EE2703: Applied Programming Lab

Assignment # 9

GAGAN VIRENDRA KUMAR, EE19B073

10 MAY, 2021

Contents

1 Introduction

Most signals aren't periodic, and even a periodic one might have an unknown period. So we should be prepared to do Fourier analysis on signals without making the comforting assumption that the signal to analyze repeats at a fixed period N. In most cases we can simply take N samples of the signal and make it periodic; this is essentially what we do in this assignment. [1]

$$Analysis\ Equation\ : X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn} \tag{1}$$

$$Synthesis\ Equation\ : x[n] = \frac{1}{N} \sum_{n=0}^{N-1} X[k] W_N^{-kn} \tag{2}$$

where $W_N = e^{j2\pi/N}$

2 Problem 1

Import the following libraries.

```
import numpy as np
import scipy.signal as sp
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
```

2.1 $f(t) = sin(\sqrt{2}t)$

2.1.1 Time Domain Signal

```
N = 512
t = np.linspace(-np.pi,np.pi,N+1);t = t[:-1]
t1 = np.linspace(-3*np.pi,-np.pi,N+1);t1 = t1[:-1]
t2 = np.linspace(np.pi,3*np.pi,N+1);t2 = t2[:-1]
dt = t[1] - t[0];fmax = 1/dt
y = np.sin(np.sqrt(2)*t)
n = np.arange(N)
wnd = np.fft.fftshift(0.54+0.46*np.cos(2*np.pi*n/(N-1)))
y = y*wnd
plt.figure(figsize=(16,8))
plt.grid()
plt.title(r'$sin(\sqrt{2}t)w(t)$')
plt.ylabel(r'$y\rightarrow$')
plt.xlabel(r'$t\rightarrow$')
plt.plot(t,y,'r')
plt.plot(t1,y,'b')
plt.plot(t2,y,'b')
plt.show()
```

2.1.2 Magnitude Response

```
t = np.linspace(-8*np.pi,8*np.pi,N+1);t = t[:-1]
dt = t[1] - t[0];fmax = 1/dt
y = np.sin(np.sqrt(2)*t)
n = np.arange(N)
wnd = np.fft.fftshift(0.54+0.46*np.cos(2*np.pi*n/(N-1)))
```

[1] http://msp.ucsd.edu/techniques/v0.11/book-html/node171.html

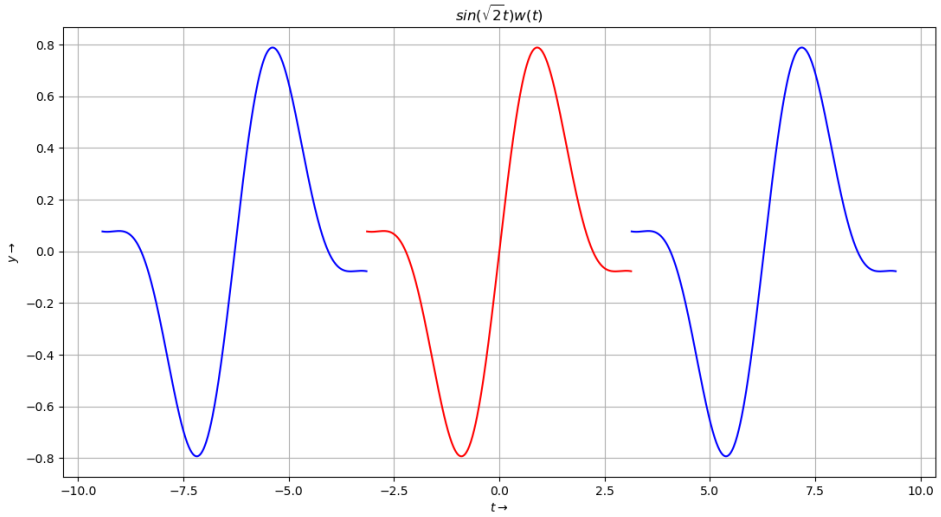

Figure 1: $sin(\sqrt{2}t)$ with windowing

```
y = y*wnd
y[0] = 0
y = np.fft.fftshift(y)
y = np.fft.fftshift(np.fft.fft(y))/N
w = np.linspace(-np.pi*fmax,np.pi*fmax,N+1);w = w[:-1]
plt.figure(figsize=(16,8))
plt.grid()
plt.title(r'$Magnitude\;of\;sin(\sqrt{2}t)$')
plt.ylabel(r'$|Y|\rightarrow$')
plt.xlabel(r'$\omega\rightarrow$')
plt.xlim([-5,5])
plt.stem(w,abs(y),markerfmt='.')
plt.show()
```

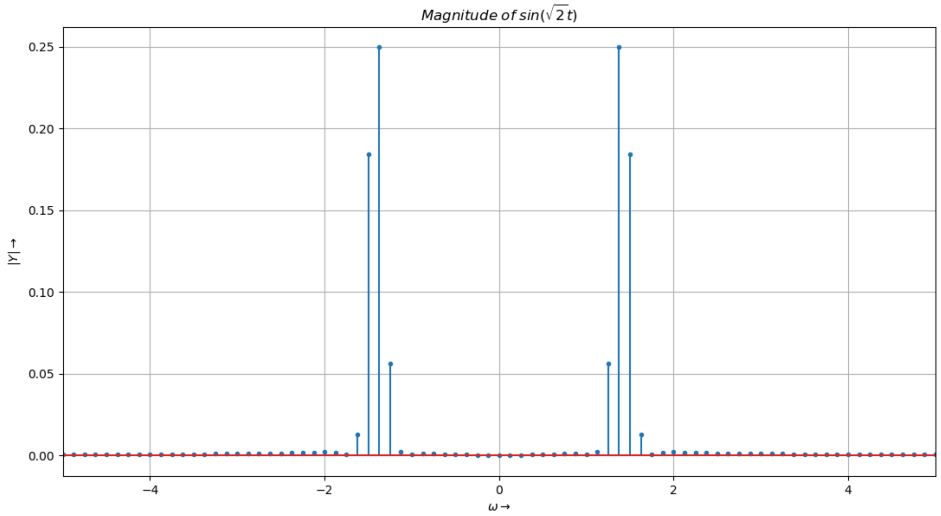

Figure 2: Magnitude of $\sin(\sqrt{2}t)$

3 Problem 2

3.1 $f(t) = cos^3(0.86t)$

3.1.1 Time Domain Signal

```
N = 1024
t = np.linspace(-np.pi,np.pi,N+1);t = t[:-1]
t1 = np.linspace(-3*np.pi,-np.pi,N+1);t1 = t1[:-1]
t2 = np.linspace(np.pi,3*np.pi,N+1);t2 = t2[:-1]
dt = t[1] - t[0];fmax = 1/dt
y = (np.cos(0.86*t))**3
n = np.arange(N)
plt.figure(figsize=(16,8))
plt.grid()
plt.title(r'$cos^3(0.86t)$')
plt.ylabel(r'$y\rightarrow$')
plt.xlabel(r'$t\rightarrow$')
plt.plot(t,y,'r')
plt.plot(t1,y,'b')
plt.plot(t2,y,'b')
plt.show()
```

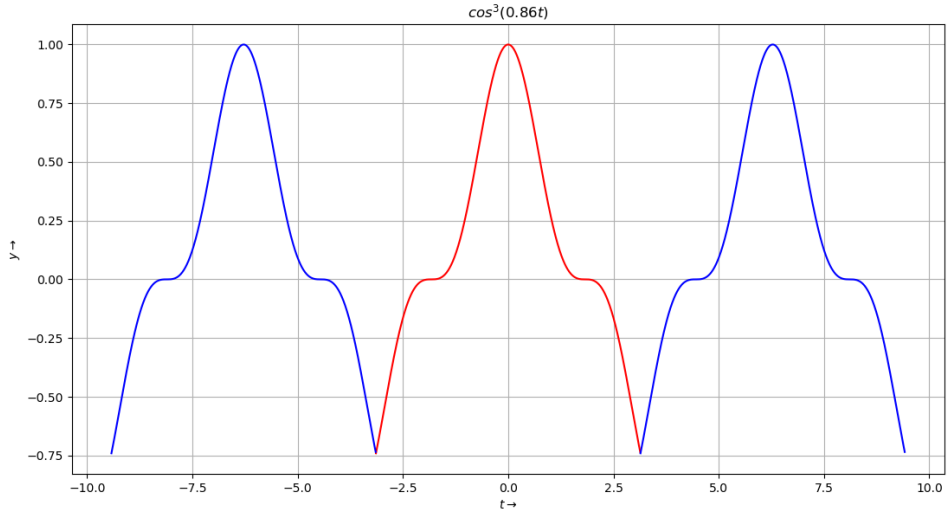

Figure 3: $cos^3(0.86t)$

3.1.2 Magnitude Response

```
t = np.linspace(-32*np.pi,32*np.pi,N+1);t = t[:-1]
dt = t[1] - t[0];fmax = 1/dt
y = (np.cos(0.86*t))**3
n = np.arange(N)
y[0] = 0
y = np.fft.fftshift(y)
y = np.fft.fftshift(np.fft.fft(y))/N
w = np.linspace(-np.pi*fmax,np.pi*fmax,N+1);w = w[:-1]
plt.figure(figsize=(16,8))
plt.grid()
```

```
plt.title(r'$Magnitude\;of\;cos^3(0.86t)\;without\;Hamming\;window$')
plt.ylabel(r'$|Y|\rightarrow$')
plt.xlabel(r'$\omega\rightarrow$')
plt.xlim([-5,5])
plt.stem(w,abs(y),markerfmt='.')
plt.show()
```

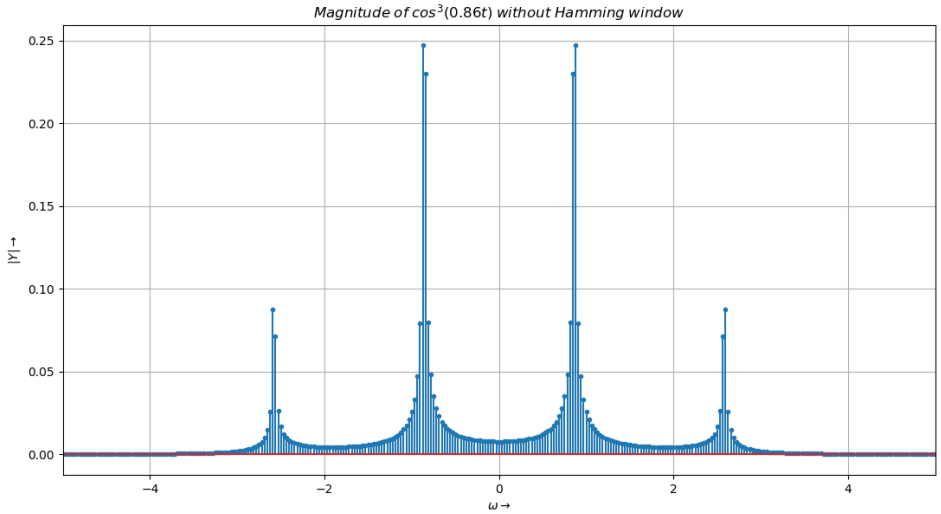

Figure 4: *Magnitude of* $cos^3(0.86t)$ *without Hamming window*

3.2 $f(t) = cos^3(0.86t)w(t)$

3.2.1 Time Domain Signal

```
N = 1024
t = np.linspace(-np.pi,np.pi,N+1);t = t[:-1]
t1 = np.linspace(-3*np.pi,-np.pi,N+1);t1 = t1[:-1]
t2 = np.linspace(np.pi,3*np.pi,N+1);t2 = t2[:-1]
dt = t[1] - t[0];fmax = 1/dt
y = (np.cos(0.86*t))**3
wnd = np.fft.fftshift(0.54+0.46*np.cos(2*np.pi*n/(N-1)))
y = y*wnd
n = np.arange(N)
plt.figure(figsize=(16,8))
plt.grid()
plt.title(r'$cos^3(0.86t)w(t)$')
plt.ylabel(r'$y\rightarrow$')
plt.xlabel(r'$t\rightarrow$')
plt.plot(t,y,'r')
plt.plot(t1,y,'b')
plt.plot(t2,y,'b')
plt.show()
```

3.2.2 Magnitude Response

```
t = np.linspace(-32*np.pi,32*np.pi,N+1);t = t[:-1]
dt = t[1] - t[0];fmax = 1/dt
```

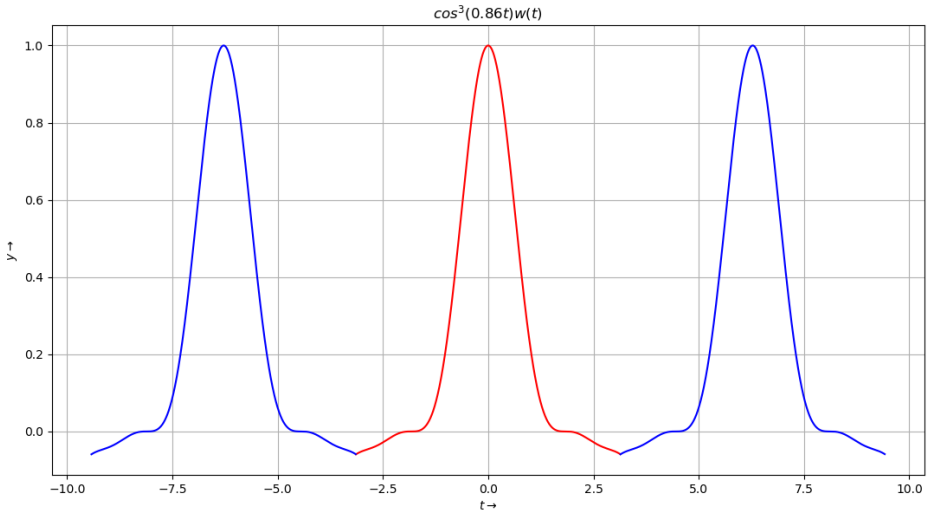

Figure 5: $cos^3(0.86t)w(t)$

```
y = (np.cos(0.86*t))**3
wnd = np.fft.fftshift(0.54+0.46*np.cos(2*np.pi*n/(N-1)))
y = y*wnd
n = np.arange(N)
y[0] = 0
y = np.fft.fftshift(y)
y = np.fft.fftshift(np.fft.fft(y))/N
w = np.linspace(-np.pi*fmax,np.pi*fmax,N+1);w = w[:-1]
plt.figure(figsize=(16,8))
plt.grid()
plt.title(r'$Magnitude\;of\;cos^3(0.86t)\;with\;Hamming\;window$')
plt.ylabel(r'$|Y|\rightarrow$')
plt.xlabel(r'$\omega\rightarrow$')
plt.xlim([-5,5])
plt.stem(w,abs(y),markerfmt='.')
plt.show()
```

3.3 Analysis

The purpose of the Hamming Window is to try to make the signal periodic.We see that as the window is multiplied with the signal the peaks in the frequency domain becomes much sharper.

4 Problem 3

4.1 Time Domain Signal

```
seed_omega = 485
np.random.seed(seed_omega)
omega = np.random.random() + 0.5
seed_phi = 5655
np.random.seed(seed_phi)
phi = np.random.random()

print('Actual value of omega:{}'.format(omega))
```

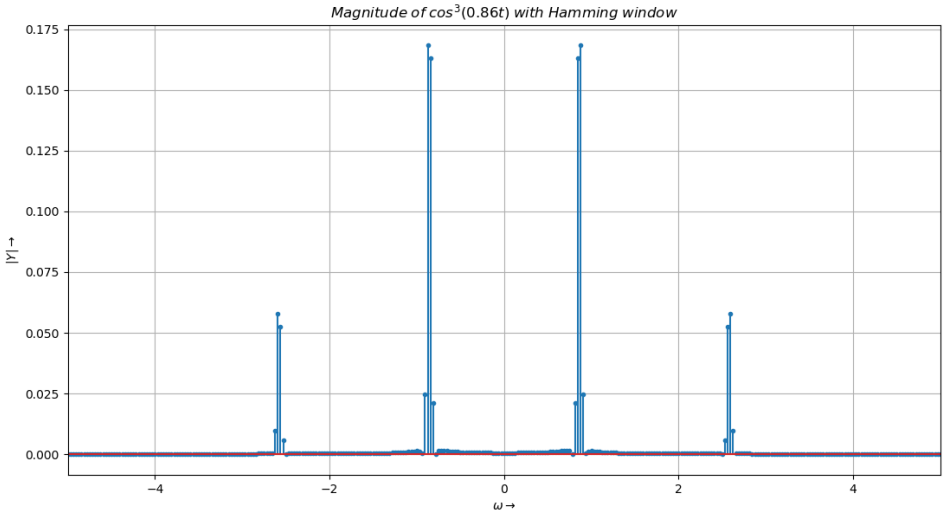

Figure 6: *Magnitude of* $cos^3(0.86t)$ *with Hamming window*

```
print('Actual value of phi:{}'.format(phi))
N_sam = 128
t = np.linspace(-np.pi,np.pi,N_sam+1);t = t[:-1]
t1 = np.linspace(-3*np.pi,-np.pi,N_sam+1);t1 = t1[:-1]
t2 = np.linspace(np.pi,3*np.pi,N_sam+1);t2 = t2[:-1]
dt = t[1] - t[0];fmax = 1/dt
y = np.cos(omega*t + phi)
plt.figure(figsize=(16,8))
plt.grid(True)
plt.title(r'cos($\omega t + \phi$) without Hamming Window')
plt.xlabel(r'$t\rightarrow$')
plt.ylabel(r'$y\rightarrow$')
plt.plot(t,y)
plt.plot(t1,y,'r')
plt.plot(t2,y,'r')
plt.show()

t = np.linspace(-np.pi,np.pi,N_sam+1);t = t[:-1]
n = np.arange(N_sam)
wnd = np.fft.fftshift(0.54+0.46*np.cos(2*np.pi*n/(N_sam-1)))
y = y*wnd
plt.figure(figsize=(16,8))
plt.grid(True)
plt.title(r'cos($\omega t + \phi$) with Hamming Window')
plt.xlabel(r'$t\rightarrow$')
plt.ylabel(r'$y\rightarrow$')
plt.plot(t,y)
plt.plot(t1,y,'r')
plt.plot(t2,y,'r')
plt.show()
```

4.2 Magnitude and Phase Response

```
t = np.linspace(-np.pi,np.pi,N_sam+1);t = t[:-1]
n = np.arange(N_sam)
```

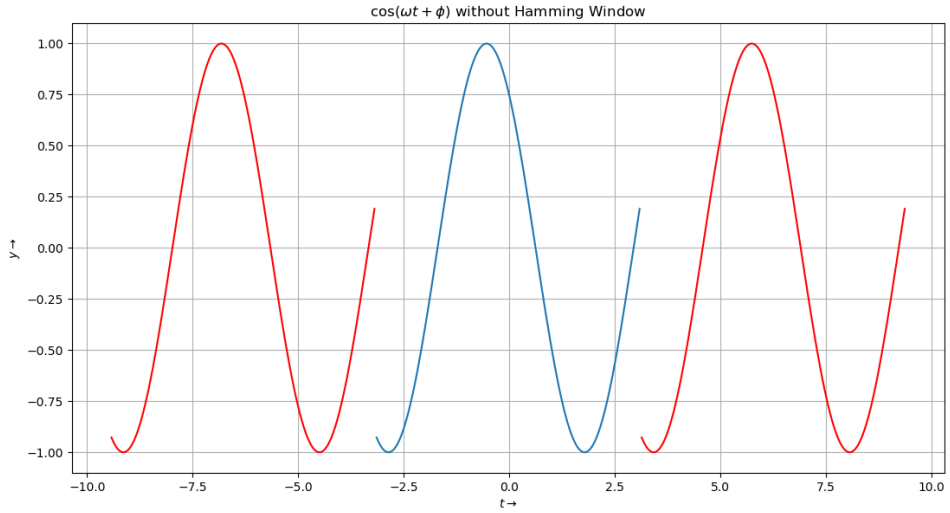

Figure 7: $\cos(\omega t + \phi)$ without Hamming Window

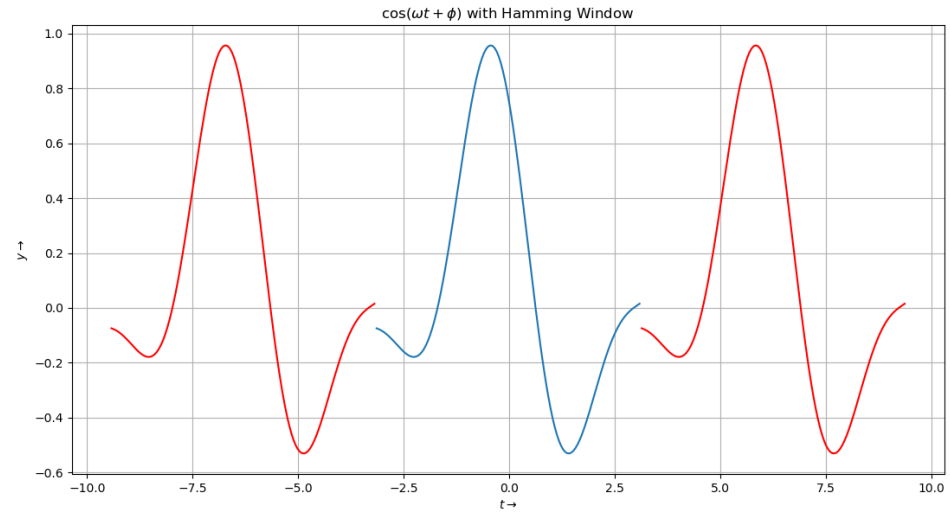

Figure 8: $\cos(\omega t + \phi)$ with Hamming Window

```
wnd = np.fft.fftshift(0.54+0.46*np.cos(2*np.pi*n/(N_sam-1)))
y = np.cos(omega*t + phi)
y = y*wnd
y[0] = 0
y = np.fft.fftshift(y)
y = np.fft.fftshift(np.fft.fft(y))/N_sam
w = np.linspace(-np.pi*fmax,np.pi*fmax,N_sam+1);w = w[:-1]

plt.figure(figsize=(16,8))
plt.subplot(2,1,1)
plt.grid(True)
plt.title(r'Magnitude of cos($\omega t + \phi$)')
plt.xlabel(r'$k\rightarrow$')
plt.ylabel(r'$|Y|\rightarrow$')
```

```
plt.xlim([-40,40])
plt.stem(w,abs(y),markerfmt='.')

plt.subplot(2,1,2)
plt.grid(True)
plt.title(r'Phase of cos($\omega t + \phi$)')
plt.xlabel(r'$k\rightarrow$')
plt.ylabel(r'$|Y|\rightarrow$')
plt.xlim([-40,40])
ii = np.where(abs(y)>=0.001)
jj = np.where(abs(z)>=0.001)
plt.stem(w[ii],np.angle(y[ii]),markerfmt='.')
plt.stem(w[jj],np.angle(z[jj]),markerfmt='+')
plt.show()
```

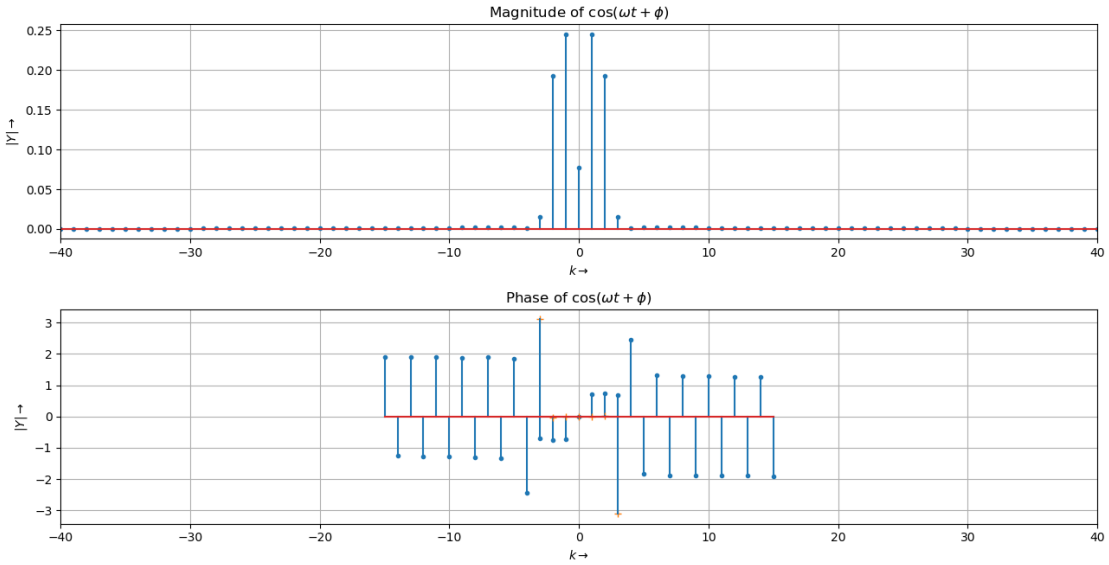

Figure 9: Magnitude and Phase Plot of $cos(\omega t + \phi)$

4.3 Extrapolating the Graph

4.3.1 Finding ω

The magnitude response is not resolvable because $0.5 \leq \omega \leq 1.5$.So,we use a form of weighted sum to find the frequency of the signal.The formula used is:

$$\hat{\omega} = \frac{\sum_{k=-N/2}^{k=+N/2} |Y(k)|^{2.1} \omega_k}{\sum_{k=-N/2}^{k=+N/2} |Y(k)|^{2.1}} \tag{3}$$

The value calculated is 1.345238 compared to the actual value 1.352253.

4.3.2 Finding ϕ

Plot the phase response of $g(t) = cos(t)$. here,we get two peaks only. Subtract the magnitude of the phase of $g(t)$ at $\omega = 1$ from the magnitude of the phase of $f(t)$ at $\omega = 1$.
The calculated value is 0.723299 compared to the actual value 0.723002.

5 Problem 4

5.1 Time Domain Signal

```
seed_omega = 485
np.random.seed(seed_omega)
omega = np.random.random() + 0.5
seed_phi = 5655
np.random.seed(seed_phi)
phi = np.random.random()

print('Actual value of omega:{}'.format(omega))
print('Actual value of phi:{}'.format(phi))
N_sam = 128
t = np.linspace(-np.pi,np.pi,N_sam+1);t = t[:-1]
t1 = np.linspace(-3*np.pi,-np.pi,N_sam+1);t1 = t1[:-1]
t2 = np.linspace(np.pi,3*np.pi,N_sam+1);t2 = t2[:-1]
dt = t[1] - t[0];fmax = 1/dt
seed_noise = 565
np.random.seed(seed_noise)
y = np.cos(omega*t + phi) + 0.1*np.random.randn(N_sam)
plt.figure(figsize=(16,8))
plt.grid(True)
plt.title(r'cos($\omega t + \phi$) without Hamming Window')
plt.xlabel(r'$t\rightarrow$')
plt.ylabel(r'$y\rightarrow$')
plt.plot(t,y)
plt.plot(t1,y,'r')
plt.plot(t2,y,'r')
plt.show()

t = np.linspace(-np.pi,np.pi,N_sam+1);t = t[:-1]
n = np.arange(N_sam)
wnd = np.fft.fftshift(0.54+0.46*np.cos(2*np.pi*n/(N_sam-1)))
y = y*wnd
plt.figure(figsize=(16,8))
plt.grid(True)
plt.title(r'cos($\omega t + \phi$)+Gaussian Noise with Hamming Window')
plt.xlabel(r'$t\rightarrow$')
plt.ylabel(r'$y\rightarrow$')
plt.plot(t,y)
plt.plot(t1,y,'r')
plt.plot(t2,y,'r')
plt.show()
```

5.2 Magnitude and Phase Response

```
t = np.linspace(-np.pi,np.pi,N_sam+1);t = t[:-1]
n = np.arange(N_sam)
wnd = np.fft.fftshift(0.54+0.46*np.cos(2*np.pi*n/(N_sam-1)))
y = np.cos(omega*t + phi)
y = y*wnd
y[0] = 0
y = np.fft.fftshift(y)
y = np.fft.fftshift(np.fft.fft(y))/N_sam
w = np.linspace(-np.pi*fmax,np.pi*fmax,N_sam+1);w = w[:-1]
```

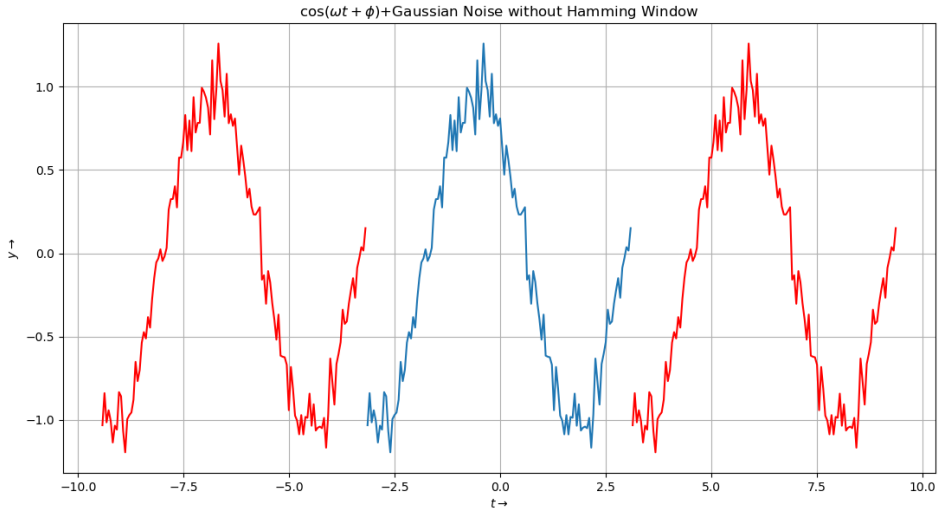

Figure 10: $\cos(\omega t + \phi)$+Gaussian Noise without Hamming Window

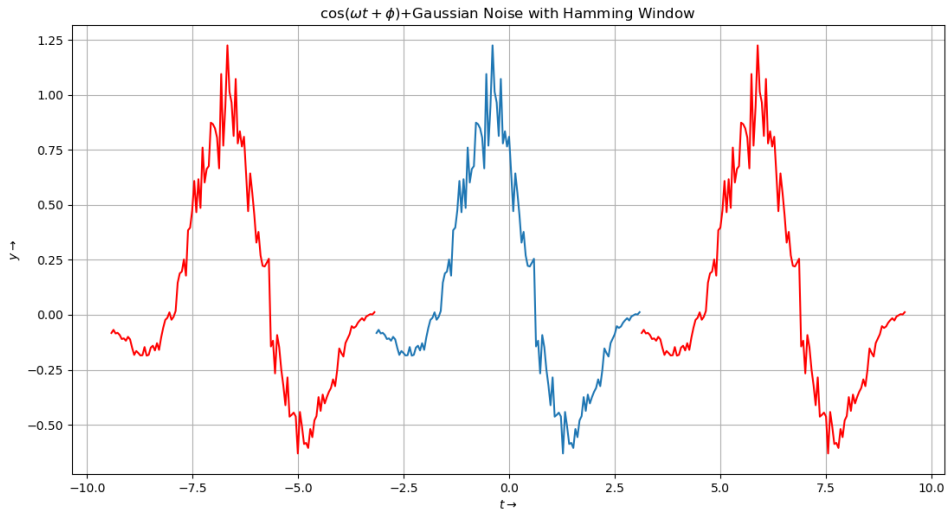

Figure 11: $\cos(\omega t + \phi)$+Gaussian Noise with Hamming Window

```
plt.figure(figsize=(16,8))
plt.subplot(2,1,1)
plt.grid(True)
plt.title(r'Magnitude of cos($\omega t + \phi$)+Gaussian Noise')
plt.xlabel(r'$k\rightarrow$')
plt.ylabel(r'$|Y|\rightarrow$')
plt.xlim([-40,40])
plt.stem(w,abs(y),markerfmt='.')

plt.subplot(2,1,2)
plt.grid(True)
plt.title(r'Phase of cos($\omega t + \phi$)+Gaussian Noise')
plt.xlabel(r'$k\rightarrow$')
plt.ylabel(r'$|Y|\rightarrow$')
```

```
plt.xlim([-40,40])
ii = np.where(abs(y)>=0.001)
jj = np.where(abs(z)>=0.001)
plt.stem(w[ii],np.angle(y[ii]),markerfmt='.')
plt.stem(w[jj],np.angle(z[jj]),markerfmt='+')
plt.show()
```

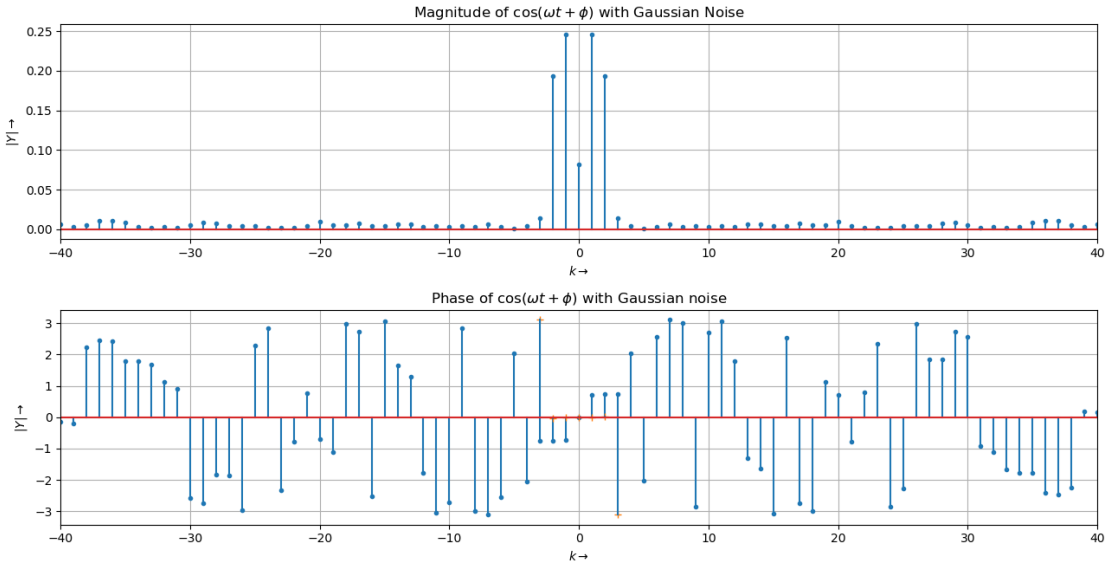

Figure 12: Magnitude and Phase Plot of $cos(\omega t + \phi)$+Gaussian Noise

5.3 Extrapolating the Graph

5.3.1 Finding ω

The magnitude response is not resolvable because $0.5 \leq \omega \leq 1.5$.So,we use a form of weighted sum to find the frequency of the signal.The formula used is:

$$\hat{\omega} = \frac{\sum_{k=-N/2}^{k=+N/2} |Y(k)|^{2.6} \omega_k}{\sum_{k=-N/2}^{k=+N/2} |Y(k)|^{2.6}} \tag{4}$$

The value calculated is 1.399889 compared to the actual value 1.352253.

5.3.2 Finding ϕ

Plot the phase response of $g(t) = cos(t)$. here,we get two peaks only. Subtract the magnitude of the phase of $g(t)$ at $\omega = 1$ from the magnitude of the phase of $f(t)$ at $\omega = 1$.
The calculated value is 0.711379 compared to the actual value 0.723002.

6 Problem 5

6.1 Time Domain Response

```
N = 1024
t = np.linspace(-np.pi,np.pi,N+1);t = t[:-1]
t1 = np.linspace(-3*np.pi,-np.pi,N+1);t1 = t1[:-1]
t2 = np.linspace(np.pi,3*np.pi,N+1);t2 = t2[:-1]
dt = t[1] - t[0];fmax = 1/dt
```

```
y = (np.cos(16*t*(1.5+t/(2*np.pi))))
n = np.arange(N)
wnd = np.fft.fftshift(0.54+0.46*np.cos(2*np.pi*n/(N-1)))
y = y*wnd
plt.figure(figsize=(16,8))
plt.grid()
plt.title(r'$cos(16t(1.5+\frac{t}{2\pi}))w(t)$')
plt.ylabel(r'$y\rightarrow$')
plt.xlabel(r'$t\rightarrow$')
plt.plot(t,y,'r')
plt.plot(t1,y,'b')
plt.plot(t2,y,'b')
plt.show()
```

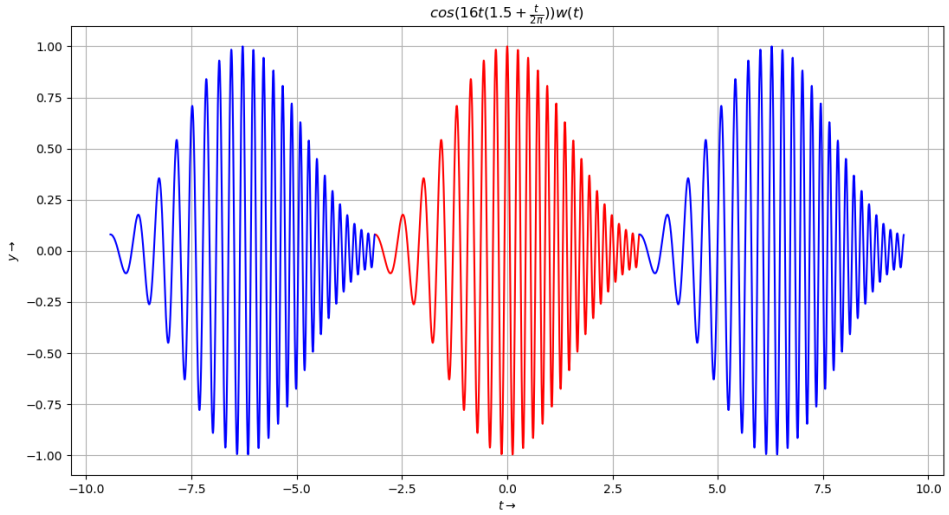

Figure 13: $cos(16t(1.5 + \frac{t}{2\pi}))w(t)$

From the graph it is evident that the frequency changes linearly from $-\pi$ to π

6.2 Magnitude Response

```
t = np.linspace(-np.pi,np.pi,N+1);t = t[:-1]
dt = t[1] - t[0];fmax = 1/dt
n = np.arange(N)
wnd = np.fft.fftshift(0.54+0.46*np.cos(2*np.pi*n/(N-1)))
y = y*wnd
y[0] = 0
y = np.fft.fftshift(y)
y = np.fft.fftshift(np.fft.fft(y))/N
w = np.linspace(-np.pi*fmax,np.pi*fmax,N+1);w = w[:-1]
plt.figure(figsize=(16,8))
plt.grid()
plt.title(r'$Magnitude\;of\;cos(16t(1.5+\frac{t}{2\pi}))$')
plt.ylabel(r'$|Y|\rightarrow$')
plt.xlabel(r'$\omega\rightarrow$')
plt.xlim([-100,100])
plt.stem(w,abs(y),markerfmt='.')
plt.show()
```

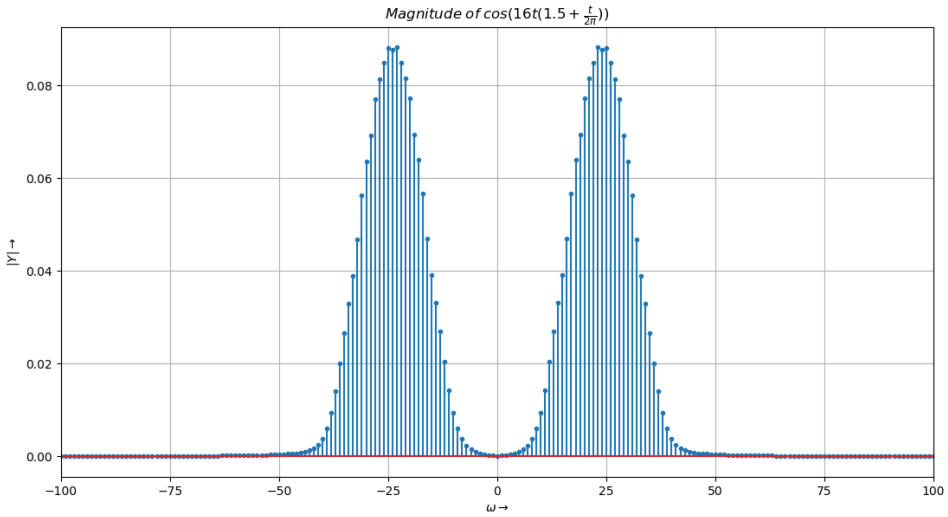

Figure 14: *Magnitude of* $cos(16t(1.5 + \frac{t}{2\pi}))$

7 Problem 6

In this problem, we basically plot the Time - Frequency Plot.We observe how the magnitude response changes as the frequency linearly over the time.

```
t = np.reshape(t,(64,-1))
Y = np.zeros_like(t,dtype=np.complex128)
for i in range(t.shape[1]):
    x = t[:,i]
    y = Y[:,i]
    y = np.cos(16*x*(1.5+x/(2*np.pi)))
    n = np.arange(64)
    wnd = np.fft.fftshift(0.54+0.46*np.cos(2*np.pi*n/(63)))
    y = y*wnd
    y[0] = 0
    y = np.fft.fftshift(y)
    Y[:,i] = np.fft.fftshift(np.fft.fft(y))/64.0
```

7.1 Contour Plot

```
x = np.arange(t.shape[1])
y = np.arange(t.shape[0])
y,x = np.meshgrid(y,x)

Y = np.fft.fftshift(Y,axes=0)
plt.figure(figsize=(16,8))
plt.grid()
plt.title(r'$Time-Frequency\;Plot$')
plt.ylabel(r'$y\rightarrow$')
plt.xlabel(r'$x\rightarrow$')
cp = plt.contour(y,x,abs(Y.T),20,cmap=cm.jet)
plt.clabel(cp,inline=True,fontsize=7)
plt.show()
```

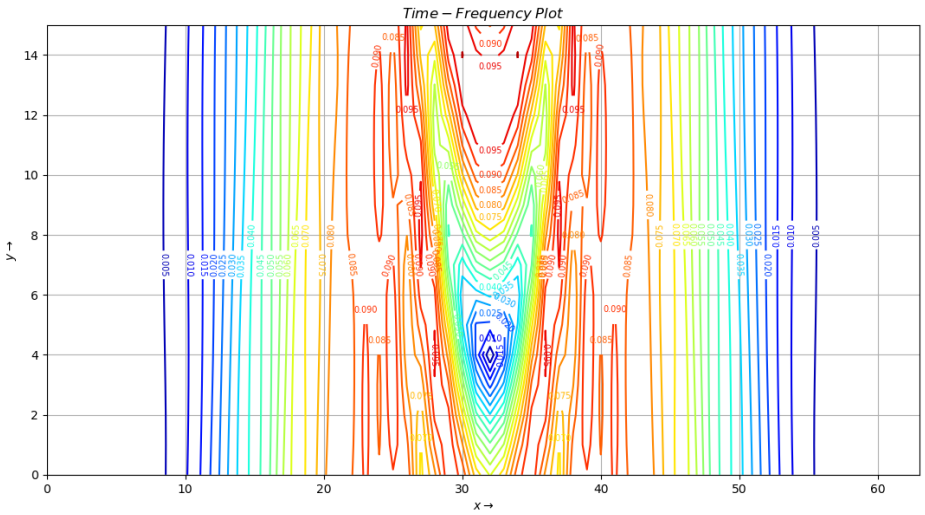

Figure 15: Contour Plot of Time Frequency Plot

7.2 Surface Plot

```
fig = plt.figure(figsize=(16,8))
ax = fig.gca(projection='3d')
surf = ax.plot_surface(y,x,abs(Y.T),cmap=cm.jet,linewidth=0, antialiased=False)
plt.title(r'Surface Plot of Time-Frequency Plot')
fig.colorbar(surf, shrink=0.5, aspect=5)
plt.show()
```

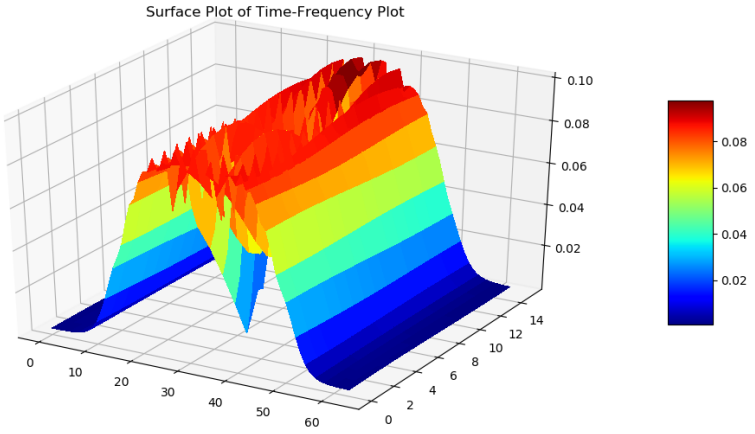

Figure 16: Surface Plot of Time Frequency Plot

8 Conclusion

In this assignment,we found out the spectrum of non-periodic signals and analyzed the frequency components in them.