Assignment 9 : Spectra of non-periodic signals

Gudivada Harshad Kumar
EE20B038

April 19, 2022

Abstract

The main aim of the assignment 9 is

- Obtaining DFT of non-periodic functions
- Using Hamming window to make better DFTs.
- Plot the graphs

DFT of $sin(\sqrt{2}t)$

The spectrum of $sin(\sqrt{2}t)$ without windowing is plotted using the following python code snippet.

```
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(num=0,figsize=(7,7))
subplot(2,1,1)
plot(w,abs(Y),lw=2)
xlim([-10,10])
ylabel(r"$|Y|\rightarrow$")
title(r"Spectrum of $\sin\left(\sqrt{2}t\right)$")
grid(True)
subplot(2,1,2)
plot(w,angle(Y),'ro',lw=2)
xlim([-10,10])
```

```
ylabel(r"Phase of $Y\rightarrow$",size=16)
xlabel(r"$\omega\rightarrow$",size=16)
grid(True)
```

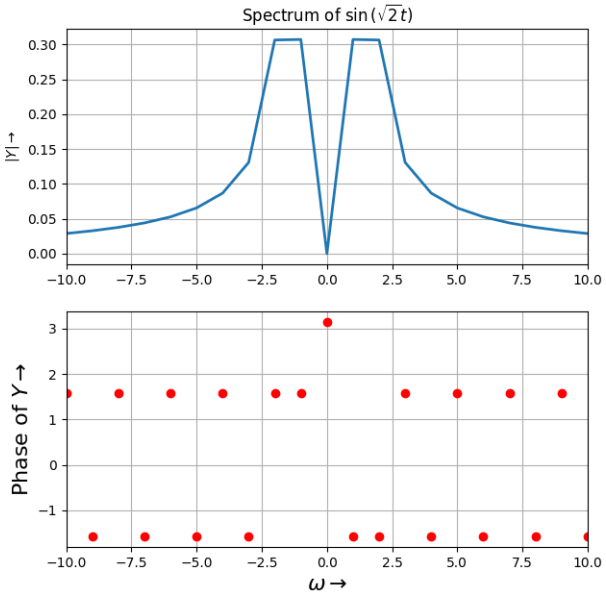

Figure 1: Spectrum of $sin(\sqrt{2}t)$ without windowing

Now we multiply the given function with hamming window function and plot the spectrum for it. The hamming window removes discontinuities by attenuating the high frequency components that cause the discontinuities. The hamming window function is given by

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

The following python code snippet helps in potting the spectrum of the windowed function.

```
t = linspace(-4*pi,4*pi,257); t = t[:-1]
dt = t[1]-t[0]; fmax = 1/dt
n = arange(256)
wnd = fftshift(0.54+0.46*cos(2*pi*n/256))
y = sin(sqrt(2)*t)*wnd
y[0] = 0
y = fftshift(y)
Y = fftshift(fft(y))/256.0
```

```
w = linspace(-pi*fmax,pi*fmax,257); w = w[:-1]
figure(num=1,figsize = (7,7))
subplot(2,1,1)
plot(w,abs(Y),lw=2)
xlim([-4,4])
ylabel(r"$|Y|\rightarrow$")
title(r"Improved Spectrum of $\sin\left(\sqrt{2}t\right)$")
grid(True)
subplot(2,1,2)
plot(w,angle(Y),'ro',lw=2)
xlim([-4,4])
ylabel(r"Phase of $Y\rightarrow$")
xlabel(r"$\omega\rightarrow$")
grid(True)
```

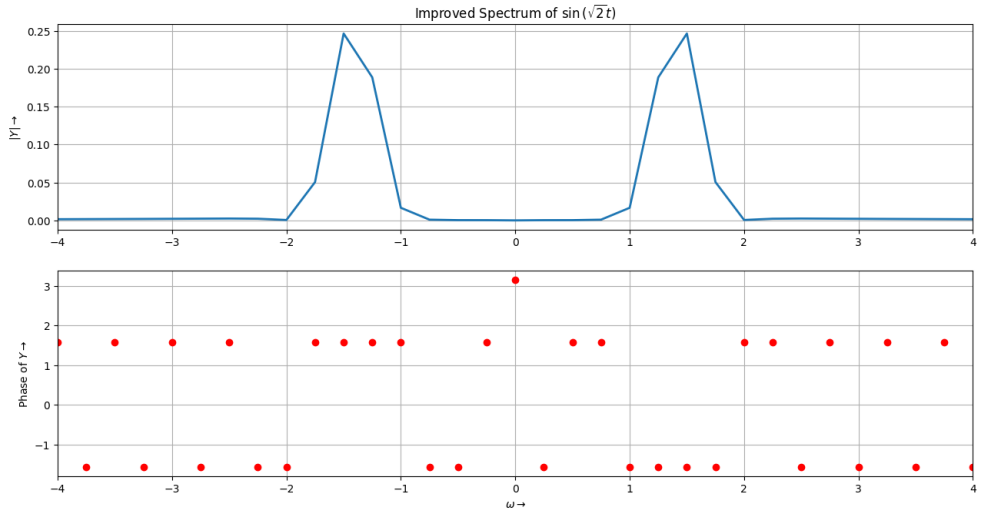

Figure 2: Spectrum of $sin(\sqrt{2}t)$ after windowing

DFT of $cos^3(0.86t)$

Now we consider the function $cos^3(0.86t)$ and obtain FFT with and without hamming window. We use the following python code snippet to do it.

```
y = cos(0.86*t)**3
y1 = y*wnd
y[0]=0
```

```
y1[0]=0
y = fftshift(y)
y1 = fftshift(y1)
Y = fftshift(fft(y))/256.0
Y1 = fftshift(fft(y1))/256.0
```

Without windowing

```
figure(num=2,figsize=(7,7))
subplot(2,1,1)
plot(w,abs(Y),lw=2)
xlim([-4,4])
ylabel(r"$|Y|\rightarrow$")
title(r"Spectrum of $\cos^{3}(0.86t)$ without Hamming window")
grid(True)
subplot(2,1,2)
plot(w,angle(Y),'ro',lw=2)
xlim([-4,4])
ylabel(r"Phase of $Y\rightarrow$")
xlabel(r"$\omega\rightarrow$")
grid(True)
```

With windowing

```
figure(num=3,figsize=(7,7))
subplot(2,1,1)
plot(w,abs(Y1),lw=2)
xlim([-4,4])
ylabel(r"$|Y|\rightarrow$")
title(r"Spectrum of $\cos^{3}(0.86t)$ with Hamming window")
grid(True)
subplot(2,1,2)
plot(w,angle(Y1),'ro',lw=2)
xlim([-4,4])
ylabel(r"Phase of $Y\rightarrow$")
xlabel(r"$\omega\rightarrow$")
grid(True)
```

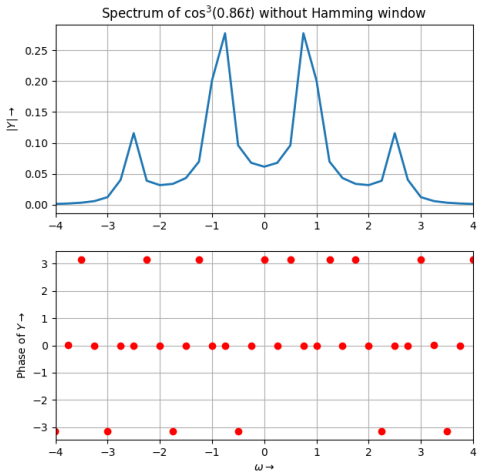

Figure 3: Spectrum of $\cos^3(0.86t)$ without windowing

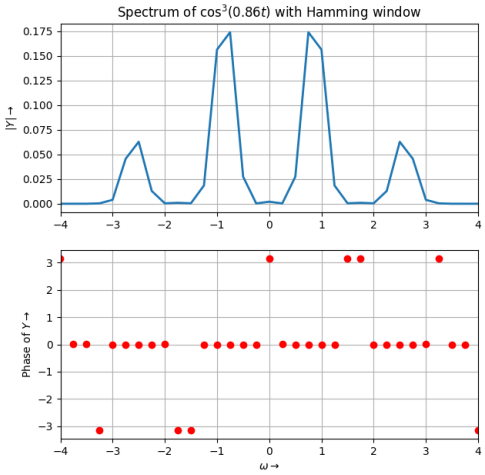

Figure 4: Spectrum of $\cos^3(0.86t)$ after windowing

It can be noticed that a lot of the energy is stored in frequencies that aren't a part of the signal. After windowing, these frequencies are attenuated and hence the peaks are sharper in the windowed function. It is still not an impulse because the convolution with the Fourier transform of the windowed function smears out the peak

Estimating ω_0 and δ

We need to estimate ω and δ for a signal $\cos(\omega t+\delta)$ for 128 samples between $[-\pi, \pi)$. We estimate omega using a weighted average. We use the following python code snippet to plot the spectrum

```
w0 = 1.5
delta = 0.5
t = linspace(-pi,pi,129)[:-1]
dt = t[1]-t[0]; fmax = 1/dt
n = arange(128)
wnd = fftshift(0.54+0.46*cos(2*pi*n/128))
y = cos(w0*t + delta)*wnd
y[0]=0
y = fftshift(y)
Y = fftshift(fft(y))/128.0
w = linspace(-pi*fmax,pi*fmax,129); w = w[:-1]
figure(num=4,figsize=(7,7))
subplot(2,1,1)
plot(w,abs(Y),lw=2)
xlim([-4,4])
ylabel(r"$|Y|\rightarrow$")
title(r"Spectrum of $\cos(w_0t+\delta)$ with Hamming window")
grid(True)
subplot(2,1,2)
plot(w,angle(Y),'ro',lw=2)
xlim([-4,4])
ylabel(r"Phase of $Y\rightarrow$")
xlabel(r"$\omega\rightarrow$")
grid(True)
```

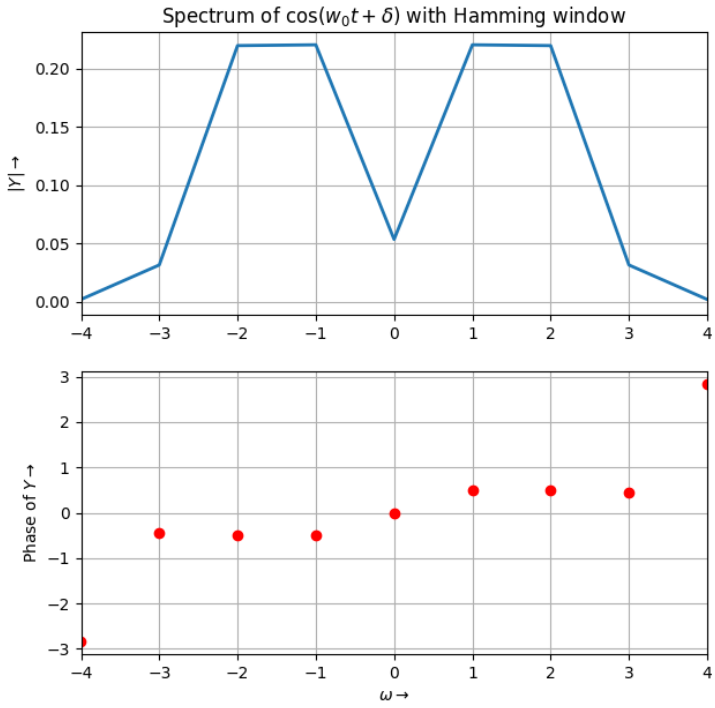

Figure 5: Spectrum of $\cos(1.5t + 0.5)$ after windowing

We use the following functions to find the values of ω_0 and δ

```
def est_omega(w,Y):
    ii1 = where(w>0)
    omega = (sum(abs(Y[ii1])**2*w[ii1])/sum(abs(Y[ii1])**2))
    return omega

def est_delta(w,Y,sup = 1e-4,window = 1):
    ii2=np.where(np.logical_and(np.abs(Y)>sup, w>0))[0]
    np.sort(ii2)
    points=ii2[1:window+1]
    return (np.sum(np.angle(Y[points]))/len(points))
```

In this no noise case we get the valeus of ω_0 and δ to be 1.5163179648582412 and 0.506776265719626 respectively

Estimating ω_0 and δ when there is a noise

Now we repeat the same process as above with added noise and find the values of ω_0 and δ. We use the following python code snippet to plot the spectrum of this noise added signal.

```
y = (cos(w0*t + delta) + 0.1*randn(128))*wnd
y[0]=0
y = fftshift(y)
Y = fftshift(fft(y))/128.0
figure(num=5,figsize=(7,7))
subplot(2,1,1)
plot(w,abs(Y),lw=2)
xlim([-4,4])
ylabel(r"$|Y|\rightarrow$")
title(r"Spectrum of a noisy $\cos(w_0t+\delta)$ with Hamming window")
grid(True)
subplot(2,1,2)
plot(w,angle(Y),'ro',lw=2)
xlim([-4,4])
ylabel(r"Phase of $Y\rightarrow$")
xlabel(r"$\omega\rightarrow$")
grid(True)
```

Here we get the values of ω_0 and δ to be 2.0565303502054797 and 0.5139013812501927 respectively

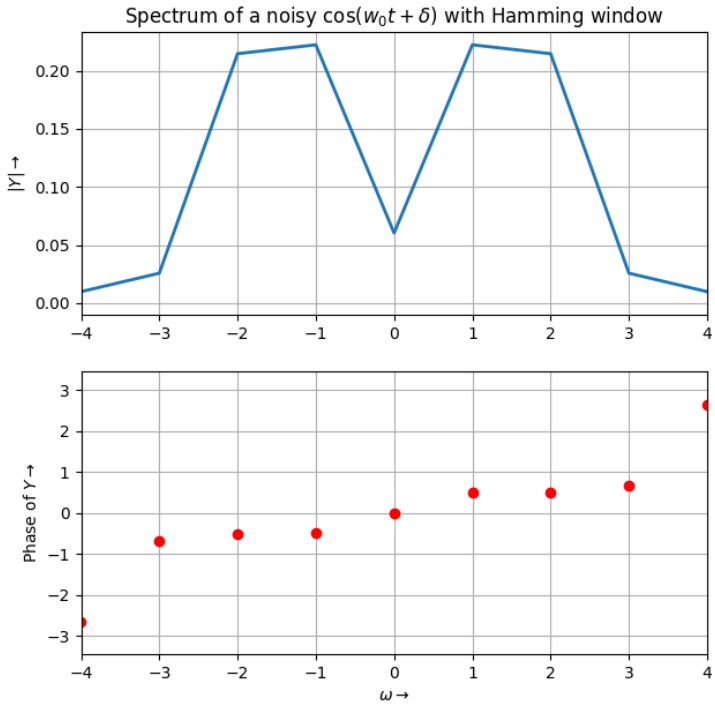

Figure 6: Spectrum of $\cos(1.5t + 0.5)$ with addednoise

Analysis of Chirp Signal

In this question we analyze a chirp signal which is an FM signal where frequency is directly proportional to time. A chirp signal we shall consider is given by

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

The following python code snippet is used for plotting the spectrum of the chriped signal

```
t = linspace(-pi,pi,1025); t = t[:-1]
dt = t[1]-t[0]; fmax = 1/dt
n = arange(1024)
wnd = fftshift(0.54+0.46*cos(2*pi*n/1024))
y = cos(16*t*(1.5 + t/(2*pi)))*wnd
y[0]=0
y = fftshift(y)
Y = fftshift(fft(y))/1024.0
w = linspace(-pi*fmax,pi*fmax,1025); w = w[:-1]
figure(num=6,figsize=(7,7))
subplot(2,1,1)
plot(w,abs(Y),lw=2)
xlim([-100,100])
ylabel(r"$|Y|\rightarrow$")
title(r"Spectrum of chirped function")
grid(True)
subplot(2,1,2)
plot(w,angle(Y),'ro',lw=2)
xlim([-100,100])
ylabel(r"Phase of $Y\rightarrow$")
xlabel(r"$\omega\rightarrow$")
grid(True)
```

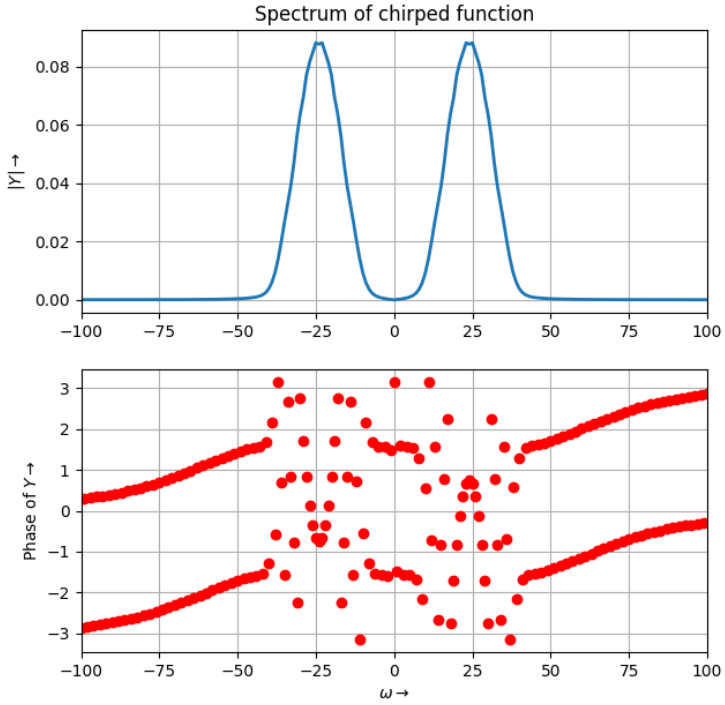

Figure 7: Spectrum of Chirp signal

Surface plot of Chirp Signal

For the same chirped signal, break the 1024 vector into pieces that are 64 samples wide. Extract the DFT of each and store as a column in a 2D array. Then plot the array as a surface plot to show how the frequency of the signal varies with time. Plot and analyse the time frequency plot, where we get localized DFTs and show how the spectrum evolves in time.
The following python code snippet helps in plotting time-frequency plot.

```
t=linspace(-pi,pi,1025);t=t[:-1]
t_arrays=split(t,16)

Y_mags=zeros((16,64))
Y_angles=zeros((16,64))
for i in range(len(t_arrays)):
n = arange(64)
wnd = fftshift(0.54+0.46*cos(2*pi*n/64))
y = cos(16*t_arrays[i]*(1.5 + t_arrays[i]/(2*pi)))*wnd
y[0]=0
y = fftshift(y)
Y = fftshift(fft(y))/64.0
Y_mags[i] = abs(Y)
Y_angles[i] = angle(Y)
t = t[::64]
w = linspace(-fmax*pi,fmax*pi,64+1); w = w[:-1]
t,w = meshgrid(t,w)

fig = figure(7)
ax = fig.add_subplot(111, projection='3d')
surf=ax.plot_surface(w,t,Y_mags.T,cmap='viridis',linewidth=0, antialiased=False)
fig.colorbar(surf, shrink=0.5, aspect=5)
ax.set_title('surface plot');
ylabel(r"$\omega\rightarrow$")
xlabel(r"$t\rightarrow$")
```

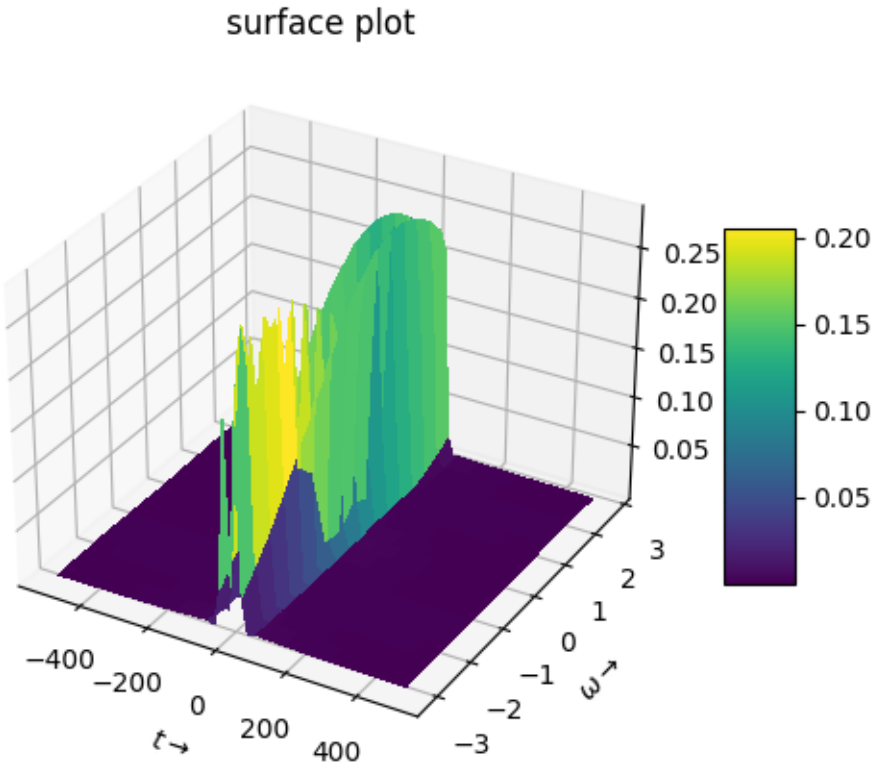

Figure 8: Spectrum of Chirp signal

Conclusion

In this assignment we have covered the requirement of windowing in the case of non-periodic series in DFT's. In particular this is to mitigate the effect of Gibbs phenomena owing to the discontinuous nature of the series $\tilde{x}[n]$ realised by a discrete fourier transform. We also addressed the time varying spectra of chirped signal where we plot fourier spectra for different time slices of a signal.