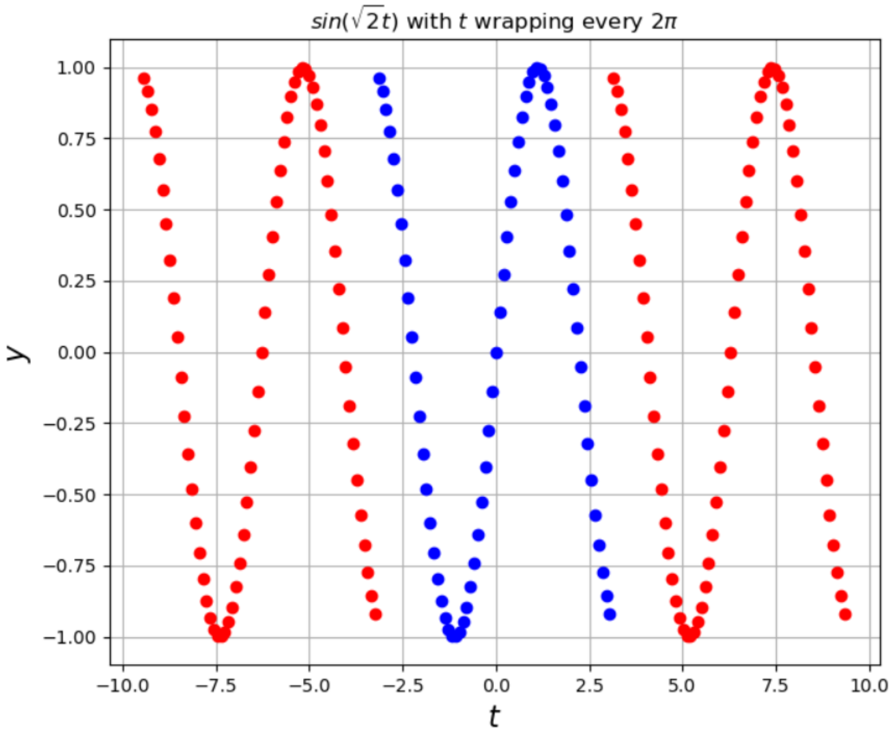


Figure 3: Periodic Extension of $sin(\sqrt{2}t)$ in $(-\pi, \pi)$ range

The code for this is as follows:

```
from pylab import *
from mpl_toolkits.mplot3d import Axes3D

# Spectrum of sin(sqrt(2)t):
t=linspace(-pi,pi,65);t=t[:-1]
dt=t[1]-t[0];fmax=1/dt
y=sin(sqrt(2)*t)
y[0]=0
y=fftshift(y)
Y=fftshift(fft(y))/64.0
w=linspace(-pi*fmax,pi*fmax,65);w=w[:-1]
figure()
subplot(2,1,1)
plot(w,abs(Y),'b',lw=2)
xlim([-10,10])
ylabel(r"$|Y|$",size=16)
title(r"Spectrum of $sin(\sqrt{2}t)$")
grid(True)
subplot(2,1,2)
plot(w,angle(Y),'ro',lw=2)
xlim([-10,10])
ylabel('\u2220$Y$',size=16)
xlabel('$\u03C9$',size=16)
grid(True)
show()

# Plotting sin(sqrt(2)t) in Time Domain:
t1=linspace(-pi,pi,65);t1=t1[:-1]
t2=linspace(-3*pi,-pi,65);t2=t2[:-1]
t3=linspace(pi,3*pi,65);t3=t3[:-1]
figure()
plot(t1,sin(sqrt(2)*t1),'b',lw=2)
plot(t2,sin(sqrt(2)*t2),'r',lw=2)
plot(t3,sin(sqrt(2)*t3),'r',lw=2)
ylabel("y",size=16)
xlabel("t",size=16)
title(r"$sin(\sqrt{2}t)$")
grid(True)
show()

# Periodic repetition of sin(sqrt(2)t) from -pi to +pi:
t1=linspace(-pi,pi,65);t1=t1[:-1]
t2=linspace(-3*pi,-pi,65);t2=t2[:-1]
t3=linspace(pi,3*pi,65);t3=t3[:-1]
y=sin(sqrt(2)*t1)
figure()
plot(t1,y,'bo',lw=2)
plot(t2,y,'ro',lw=2)
plot(t3,y,'ro',lw=2)
ylabel(r"$y$",size=16)
xlabel(r"$t$",size=16)
title(r"$sin(\sqrt{2}t)$ with $t$ wrapping every 2$\pi$ ")
grid(True)
show()
```

Spectrum of a Digital Ramp

The ramp function is also discontinuous in time, i.e., it is a non-periodic function and hence it is similar to the above discussed sinusoid. The DFT samples decay as $\frac{1}{\omega}$. SWe plot the spectrum of the digital ramp as shown below:

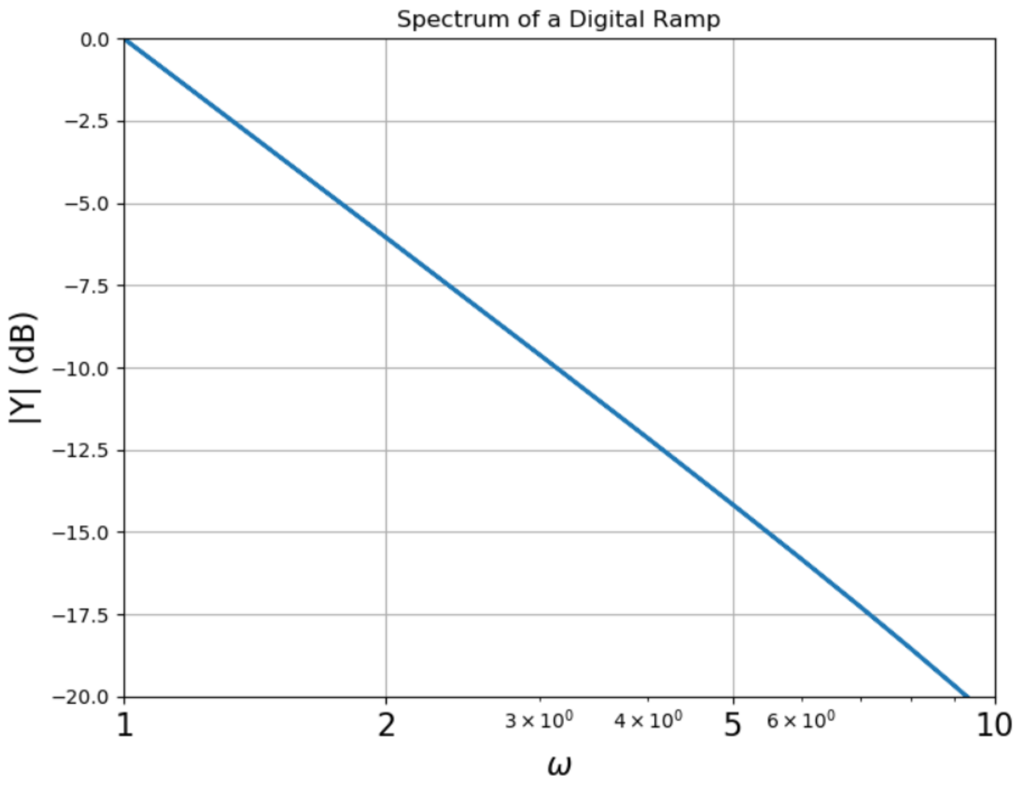


Figure 4: Spectrum of Digital Ramp - Decaying as $\frac{1}{\omega}$

The code is as follows:

```
# Spectrum of a digital ramp:
t=linspace(-pi,pi,65);t=t[:-1]
dt=t[1]-t[0];fmax=1/dt
y=t
y[0]=0 # The sample corresponding to -tmax should be set zero
y=fftshift(y) # Start with y(0)
Y=fftshift(fft(y))/64.0
w=linspace(-pi*fmax,pi*fmax,65);w=w[:-1]
figure()
semilogx(abs(w),20*log10(abs(Y)),lw=2)
xlim([1,10])
ylim([-20,0])
xticks([1,2,5,10],["1","2","5","10"],size=16)
ylabel("|Y| (dB)",size=16)
title("Spectrum of a Digital Ramp")
xlabel("$\u03C9$",size=16)
grid(True)
show()
```

Windowing: Hamming Window

We observe that the spikes happen at the end of the periodic interval. So, we damp the function there, i.e., we multiply our function sequence *f[n]* by a window sequence *w[n]*.

$$g(n) = f(n)w(n)$$

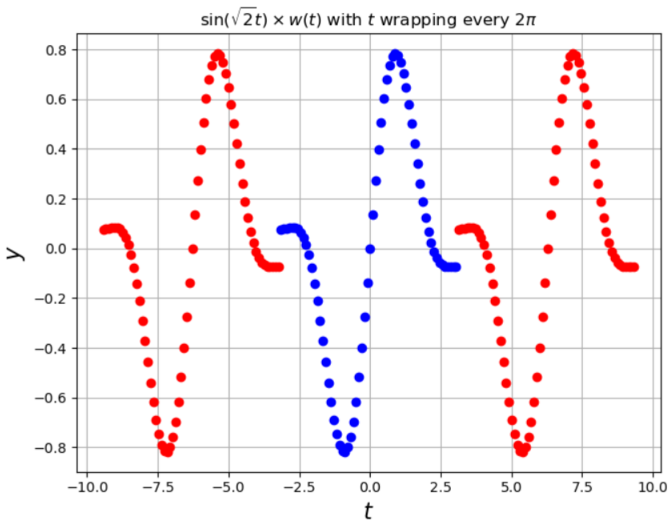
The new spectrum is obtained by convolving the two fourier transforms:

$$G_k = \sum_{n=0}^{N-1} F_n W_{k-n}$$

The window we will use is called the *Hamming window* and is given by:

$$w[n] = 0.54 + 0.46cos(\frac{2\pi n}{N-1})$$

We observe the plot of $sin(\sqrt{2}t)w(t)$ with t wrapping every 2π. This is shown in the below figure:




```python
# sin(sqrt(2)t) with Hamming window in Time domain:
t1=linspace(-pi,pi,65);t1=t1[:-1]
t2=linspace(-3*pi,-pi,65);t2=t2[:-1]
t3=linspace(pi,3*pi,65);t3=t3[:-1]
n=arange(64)
wnd=fftshift(0.54+0.46*cos(2*pi*n/63))
y=sin(sqrt(2)*t1)*wnd
figure()
plot(t1,y,'bo',lw=2)
plot(t2,y,'ro',lw=2)
plot(t3,y,'ro',lw=2)
ylabel(r"$y$",size=16)
xlabel(r"$t$",size=16)
title(r"$\sin(\sqrt{2}t)\times w(t)$ with $t$ wrapping every 2$\pi$ ")
grid(True)
show()
```

We now plot the spectrum of $sin(\sqrt{2}t)$ after windowing. We obtain the spectrum as follows:

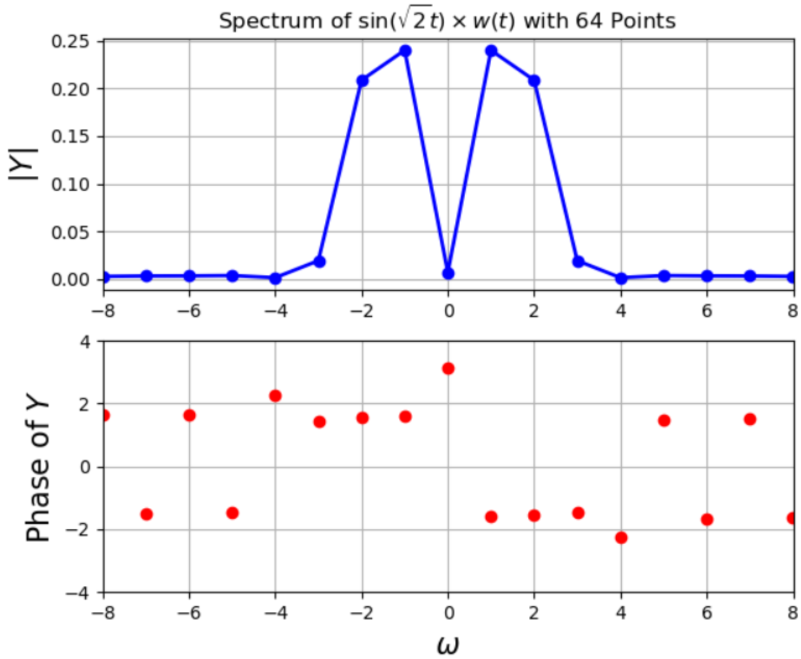


Figure 5: Spectrum of $sin(\sqrt{2}t)$ with Hamming Window - 64 Samples

We now increase the number of samples by 4 times. We observe that we obtain an improved and more accurate spectrum as shown in the figure below.

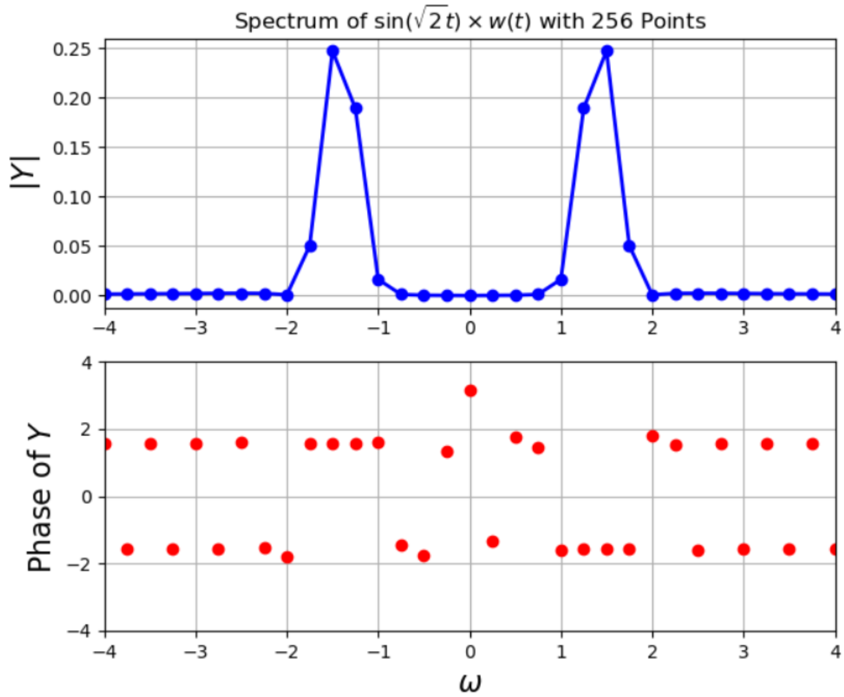


Figure 6: Spectrum of $sin(\sqrt{2}t)$ with Hamming Window - 256 Samples

Code: Function for Plotting Spectra with/without Hamming Window

```
def HammingWin(index,k,N,xlimit,heading,Hamming=True,EST=False):
    t = linspace(-k*pi,k*pi,N+1);t=t[:-1]
    dt=t[1]-t[0];fmax=1/dt
    if Hamming == True:
        n=arange(N)
        wnd=fftshift(0.54+0.46*cos(2*pi*n/(N-1)))
        y=func(index,t,N)*wnd
    elif Hamming == False:
        y=func(index,t,N)
    y[0]=0
    y=fftshift(y)
    Y=fftshift(fft(y))/N
    w=linspace(-pi*fmax,pi*fmax,N+1);w=w[:-1]
    figure()
    subplot(2,1,1)
    plot(w,abs(Y),'b',w,abs(Y),'bo',lw=2)
    xlim([-xlimit,xlimit])
    ylabel(r"$|Y|$",size=16)
    title(heading)
    grid(True)
    subplot(2,1,2)
    plot(w,angle(Y),'ro',lw=2)
    xlim([-xlimit,xlimit])
    ylim([-4,4])
    ylabel(r"Phase of $Y$",size=16)
    xlabel(r"$\omega$",size=16)
    grid(True)
    show()
```

Spectrum of $cos^3(0.86t)$

We plot the spectrum of $cos^3(0.86t)$ with and without the Hamming window as shown in Figure 7 and Figure 8 respectively. We observe that the spectrum without the Hamming window has broader peaks, which shows that the peak energy is spread to the nearby frequencies too.

When we multiply the function with the Hamming window and then obtain the Discrete Fourier Transform spectra, we obtain a more accurate spectrum than the previous one. By multiplying with the Hamming window, we attenuate the frequencies around the peak which leads us to obtaining a spectrum with higher accuracy.

```
# Without Hamming Window
HammingWin(1,4,256,5,heading=r"Spectrum of $\cos^3(0.86t)$ without Hamming Window",Hamming=False)

# With Hamming Window
HammingWin(1,4,256,5,heading=r"Spectrum of $\cos^3(0.86t)$ with Hamming Window")
```

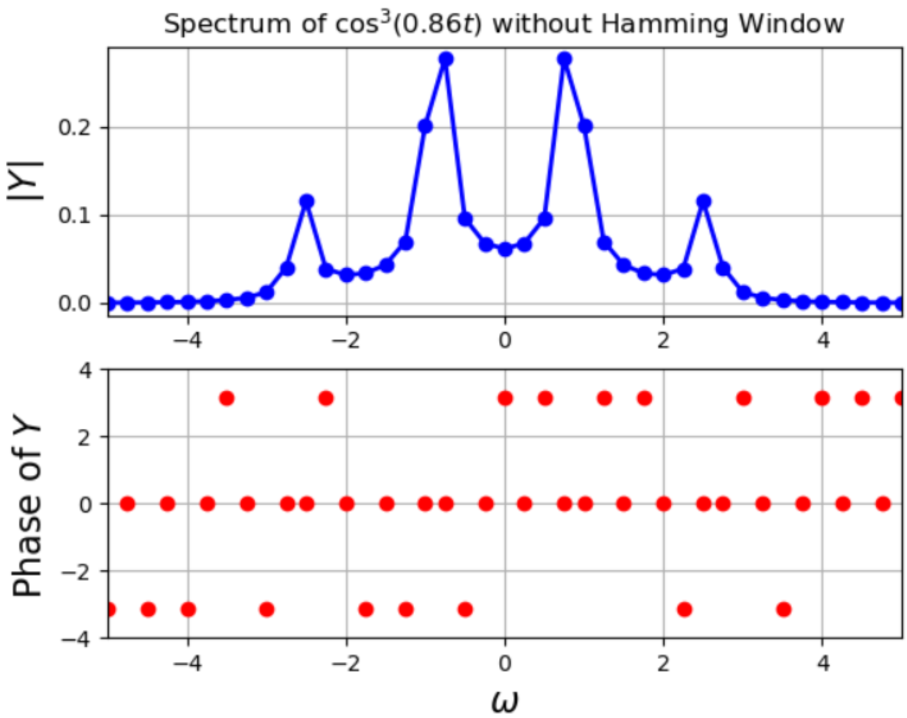


Figure 7: Spectrum of $cos^3(0.86t)$ without Hamming Window

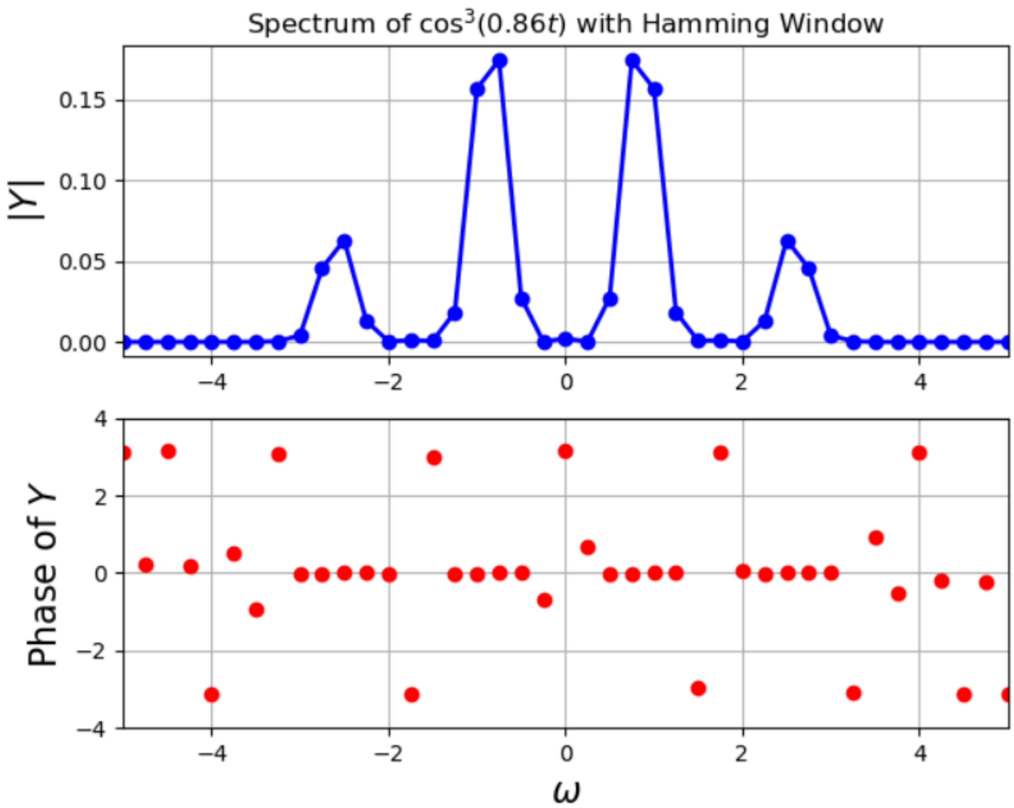


Figure 8: Spectrum of $cos^3(0.86t)$ with Hamming Window

Extracting and Plotting the Spectrum of $cos(\omega t + \delta)$

We obtain the DFT of the function $cos(\omega t + \delta)$ for $\omega = 0.8$ and $\delta = \pi$. We plot the spectrum of this function with the Hamming window as shown below.

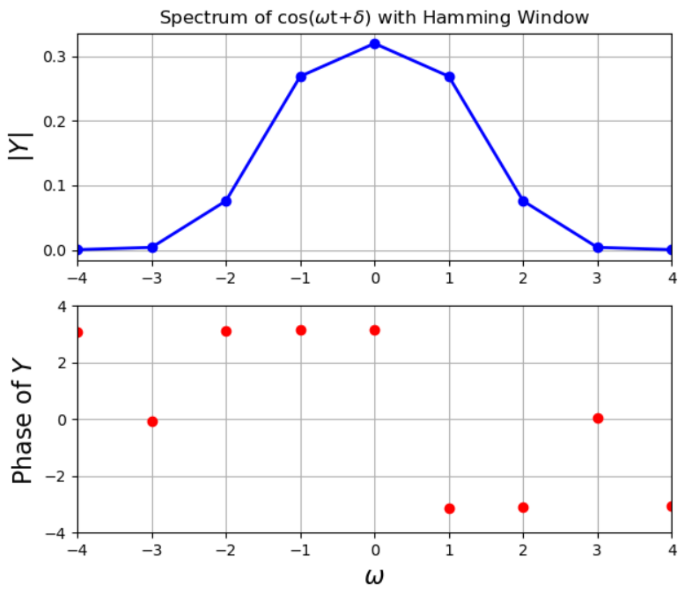


Figure 9: Spectrum of $cos(\omega t + \delta)$ with Hamming Window

We also obtain the spectrum of $cos(\omega t + \delta)$ with added Gaussian noise and plot the spectrum as shown below.

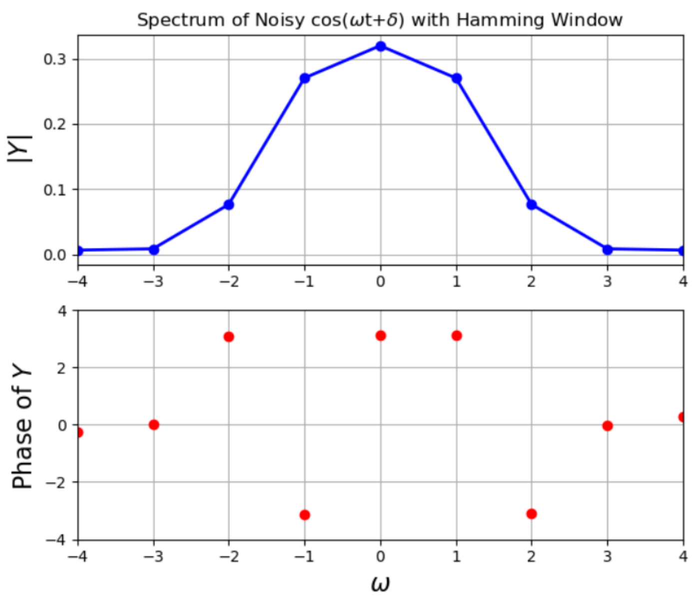


Figure 10: Spectrum of Noisy $cos(\omega t + \delta)$ with Hamming Window

Estimation of ω and δ

For estimating the values of ω and δ from the extracted Discrete Time Fourier Transform (DFT), we find the weighted average of frequencies. The weight of each frequency is taken as the magnitude of DFT at that frequency. We then estimate the value of peak frequency. The suitable power of magnitudes was set as $p = 1.6$. The phase can also be estimated by observing the phase at points where the magnitude peaks (ω_0)

```
omega=0.8
delta=pi

# QUESTION 3: Estimating Omega and Delta:
HammingWin(2,1,128,4,heading=r"Spectrum of cos($\omega$t+$\delta$) with
    Hamming Window",EST=True)

# QUESTION 4: Estimation with added Noise:
HammingWin(3,1,128,4,heading=r"Spectrum of Noisy cos($\omega$t+$\delta$)
    with Hamming Window",EST=True)

# For Estimation:
p=1.6
phase = angle(Y[::-1[argmax(abs(Y[::-1]))]])
w0 = sum(abs(Y**p*w))/sum(abs(Y)**p)
print(f'Estimated value of frequency, with noise : {w0}')
print(f'Estimated value of phase, with noise : {phase}')
```

- Estimated values of ω and δ for the function $cos(\omega t+\delta)$ are: $\omega = 0.75350$ and $\delta = 3.14159$
- Estimated values of ω and δ for the function $cos(\omega t + \delta)$ with added Gaussian noise are: $\omega = 2.88207$ (varies) and $\delta = 3.14159$

Spectrum of the Chirped Signal

The Chirped Signal is given as follows:

$$cos(16(1.5 + \frac{t}{2\pi})t)$$

The frequency of the chirped signal varies from 16 rad/sec at $-\pi$ to 32 rad/sec as we move towards π. We plot and observe the DFT Spectrum of the chirped signal with and without the Hamming Window.

```
# QUESTION 5: DFT of Chirped Signal:
## Without Windowing:
HammingWin(4,1,1024,50,heading=r"Spectrum of Chirped Signal without Hamming
    window",Hamming=False)

## With Windowing:
HammingWin(4,1,1024,50,heading=r"Spectrum of Chirped Signal with Hamming
    window")
```

The Plots obtained are as follows:

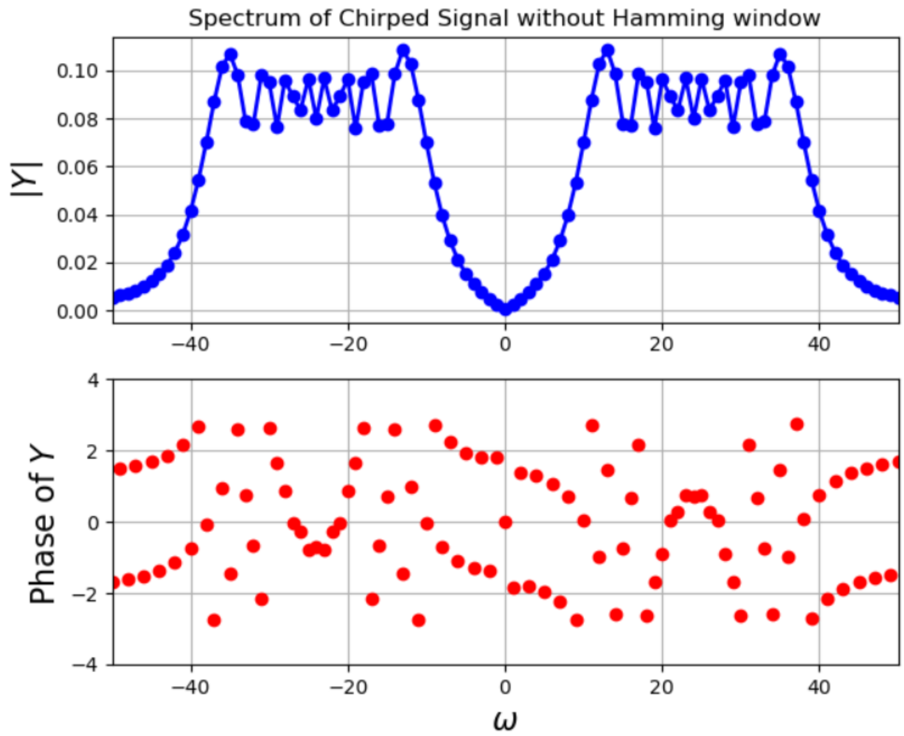


Figure 11: Spectrum of Chirped Signal without Hamming Window

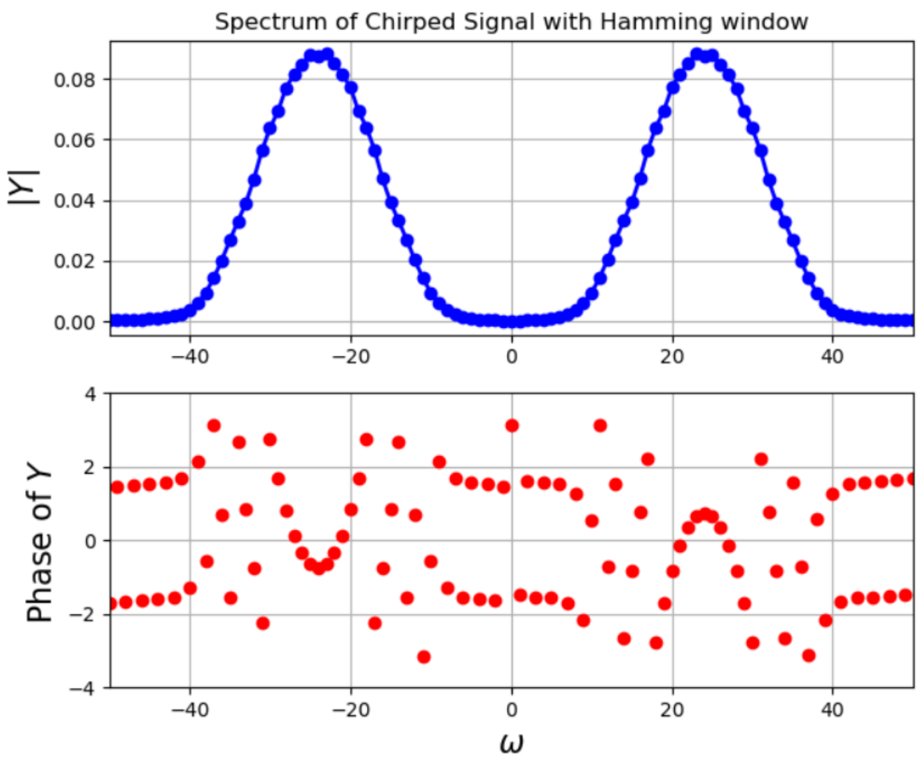


Figure 12: Spectrum of Chirped Signal with Hamming Window

Surface Plot: Frequency variation with Time - Chirped Signal

We plot a Surface Plot to observe the frequency-time variation of the DFT magnitude. As mentioned before, the frequency of this signal varies from 16 rad/sec to 32 rad/sec as time varies. The plots are obtained for the chirped signal with and without windowing, as shown in Figure 13 and Figure 14 respectively.

The code is as follows:

```
# QUESTION 6: Surface Plot - Chirped Signal:
## Without Hamming Window:
t=linspace(-1*pi,1*pi,1025);t=t[:-1]
dt=t[1]-t[0];fmax=1/dt
y=cos(16*(1.5+t/(2*pi))*t)
y_=np.zeros((16,64),dtype=complex)
for i in range(16):
  y_[i]=fftshift(fft(fftshift(y[64*i:64*(i+1)])))
w=linspace(-pi*fmax,pi*fmax,1025);w=w[:-1]
n=arange(64)
t1 = np.array(range(16))
t1,n = meshgrid (t1,n)
ax=Axes3D(figure())
surf = ax.plot_surface(t1,n,abs(y_).T,rstride =1,cstride=1,cmap='inferno')
ylabel('\u03C9')
xlabel('t')
title("Surface Plot: Variation of frequency with time - Chirped Signal without Hamming Window")
ax.set_zlabel('|Y|')
show()

## With Hamming Window:
t=linspace(-1*pi,1*pi,1025);t=t[:-1]
dt=t[1]-t[0];fmax=1/dt
n0=arange(1024)
wnd=fftshift(0.54+0.46*cos(2*pi*n0/(1023)))
y=cos(16*(1.5+t/(2*pi))*t)*wnd
y_=np.zeros((16,64),dtype=complex)
for i in range(16):
  y_[i]=fftshift(fft(fftshift(y[64*i:64*(i+1)])))
w=linspace(-pi*fmax,pi*fmax,1025);w=w[:-1]
n=arange(64)
t1 = np.array(range(16))
t1,n = meshgrid (t1,n)
ax=Axes3D(figure())
surf = ax.plot_surface(t1,n,abs(y_).T,rstride =1,cstride=1,cmap='inferno')
ylabel('\u03C9')
xlabel('t')
title("Surface Plot: Variation of frequency with time - Chirped Signal with Hamming Window")
ax.set_zlabel('|Y|')
show()
```

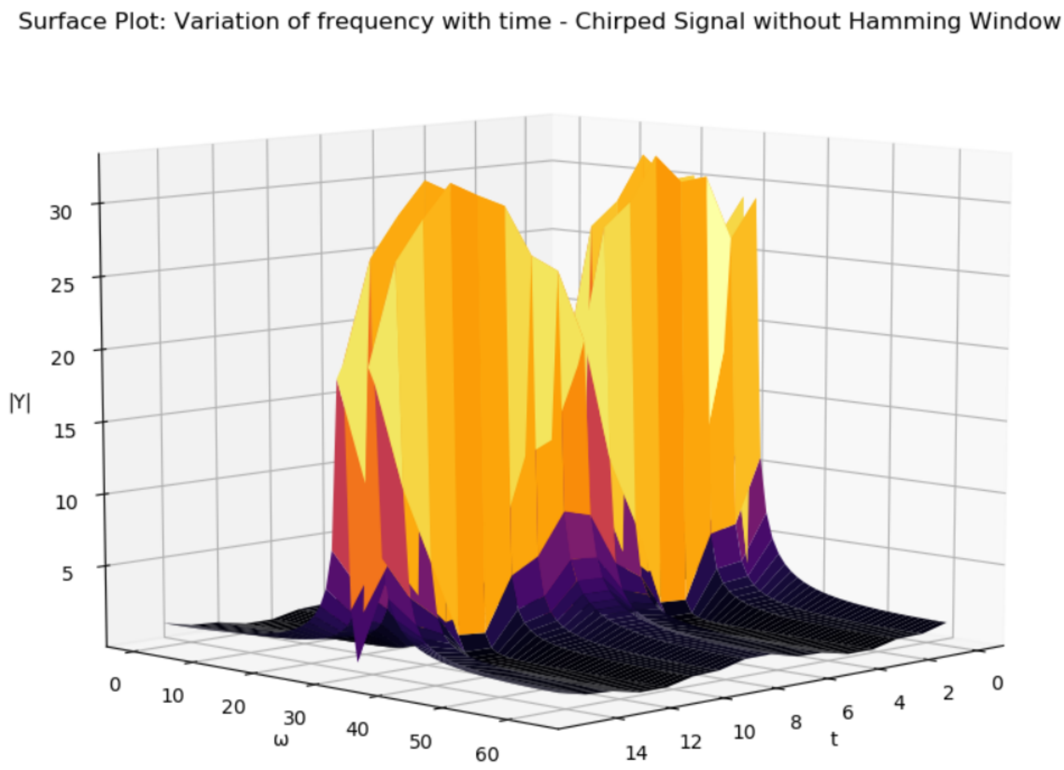


Figure 13: Chirped Signal without Hamming Window: Frequency variation with Time

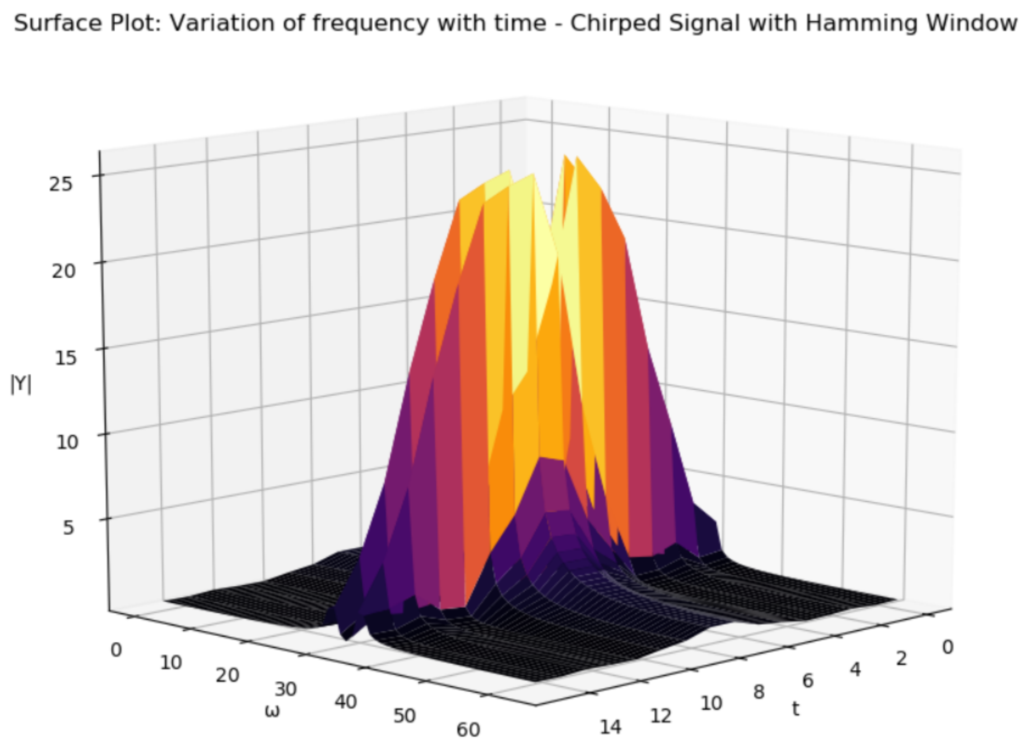


Figure 14: Chirped Signal with Hamming Window: Frequency variation with Time