EE2703: Applied Programming Lab
Week 9: Spectra of Non-Periodic Signals

Author: Surya Prasad S, EE19B121

May 22nd, 2021

1 Abstract:

In last week's assignment, we had analysed periodic functions. This week we shall analyse non-periodic signals and minimise the errors that arise from Gibbs phenomenon. We shall also use a techinique called Hamming Windowing to reduce the errors.

2 Code and Plot Analysis:

2.1 Libraries imported:

```
from pylab import *
```

2.2 Common Function to perform FFT:

```
def plot_FFT(func, TITLE, r = [-pi, pi], N = 64, xlimit = 10, odd_signal =
    False, indB = False, estimate_flag = False):
    ## Time vector is being declared. Here we remove the last point because it
        will overlap with the initial point
    t = linspace(r[0], r[1], N + 1)[:-1]

    ## Frequency vector is being declared. temp is a just a temporary variable
        for the purpose of calculation
    temp = N * (pi/(r[1] - r[0]))
    w = linspace(-temp, temp, N + 1)[:-1]

    ## Computing functional values in the time domain and frequency domain
    ### For odd signals we make y[0] as 0 to reduce the error
    y = func(t)

    if odd_signal:
        y[0] = 0

    Y = fftshift(fft(fftshift(y)))/float(N)

    ## Plotting phase and magnitude plots for DFT of the signals
    figure()

    if indB:
        subplot(2, 1, 1)
        title("Frequency Spectrum of " + TITLE)
        semilogx(w, 20 * log10(abs(Y)), lw = 2)
        xlim([1, 10])
        ylim([-20, 0])
        ylabel(r"$|Y|$ (in dB)", size = 16)
        grid(True)

    else:
        subplot(2, 1, 1)
```

```
    title("Frequency Spectrum of " + TITLE)
    plot(w, abs(Y), lw = 2)
    xlim([-xlimit, xlimit])
    ylabel(r"$|Y|$", size = 16)
    grid(True)

subplot(2, 1, 2)
ii = where(abs(Y) > 1e-3)
scatter(w, angle(Y), marker = 'o', color = '#D9D9D9')
plot(w[ii], angle(Y[ii]), 'go', lw = 2)
xlim([-xlimit, xlimit])
ylabel(r"$\angle$$Y$ (in rad)", size = 16)
xlabel(r"$\omega$ (in rad/s)", size = 16)
grid(True)

show()

if estimate_flag:
    return w, abs(Y), angle(Y)
```

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

Let's try to find the DFT of the sinusoid.

$$Y[k] = -2j\sum_{n=0}^{\frac{N}{2}-1} y[n]sin(\frac{2\pi kn}{N}) + (-1)^k y[\frac{N}{2}]$$

$$y[0] = 0$$

For $i \in 1, 2, ..., \frac{N}{2} - 1$

$$y[i] = -y[N-i]$$

$$y[\tfrac{N}{2}] = sin(t_{\frac{N}{2}}) = sin(-t_{max})$$

$$Y[k] = \sum_{n=0}^{\frac{N}{2}-1} y[n](exp(j\frac{2\pi kn}{N}) - exp(-j\frac{2\pi kn}{N})) + y[\frac{N}{2}]exp(j\pi k)$$

$$Y[k] = -2j\sum_{n=0}^{\frac{N}{2}-1} y[n]sin(\frac{2\pi kn}{N}) + (-1)^k y[\frac{N}{2}]$$

We can see that the DFT is not completely imaginary even though the CTFT is purely imaginary. Hence we shall make $y[0]$ to be 0. Using this result, let's plot the Frequency Spectrum of $sin(\sqrt{2}t)$.

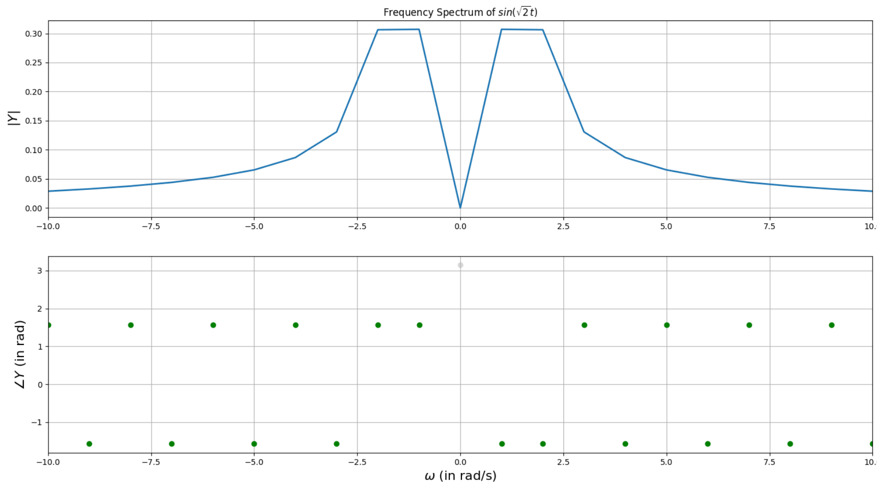


Figure 1: Frequency Spectrum of $sin(\sqrt{2}t)$ sampled 64 times

As we can see in the plot instead of two peaks at $\pm\sqrt{2}$, we got two peaks each with two values and a gradually decaying magnitude. The phase plot is as expected. If we look at the region we have sampled, and construct a periodic function with it we get the following plots:

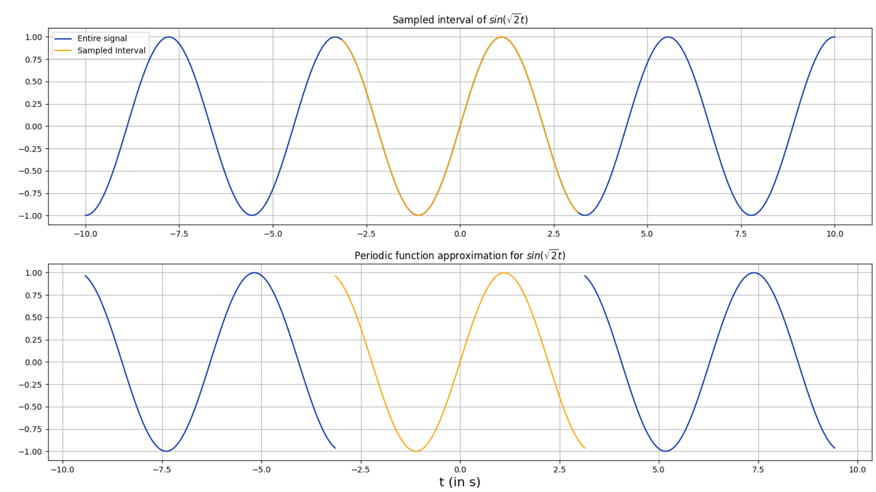


Figure 2: Periodic plot of sampled interval

The plot is discontinous. This leads to a large number of errors because of Gibbs phenomenon. Hence we see significant magnitudes at higher frequencies.

Let's try to understand this concept by analysing the unit ramp function.

$$f(t) = t, -\pi < t < \pi$$

The Fourier series of this ramp is

$$f(t) = 2(\frac{sint}{1} - \frac{sin2t}{2} + \frac{sin3t}{3} - \ldots)$$

So here the coefficients are expected to decay very slowly. Let's plot the magnitude response for the ramp function.

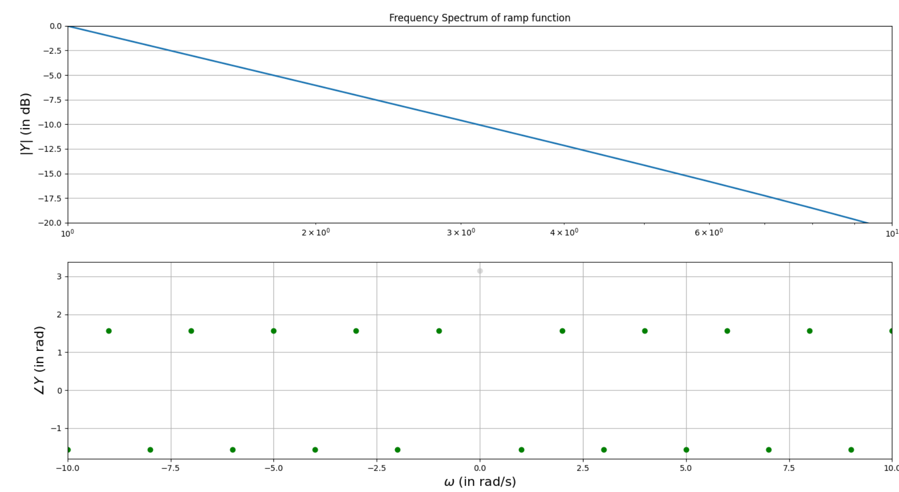


Figure 3: Frequency Spectrum of ramp function

From the plot, we can say that the spectrum decays at the rate of 20dB per decade (which is due to $1/\omega$). The big jump at $n\pi$ force this slowly decaying spectrum, which is why we don't see the expected spikes for the spectrum of $sin(\sqrt{2}t)$.

2.4 Windowing:

Windowing is a technique to reduce the discontinuities by damping the function near the discontinuities. For this we shall multiply the given function sequency by a "window" sequence $w[n]$:

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

The Hamming Window function we will be using is the following:

$$w[n] = \begin{cases} 0.54 + 0.46cos(\frac{2\pi n}{N-1}) & |n| \leq \frac{N-1}{2} \\ 0 & else \end{cases}$$

Code snippet for windowing is as follows:

```
def HammingWindow(a, b, n):
    return fftshift(a + b * cos((2 * pi * n)/(len(n) - 1)))
```

Now let's plot the sinusoidal function after applying Hamming Window.

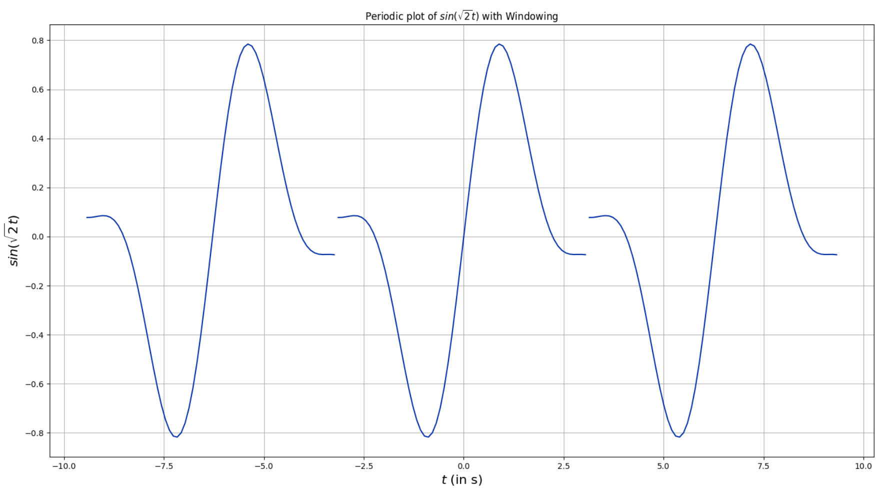


Figure 4: Periodic plot of $sin(\sqrt{2}t)$ after Windowing

The discontinuity is reduced significantly. The corresponding DFT plots are:

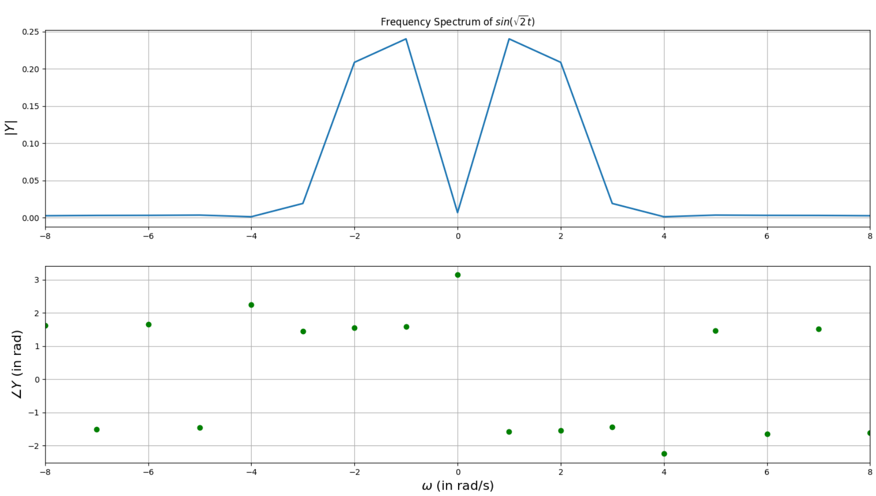


Figure 5: Frequency Spectrum of $sin(\sqrt{2}t)w(n)$

The magnitude response is much better now as there is only one peak. Let's try to increase the resolution to get better defined peaks.

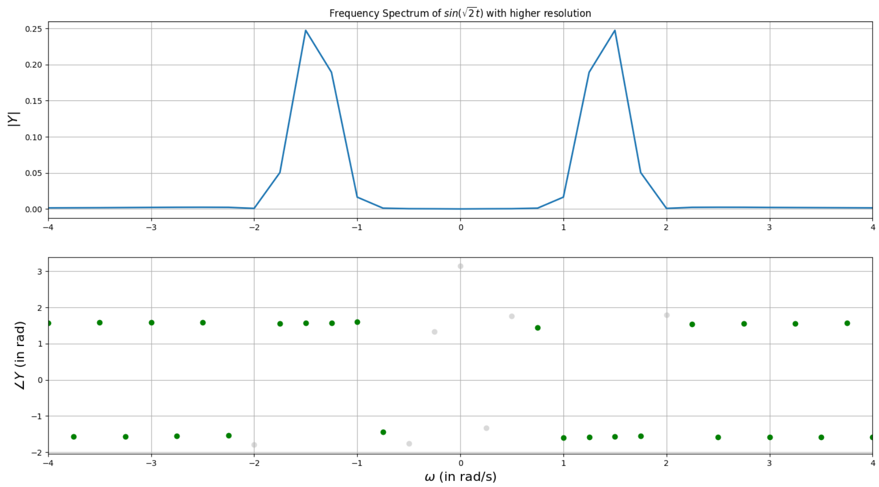


Figure 6: Frequency Spectrum of $sin(\sqrt{2}t)w(n)$ with higher resolution

The peaks are more well defined and the magnitude response falls more rapidly for higher values. We notice that the width of the peaks have increased a bit with windowing. Let's test this with another sinusoid.

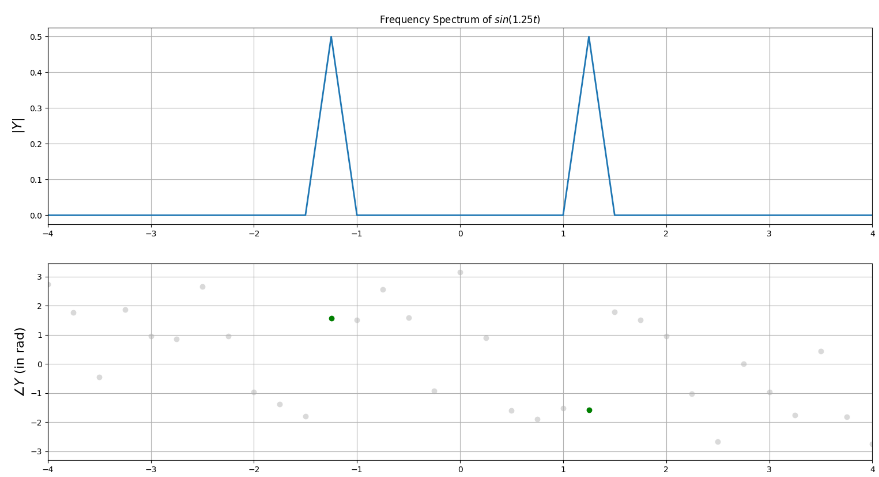


Figure 7: Frequency Spectrum of $sin(1.25t)$ without Windowing

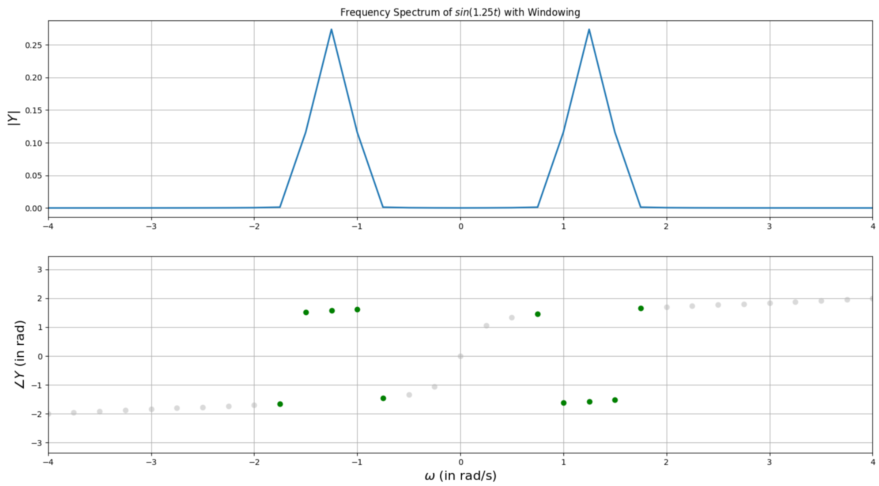


Figure 8: Frequency Spectrum of $sin(1.25t)$ with Windowing

For the same resolution, Hamming Window is actually increasing the width of the peak. This would create a problem if our resolution is not sufficiently high.

2.5 DFT Analysis of $cos^3(\omega_0 t)$:

Let's follow the same procedure for $cos^3(\omega_0 t)$ with $\omega_0 = 0.86$.

$$cos^3(0.86t) = \frac{1}{4}cos(3(0.86)t) + \frac{3}{4}cos(0.86t)$$

So we expect the peaks to be at ± 0.86 and ± 2.58.

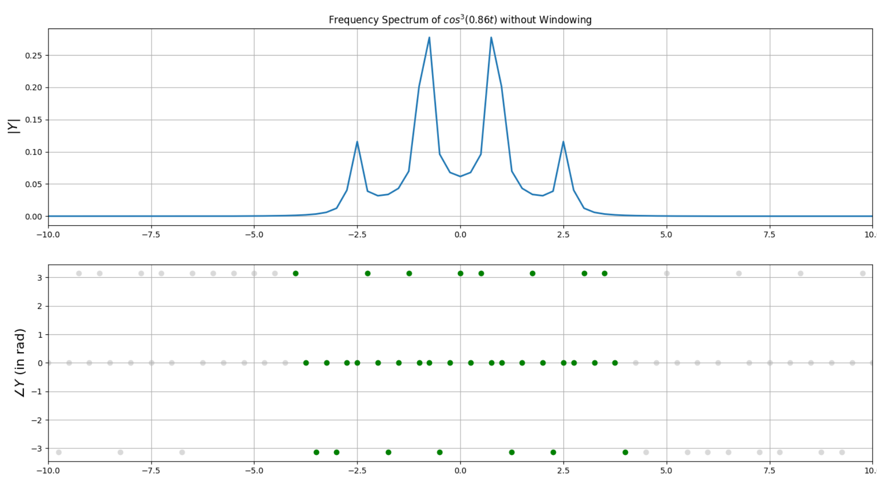


Figure 9: Frequency Spectrum of $cos^3(0.86t)$ without Windowing

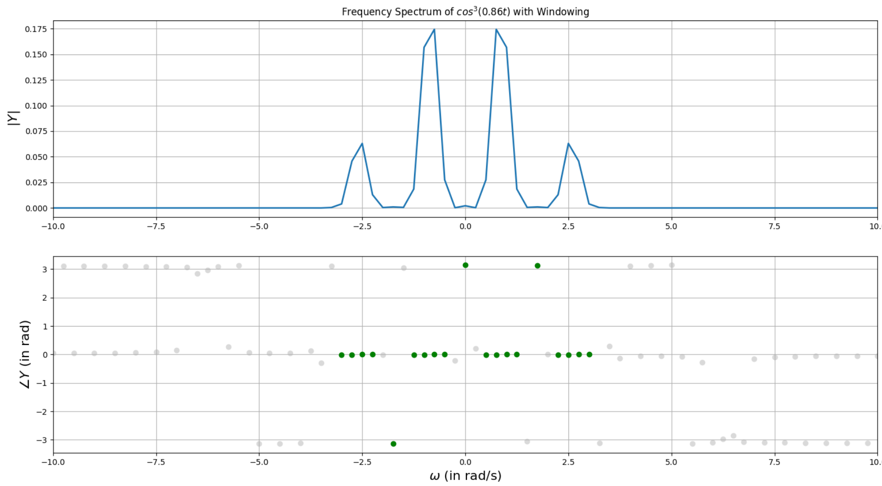


Figure 10: Frequency Spectrum of $cos^3(0.86t)$ with Windowing

We get the plots as expected with and without the Hamming Window.

2.6 DFT Analysis of $cos(\omega_0 t + \delta)$:

Let's follow the same procedure for $cos(\omega_0 t + \delta)$ with $\omega_0 = 0.6$ and $\delta = 1$. We would expect the peaks of the frequency spectrum to be located at $\pm\omega_0$.

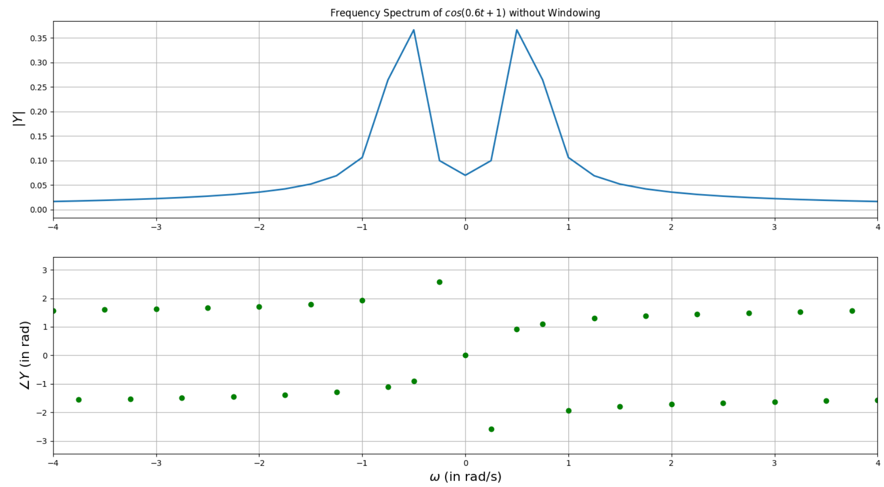


Figure 11: Frequency Spectrum of $cos(0.6t + 1)$ without Windowing

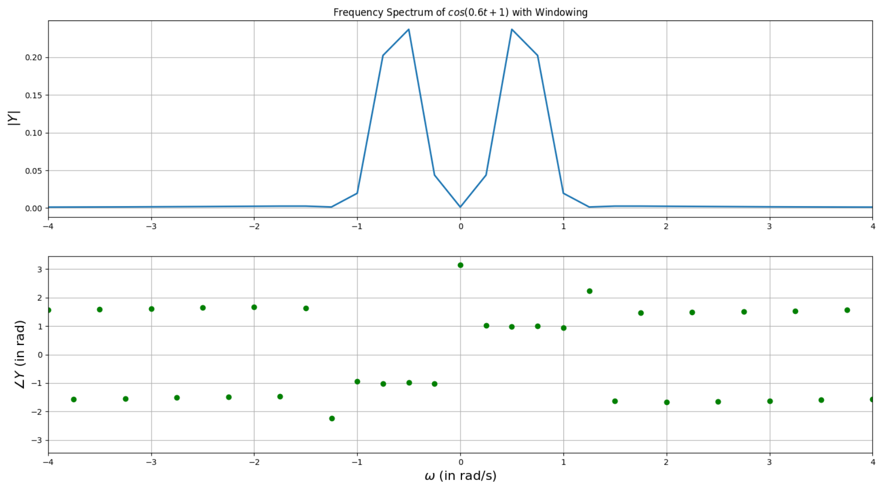


Figure 12: Frequency Spectrum of $cos(0.6t + 1)$ with Windowing

From the plots, we can see that the peaks are approximately at ± 0.6.

We can also try to estimate more accurately the values by performing a weighted average with the magnitude squared being the weight for all the frequencies under the peak. The code snippet for estimating ω_0 and δ is as follows:

```
def estimate(w, mag, phase):
    actual_mag = where(mag > 0.2)

    w_avg = sum((mag[actual_mag]**2) * abs(w[actual_mag]))/sum(mag[actual_mag]**2)

    phase_avg = mean(abs(phase[actual_mag]))

    print("Estimated w0:", w_avg)
    print("Estimated delta:", phase_avg)
```

```
Estimations for cos(0.6t+1):
Estimated w0: 0.5856280489361705
Estimated delta: 1.0062163015426053

Estimations for cos(0.6t+1)w(t):
Estimated w0: 0.6054218881468985
Estimated delta: 1.0010884777925986
```

The estimated values are very close to actual values.

2.7 DFT Analysis of $cos(\omega_0 t + \delta)$ with added white gaussian noise:

We shall now add some white gaussian noise to the previous cosine signal. So the resulting signal will be:

```
def cos_delta_with_noise(x):
    return cos(0.6 * x + 1) + 0.1 * randn(len(x))
```

We shall perform the same analysis for this too:

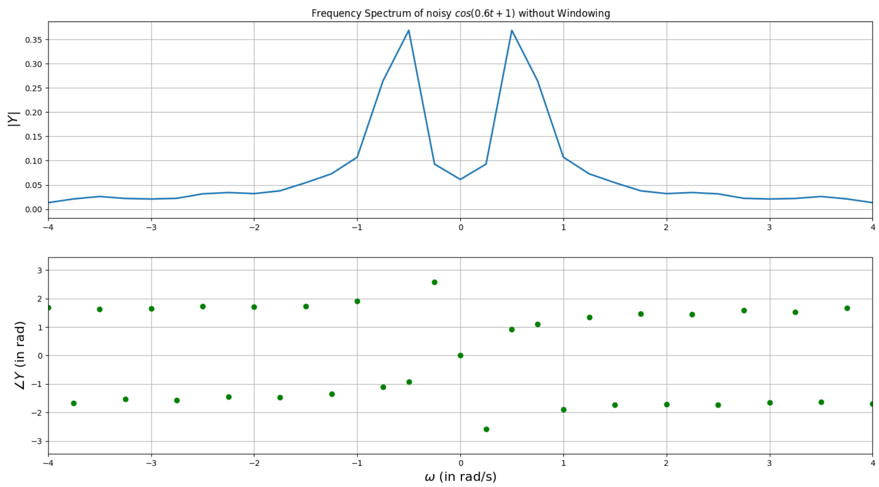


Figure 13: Frequency Spectrum of noisy $cos(0.6t + 1)$ without Windowing

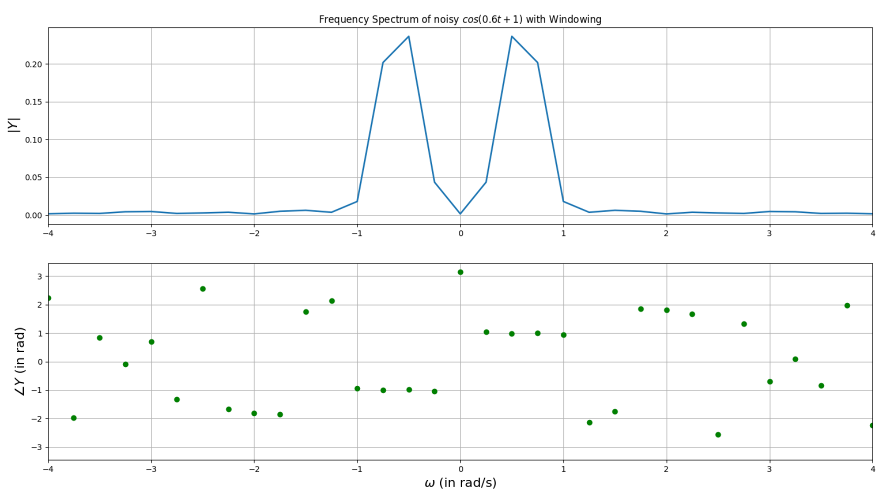


Figure 14: Frequency Spectrum of noisy $cos(0.6t + 1)$ with Windowing

As we can see, there is a very small distortion in the frequency spectra. We can also do the same estimation of ω_0 and δ.

```
Estimations for noisy cos(0.6t+1):
Estimated w0: 0.5849385977821686
Estimated delta: 0.999694716712083

Estimations for noisy cos(0.6t+1)w(t):
Estimated w0: 0.6043866653600212
Estimated delta: 1.0027201730163195
```

As we can see, the values estimated are very close to the actual values. There is very less effect due to the noise. Let us look at the effect for higher noise:

```
Estimations for noisy cos(0.6t+1):
Estimated w0: 0.5872547551846261
Estimated delta: 1.05992394655917

Estimations for noisy cos(0.6t+1)w(t):
Estimated w0: 0.6184164246038958
Estimated delta: 1.0089962410560627
```

For 10 times more noise, the error increased very slightly.

2.8 DFT Analysis of Chirped signal:

Let's first have a look at the plots for a chirped signal which is given by:

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

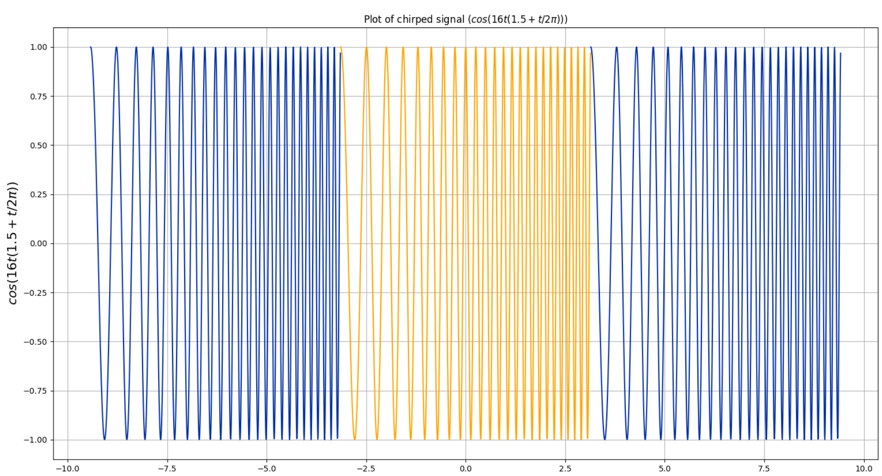


Figure 15: Periodic approximation of chirped signal

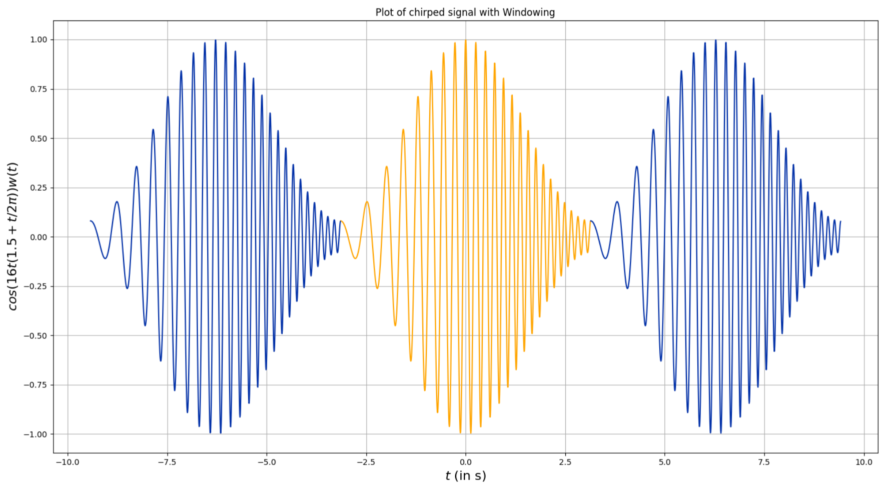


Figure 16: Chirp signal with Windowing

As we can see, the discontinuity appears to have been made neglible by windowing. Code snippet for chirped signal:

```
## Function definition for chirp function
def chirp(x):
    return cos(16 * x * (1.5 + x/(2 * pi)))

## Function definition for chirp function with Windowing
def chirp_hamming(x):
    return chirp(x) * HammingWindow(0.54, 0.46, arange(len(x)))
```

Plots of Magnitude and Phase response for a chirped signal:

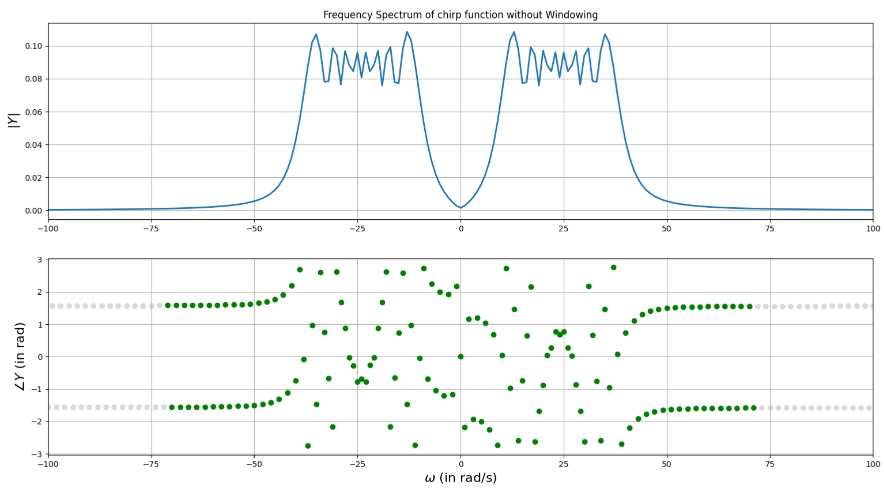


Figure 17: Frequency Spectrum of chirp function without Windowing

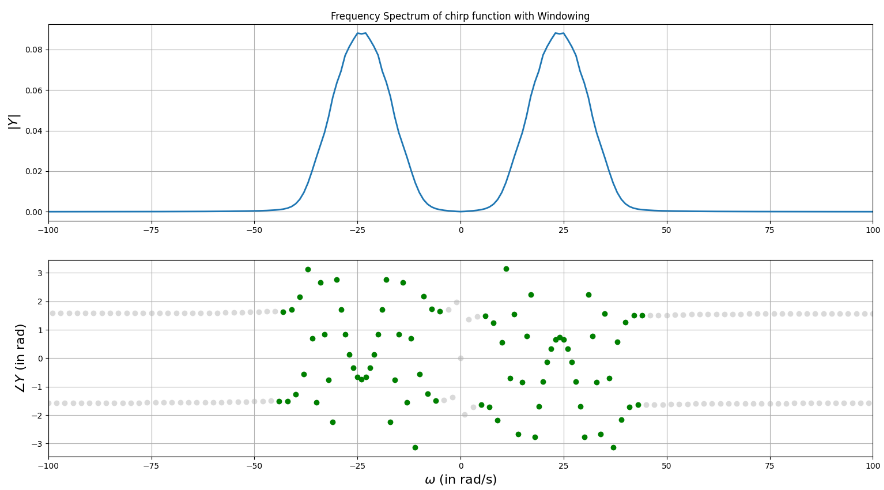
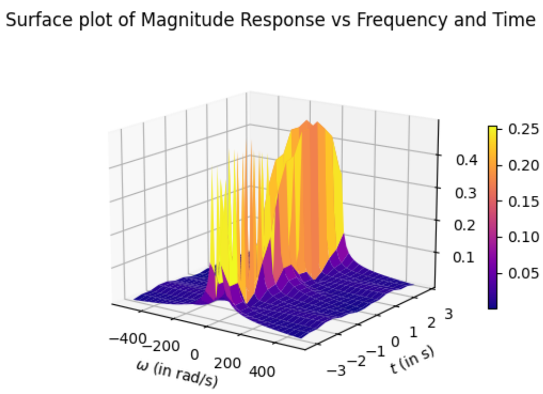


Figure 18: Frequency Spectrum of chirp function with Windowing

There's a drastic change in the plot. This is because the window function removed the discontinuities almost completely. This is the advantage of Hamming Window. The main frequency of the signal was isolated and made more easily identifiable by the window function. In the case of chirped signal, the first term ($24t$) is clearly the more significant term and in the final plot we see the peak to be at around 24. We also can expect the spectra to change with time interval. We shall break 1024 samples from $[-\pi, \pi)$ into 16 contiguous pieces of 64 samples each and find DFT for each interval.



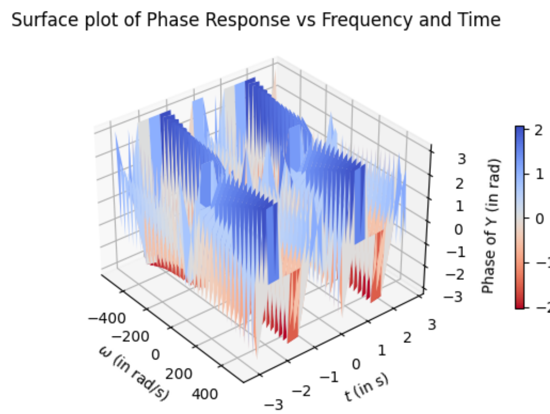


Figure 19: Frequency Spectra of chirped signal for different time ranges

The gap between the peaks increases with time. Also the DFT gets more well defined peaks with increasing time.

3 Conclusion:

Thus, the Frequency Spectrum of some non-periodic signals were analysed and plotted. We made use of Hamming Window and properties of odd signals to reduce the errors which arise due to their non-periodic nature. We also notice that Hamming Window increases the width of the peaks slightly but more importantly it defines the peaks better. We also extracted the frequency and phase shift using weighted average method. We also analysed chirped signal at different time ranges and understood their time variation of DFT.