EE2703: Applied Programming Lab

Week 8: Digital Fourier Transform

Author: Surya Prasad S, EE19B121

5th May, 2021

1 Abstract:

This week's assignment deals with analysing Digital Fourier Transform (DFT) using Numpy's Fast Fourier Transform (FFT) library. FFT is an effective and fast implementation of DFT. We shall first focus on periodic signals and also attempt to approximate the Continous Time Fourier Transform (CTFT) of a Gaussian function. We shall also the effect of Time Range and Number of Samples on DFT.

2 Code and Plot Analysis:

2.1 Libraries Imported:

```
import sys
from pylab import *
```

We are importing sys to allow the user to enter the number of samples in a time range (N) which is by default 2048.

2.2 DFT Analysis of sinusoids:

Let's first try to estimate the DFT of a *sine* function.

$$y = sin(kx) = \frac{e^{jkx} - e^{-jkx}}{2j}$$

$$Y(\omega) = \frac{1}{2j}[\delta(\omega - k) - \delta(\omega + k)]$$

We expect an impulse response at k and $-k$. The phase response will be $-\frac{\pi}{2}$ and $\frac{\pi}{2}$ at the two points.

Similarly let's find out the DFT for a *cosine* function.

$$y = cos(kx) = \frac{e^{jkx} + e^{-jkx}}{2}$$

$$Y(\omega) = \frac{1}{2}[\delta(\omega - k) + \delta(\omega + k)]$$

Here too we expect an impulse response at k and $-k$. There will be 0 phase response at all points. So for more complicated periodic signals, we expect the response to include multiple impulses. We shall use this idea to understand for more complicated functions.

2.3 Function for DFT Analysis:

We shall use Numpy's fft module for our DFT Analysis. The output of this function is for the range from 0 to 2π so we need to shift the output to make it available within $|\pi|$. We shall use Numpy's fftshift() for this. Finally we also need to divide the value by N (number of samples) to get the computed DFT value. We shall also plot the magnitude and phase response of the signal.

Code snippet for this function:

```
# Function to analyse given signal's DFT
## The inputs to this function are the function definition, Title to be given
    to the plot, the range of time and the endpoints of the x axis
def plot_analysis(func, TITLE, r, xlimit = 10):
    ## 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)
    t = t[:-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)
    w = w[:-1]

    ## Computing functional values in the time domain and frequency domain
    ### For normal functions we shall be passing the function definition and
        for gaussian alone we shall just pass the string
    if func == 'Gauss':
        y = exp(-t**2/2)
        Y = fftshift(abs(fft(y)))/float(N)

        ### Normalizing for the case of gaussian
        Y = Y * sqrt(2 * pi)/max(Y)
        Y_ = exp(-w**2/2) * sqrt(2 * pi)
        print("Max error for time range as [{} pi, {} pi) is {}".format(round(r
            [0]/pi), round(r[1]/pi), abs(Y - Y_).max()))

    else:
        y = func(t)
        Y = fftshift((fft(y)))/float(N)

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

    subplot(2, 1, 1)
    title(TITLE)
    plot(w, abs(Y), lw = 2)
    xlim([-xlimit, xlimit])
    ylabel(r"Magnitude Response ($|y|$)", 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"Phase Response ($\angle$Y$)", size = 16)
xlabel(r"$k$", size = 16)
grid(True)

show()
```

2.4 Effect of Time range and Number of Samples:

Let's briefly look at the effect of Time range (T_0) and Number of Samples (N) on DFT. Let's first list out all the parameters which are of concern. Sampling frequency is given by:

$$\omega_s = \frac{2\pi N}{T_0}$$

Resolution of DFT is given by:

$$Resolution\ of\ DFT = \frac{T_0}{2\pi}$$

Numpy's fft module computes the DFT for the same number of input points given to it. This implies that the range of frequencies for which DFT is computed is dependent on N.

$$Range\ of\ \omega = \frac{\pi N}{T_0}$$

We should also remember that for plotting purpose we have limited the x axis using $xlimit$. So ideally this should be less than all the frequencies present in the input signal.

Now suppose we fix T_0 and increase N, beyond a certain point there will be no effect as the resolution of the DFT plot is independent of N. It has to be noted that all the significant frequencies present in the given signal should be within $xlimit$. Suppose we plotted for the entire range of ω, then we need the highest frequency of the input signal to be less than the range.

$$\omega_{max} < \frac{\pi N}{T_0}$$
$$\omega_{max} < \frac{\omega_s}{2}$$

This condition is also known as Nyquist's criteria.

Now let's fix N at a high value and change T_0. Here we only have to be concerned by the Resolution of DFT. So to increase the resolution we simply need to increase T_0. If N is not large enough, it may happen that the sampling frequency may become too less.

In the following analysis of signals, number of samples is equal to 2048 to ensure there are no issues in sampling. We shall modify it to see its effects and try to fix our code appropriately.

2.5 Analysis of Periodic Signals

2.5.1 Simple sinusoid

Let's first analyse a simple sinusoidal signal. We shall use the following function definition for analysing $sin(5t)$.

```
## Function definition for sin(5t)
def func_sin5(x):
    return sin(5 * x)
```

$$y = sin(5x) = \frac{e^{j5x} - e^{-j5x}}{2j}$$

$$Y(\omega) = \frac{1}{2j}[\delta(\omega - 5) - \delta(\omega + 5)]$$

So we expect two impulses at 5 and -5 and the phase response to be $\frac{\pi}{2}$ and $-\frac{\pi}{2}$ at the two points.

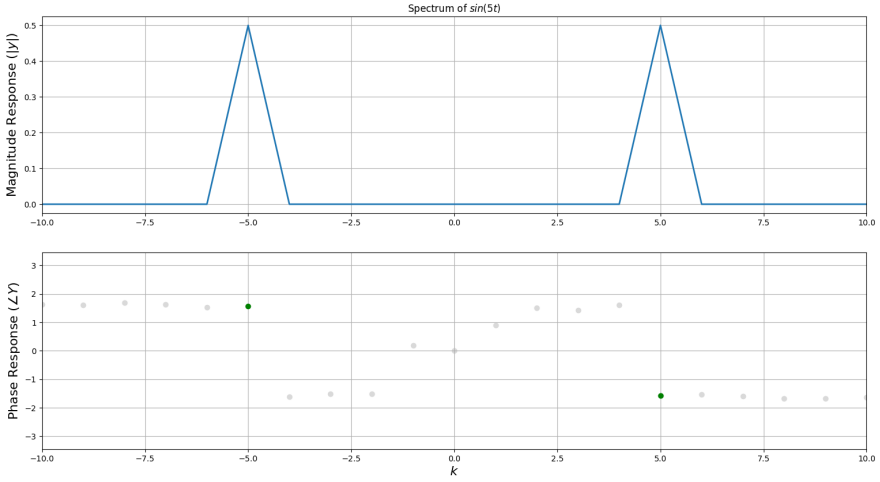

Figure 1: Magnitude and Phase Response of $sin(5t)$

2.5.2 Amplitude Modulated Signal

Now let's look at a more complicated signal such as Amplitude Modulated signal.

$$y = cos(10t)(1 + 0.1cos(t))$$

$$y = \frac{e^{j10t} + e^{-j10t}}{2} + 0.025(e^{j11t} + e^{j9t} + e^{-j11t} + e^{-j9t})$$

Here the signal is actually a combination of three different frequencies (9, 10 and 11) so we expect 6 different impulse responses. Code snippet for the function definition is as follows:

```
## Function definition for Amplitude Modulated signal
def func_AM(x):
    return cos(10 * x) * (1 + 0.1 * cos(x))
```

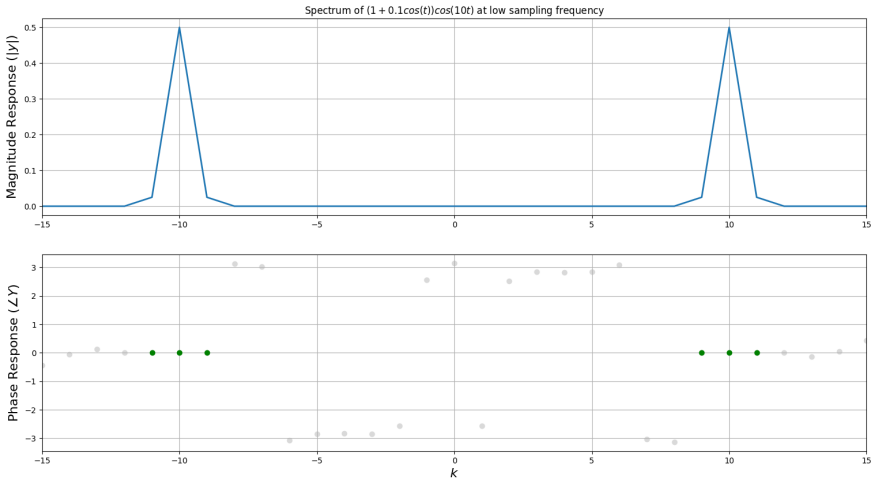

Figure 2: Magnitude and Phase Response of AM signal at low sampling frequency

As we can see from the graph, there are only two peaks instead of six. This is because of small sampling frequency. Let's increase the sampling frequency. We do this by increasing the time range.

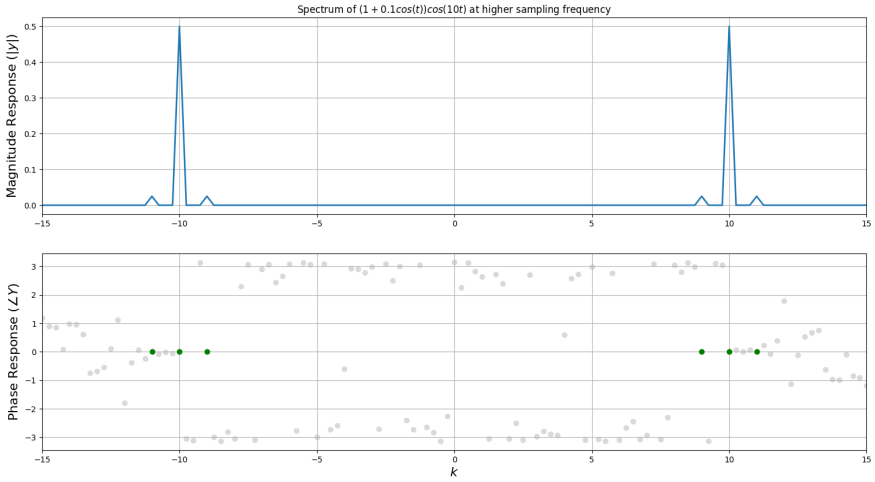

Figure 3: Magnitude and Phase Response of AM signal at high sampling frequency

Now the graph is perfect as we can see the 6 peaks corresponding to different frequencies in the signal. We also notice that the response has become more impulsive with higher sampling frequency.

2.5.3 Cubic Sinusoids

We shall analyse the DFT for sin^3t and cos^3t. The function definitions for them are as follows:

```
## Function definition for (sin(x))^3
def func1(x):
    return (sin(x))**3

## Function definition for (cos(x))^3
def func2(x):
    return (cos(x))**3
```

$$y = sin^3t = \frac{3sin(t) - sin(3t)}{4}$$

$$y = cos^3t = \frac{cos(3t) - 3cos(t)}{4}$$

As we can see, both the functions have the same signal frequencies in their expansion. So we expect 4 impulse responses for both of them. In case of sine function, the phase response corresponding to $\omega = 1$ will have the phase responses $\frac{\pi}{2}$ and $-\frac{\pi}{2}$ and the response will be opposite for $\omega = 3$. There will be 0 phase response in case of cosine function at the same frequencies.

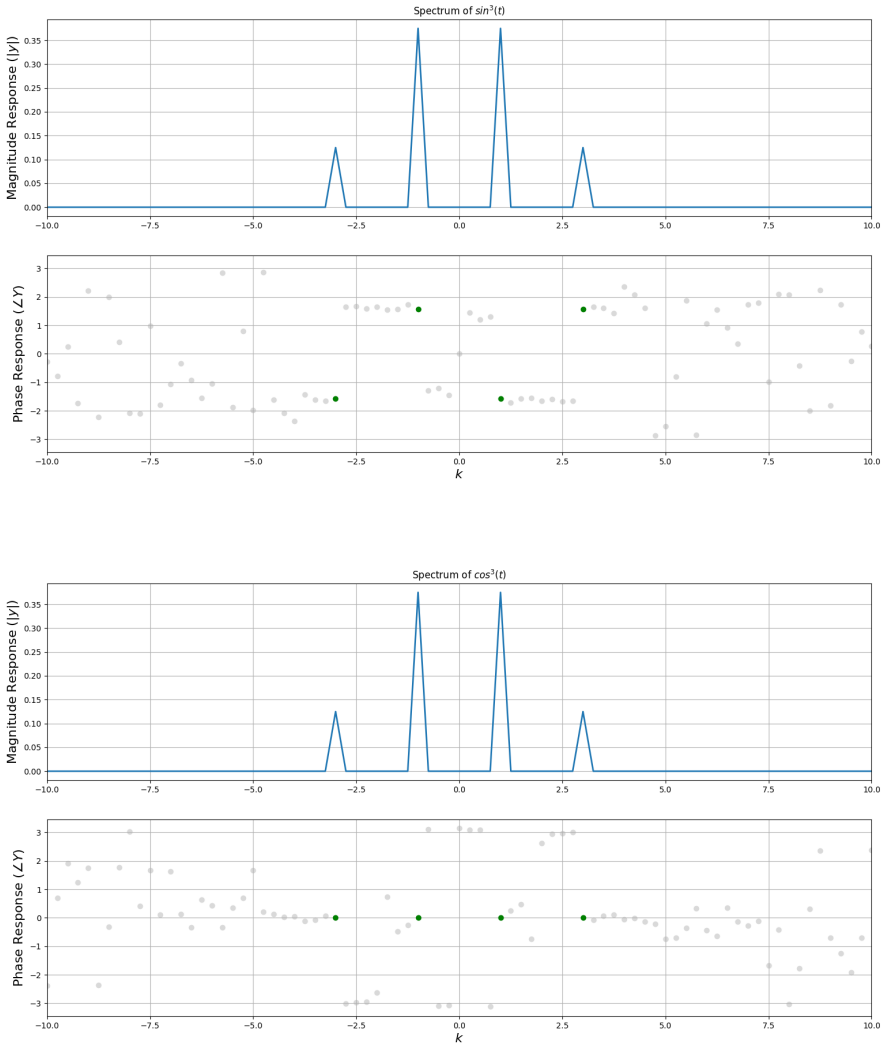

Figure 4: Magnitude and Phase Response of sin^3t and cos^3t

2.5.4 Frequency Modulated Signal

We shall now look at a fairly complicated signal. Let's consider the following frequency modulated signal.

$$y = cos(20t + 5cos(t))$$

Here we can expect the signal to have impulse responses around 20 and -20 which corresponds to $cos(20t)$. The function definition for this is as follows:

```
## Function definition for Frequency Modulated Signal
def func3(x):
    return cos(20 * x + 5 * cos(x))
```

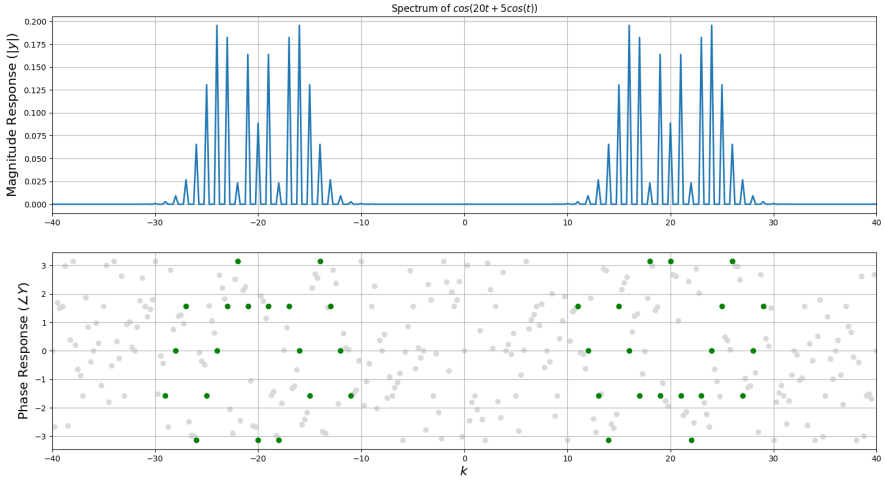

Figure 5: Magnitude and Phase Response of FM signal

As expected all the responses of the system are concentrated around $\omega = 20$ and $\omega = -20$. It is difficult to analyse this signal in the frequency domain because of its complexity. The interesting point to be noted about phase response is that all the responses are either 0, $\frac{\pi}{2}$, $-\frac{\pi}{2}$, π or $-\pi$.

Let's try to plot the same for lesser N (N=128) and same time range ($T_0 = 8\pi$).

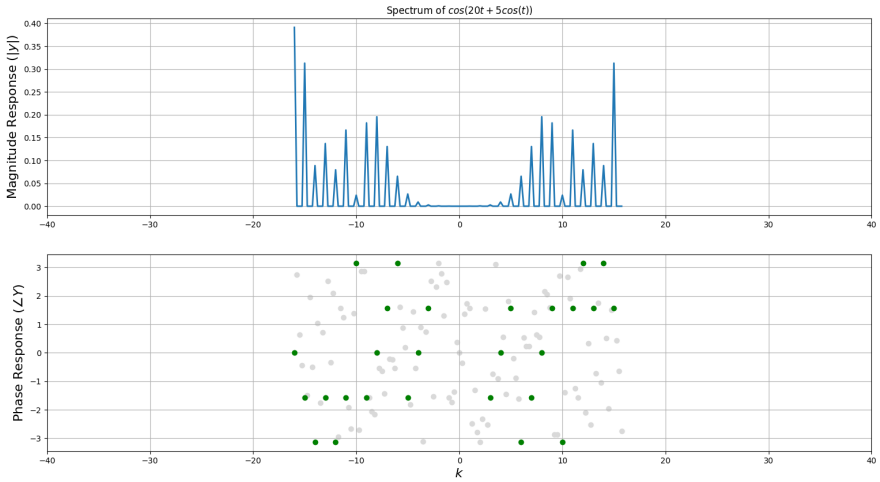

Figure 6: Magnitude and Phase Response of FM signal for low sampling frequency

Here ω_{max} is only 16 but as we saw previously the signal has frequencies upto 30. Hence we need large N to be able to compute the DFT for signals with high frequency components.

2.6 Analysis of the Gaussian function:

Gaussian function is of the following form:

$$f(t) = exp(-\frac{t^2}{2})$$

Gaussian function is not "bandlimited" in frequency. This means that it will have some response to any frequency. We also know that Gaussian function is a self function and so the CTFT of Gaussian function will also be another Gaussian function. The CTFT is given by:

$$F(w) = \sqrt{2\pi}e^{\frac{-w^2}{2}}$$

For comparison purpose, let's approximately take that the CTFT will be equal to DFT. Our aim will be to get an accurate spectrum for the Gaussian function. We need to identify the best time range so that the error is minimum.

The function is symmetric about 0 and so it would be wise to consider a time range which has equal width on both the positive and negative axis. For finding a good time range, we shall use a for loop like in the following code snippet:

```
### Here we pass only the string and identify the ideal sampling rate
for i in range(1, 11):
    plot_analysis("Gauss", r"Spectrum of Gaussian $\exp(-t^2/2)$ for time
        range: [-" + str(i) + r"pi, " + str(i) + r"pi]", [-i * pi, i * pi])
```

In the function plot_analysis, Gauss function has been handled separately. This is because we need to include the phase shift, which is $e^{-j\omega T_0}$, due to different time ranges when we try to find the error in our computation. The actual DFT of the Gauss function can be computed by multiplying the negative of the phase shift to remove it. Mathematically, we know that the CTFT of the Gauss function is real so alternatively we can simply take the absolute value of the DFT of the phase shifted function as the actual DFT. For error analysis we also need to normalise the function.

```
    if func == 'Gauss':
        y = exp(-t**2/2)
        Y = fftshift(abs(fft(y)))/float(N)

        ### Normalizing for the case of gaussian
        Y = Y * sqrt(2 * pi)/max(Y)
        Y_ = exp(-w**2/2) * sqrt(2 * pi)
        print("Max error for time range as [{}pi, {}pi) is {}".format(round(r
            [0]/pi), round(r[1]/pi), abs(Y - Y_).max()))
```

To identify the best time range, we shall use a for loop and print the maximum error.

```
for i in range(1, 11):
    plot_analysis("Gauss", r"Spectrum of Gaussian $\exp(-t^2/2)$ for time
        range: [-" + str(i) + r"pi, " + str(i) + r"pi]", [-i * pi, i * pi])
```

These are the plots we get:

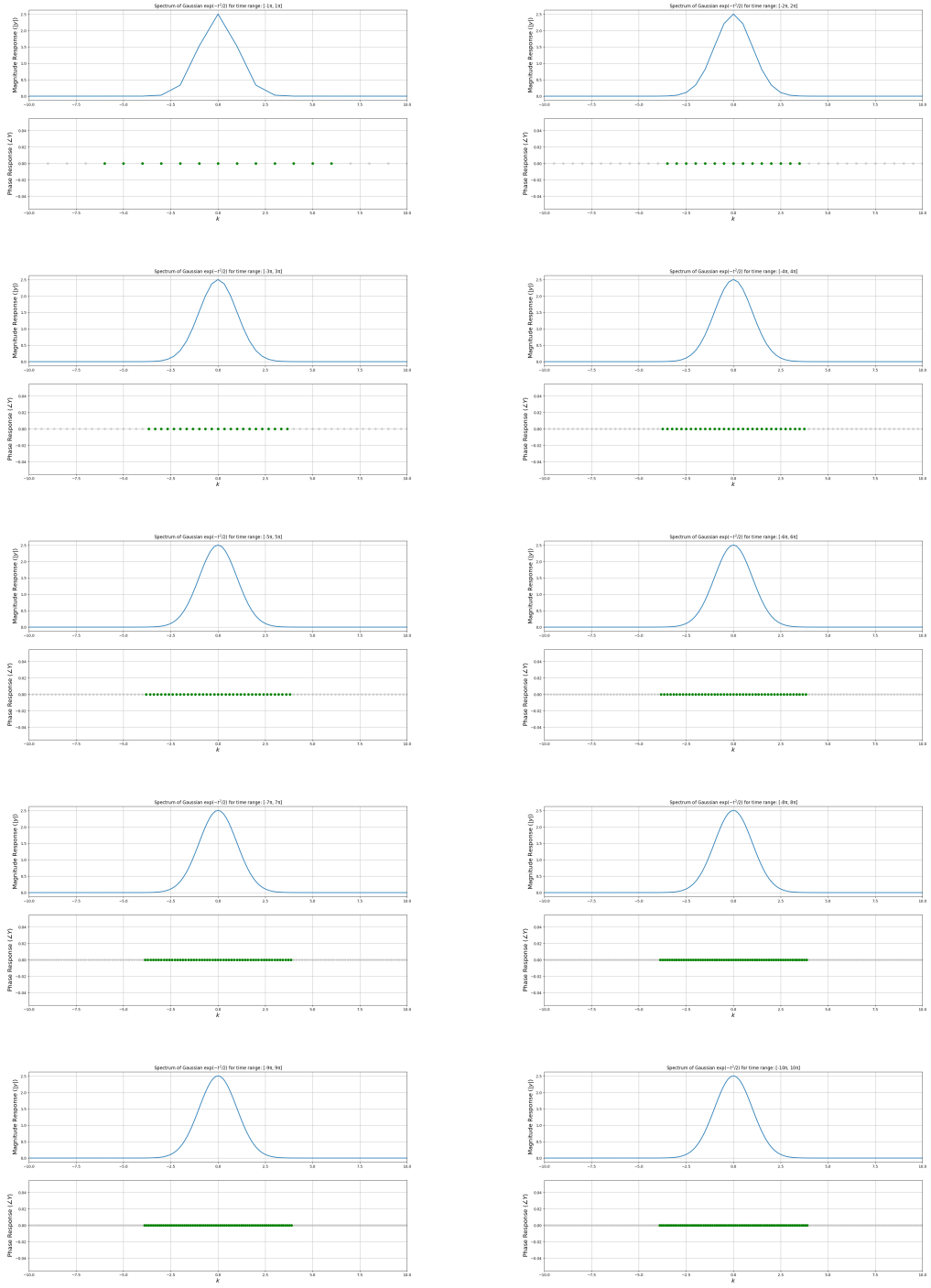

Figure 7: Magnitude and Phase Response of Gaussian function for different time ranges

From the plots we notice that the number of points used for plotting gets denser and the graph becomes smoother. Higher the time range the more likely it is that the graph is close to the exact graph. These are the errors we get for different time ranges:

```
Max error for time range as [-1pi, 1pi) is 0.006496802750855624
Max error for time range as [-2pi, 2pi) is 1.5607475312151564e-09
Max error for time range as [-3pi, 3pi) is 2.5979218776228663e-14
Max error for time range as [-4pi, 4pi) is 8.881784197001252e-16
Max error for time range as [-5pi, 5pi) is 1.6653345369377348e-14
Max error for time range as [-6pi, 6pi) is 1.3988810110276972e-14
Max error for time range as [-7pi, 7pi) is 1.5654144647214707e-14
Max error for time range as [-8pi, 8pi) is 9.921240527779468e-16
Max error for time range as [-9pi, 9pi) is 9.325873406851315e-15
Max error for time range as [-10pi, 10pi) is 8.881784197001252e-15
```

From our set of time ranges we get that the best time range is $[-4\pi, 4\pi)$.

3 Conclusion:

Thus we have analysed the Digital Fourier Transform of simple sinusoids, multiple sinusoids and non-sinusoidal signals. For Gaussian function, we estimated the error in DFT by comparing it against the CTFT of the Gaussian function. We came across the need for shifting the computed DFT which we did by using Numpy's fftshift(). We also analysed the effects of Time range and Number of samples on the DFT and plotted graphs for the same.