EE2703: Assignment 8

Nandha Varman
EE19B043

1 Introduction:

In this assignment, we will analyse DFTs using the `numpy.fft` toolbox.

2 Checking the accuracy of fft and ifft:

To check how powerful and accurate the package is, we shall take the DFT of a sequence of random numbers, then take it's IDFT and compare the two, as to how close they are. This is done using the commands `np.fft.fft` and `np.fft.ifft`. We also have to use `np.fft.fftshift` to shift the $[\pi, 2\pi]$ portion of the DFT to the left as it represents negative frequency, i.e., $[-\pi, 0]$.

For this we perform,

```
# FFT and IFFT on random sequence
x_actual = np.random.rand(128)
X = fft.fft(x_actual)
xComputed = fft.ifft(X)
plt.figure(0)
t = np.linspace(-40, 40, 129)
t = t[:-1]
plt.plot(t, x_actual, 'b', label='Original $x(t)$', lw=2)
plt.plot(t, np.abs(xComputed), 'g', label='Computed $x(t)$', lw=2)
plt.xlabel(r'$t\ \to$')
plt.grid()
plt.legend()
plt.title('Comparison of actual and computed $x(t)$')
maxError = max(np.abs(xComputed-x_actual))
print(r'Magnitude of maximum error between actual and computed values of the random sequence: ',
↪  maxError)     # order of 1e-15
```

The error came out to be of the order of 10^{-15}. When the two sequences `x_actual` and `xComputed` are plotted together, this is the result:

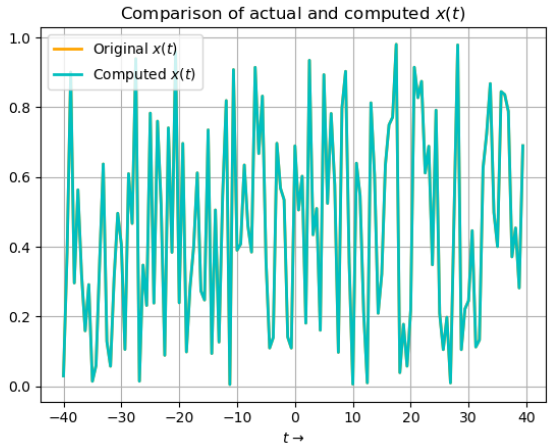

Figure 1: Comparison of the true and computed values of the random sequence

The two sequences overlap. This shows that the `numpy.fft` package is very accurate.

3 Spectrum of $sin(5t)$:

We calculate the DFT of $f(t)$ by the method mentioned in the above section.

```
# Spectrum of sin(5t)
x = np.linspace(0, 2*PI, 129)
x = x[:-1]
y = np.sin(5*x)
Y = fft.fftshift(fft.fft(y))/128.0
fig1 = plt.figure(1)
fig1.suptitle(r'FFT of $sin(5t)$')
YMag = np.abs(Y)
YPhase = np.angle(Y)
presentFreqs = np.where(YMag > 1e-3)
w = np.linspace(-40, 40, 129)
w = w[:-1]
plt.subplot(211)
plt.plot(w, YMag, lw=2)
plt.xlim([-10, 10])
plt.ylabel(r'$\|Y\|$')
plt.grid()
plt.subplot(212)
plt.xlim([-10, 10])
plt.ylim([-np.pi, np.pi])
plt.ylabel(r'$\angle Y$')
plt.xlabel(r'$k\ \to$')
plt.plot(w[presentFreqs], YPhase[presentFreqs], 'ro', lw=2)
plt.grid()
```

Then, we plot the phase and magnitude of the DFT and the following is obtained for $sin(5t)$:

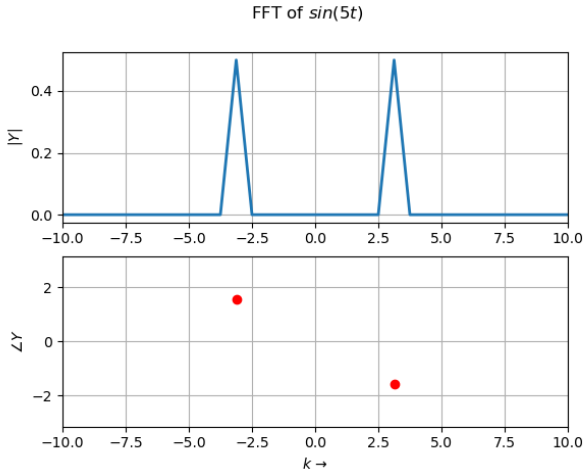

Figure 2: Spectrum of $sin(5t)$

This is expected, because:

$$sin(5t) = \frac{1}{2j}(e^{5jt} - e^{-5jt}) \tag{1}$$

So, the frequencies present in the DFT of $sin(5t)$ are $\omega = \pm 5\ rad/sec$, and the phase associated with them is $\phi = \pm\frac{\pi}{2}\ rad/sec$ respectively. This is exactly what is shown in the above plot.

4 Amplitude Modulation with $(1 + 0.1cos(t))cos(10t)$:

We have,

$$(1 + 0.1cos(t))cos(10t) = \frac{1}{2}(e^{10jt} + e^{-10jt}) + 0.1 \cdot \frac{1}{2} \cdot \frac{1}{2}(e^{11jt} + e^{-11jt} + e^{9jt} + e^{-9jt}) \tag{2}$$

Writing $(1+0.1cos(t))cos(10t)$ in a different form as shown in (2), we observe that the frequencies present in the signal are $\omega = \pm 10\ rad/sec$, $\omega = \pm 11\ rad/sec$ and $\omega = \pm 9\ rad/sec$. Thus we expect the DFT also to have non-zero magnitudes only at these frequencies.

```
# AM Modulation with (1 + 0.1cos(t))cos(10t)
x = np.linspace(-4*PI, 4*PI, 513)
x = x[:-1]
y = (1+0.1*np.cos(x))*np.cos(10*x)
Y = fft.fftshift(fft.fft(y))/512.0
fig2 = plt.figure(2)
fig2.suptitle(r'AM Modulation with $(1+0.1cos(t))cos(10t)$')
YMag = np.abs(Y)
YPhase = np.angle(Y)
presentFreqs = np.where(YMag > 1e-3)
w = np.linspace(-40, 40, 513)
w = w[:-1]
plt.subplot(211)
plt.plot(w, YMag, lw=2)
plt.xlim([-15, 15])
plt.ylabel(r'$\|Y\|$')
plt.grid()
plt.subplot(212)
plt.xlim([-15, 15])
plt.ylim([-np.pi, np.pi])
plt.ylabel(r'$\angle Y$')
plt.xlabel(r'$k\ \to$')
```

```
plt.plot(w[presentFreqs], YPhase[presentFreqs], 'ro', lw=2)
plt.grid()
```

Plotting the DFT using the `numpy.fft` package, we get:

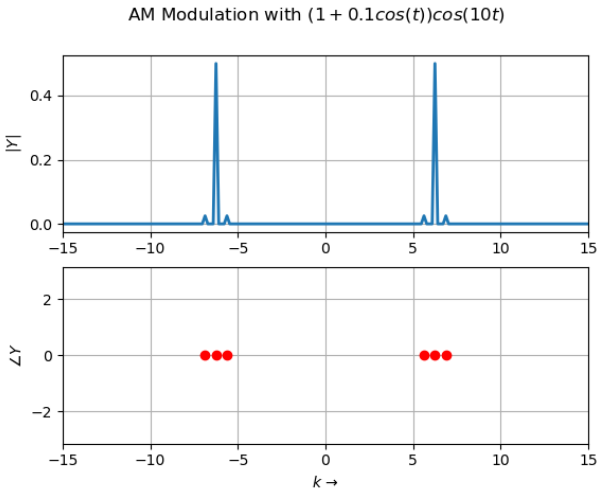

Figure 3: DFT of $(1 + 0.1cos(t))cos(10t)$

5 Spectrum of $sin^3(t)$ and $cos^3(t)$:

DFT Spectrum of $sin^3(t)$:

```
# Spectrum of sin^3(t)
x = np.linspace(-4*PI, 4*PI, 513)
x = x[:-1]
y = (np.sin(x))**3
Y = fft.fftshift(fft.fft(y))/512.0
fig3 = plt.figure(3)
fig3.suptitle(r'Spectrum of $sin^3(t)$')
YMag = np.abs(Y)
YPhase = np.angle(Y)
presentFreqs = np.where(YMag > 1e-3)
w = np.linspace(-40, 40, 513)
w = w[:-1]
plt.subplot(211)
plt.plot(w, YMag, lw=2)
plt.xlim([-5, 5])
plt.ylabel(r'$\|Y\|$')
plt.grid()
plt.subplot(212)
plt.plot(w[presentFreqs], YPhase[presentFreqs], 'ro', lw=2)
plt.xlim([-5, 5])
plt.ylim([-np.pi, np.pi])
plt.ylabel(r'$\angle Y$')
plt.xlabel(r'$k\ \to$')
plt.grid()
```

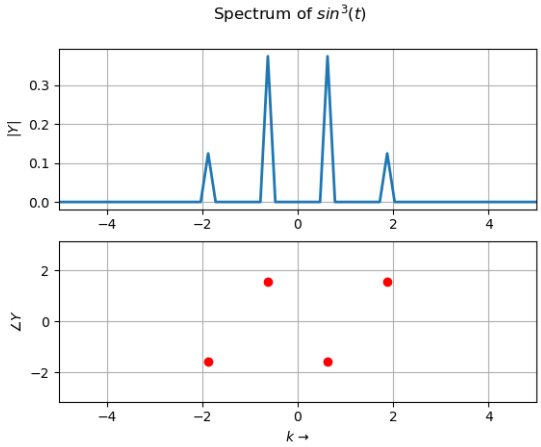

Figure 4: Spectrum of $sin^3(t)$

DFT Spectrum of $cos^3(t)$:

```
# Spectrum of cos^3(t)
x = np.linspace(-4*PI, 4*PI, 513)
x = x[:-1]
y = (np.cos(x))**3
Y = fft.fftshift(fft.fft(y))/512.0
fig4 = plt.figure(4)
fig4.suptitle(r'Spectrum of $cos^3(t)$')
YMag = np.abs(Y)
YPhase = np.angle(Y)
presentFreqs = np.where(YMag > 1e-3)
w = np.linspace(-40, 40, 513)
w = w[:-1]
plt.subplot(211)
plt.plot(w, YMag, lw=2)
plt.xlim([-5, 5])
plt.ylabel(r'$\|Y\|$')
plt.grid()
plt.subplot(212)
plt.plot(w[presentFreqs], YPhase[presentFreqs], 'ro', lw=2)
plt.xlim([-5, 5])
plt.ylim([-np.pi, np.pi])
plt.ylabel(r'$\angle Y$')
plt.xlabel(r'$k\ \to$')
plt.grid()
```

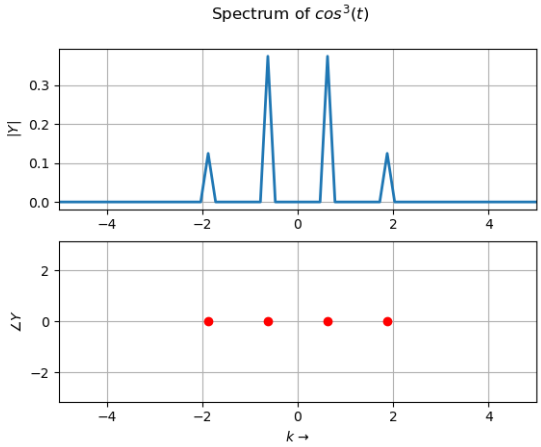

Figure 5: Spectrum of $cos^3(t)$

The above 2 figures are expected because:

$$sin^3(t) = \frac{3}{4}sin(t) - \frac{1}{4}sin(3t) \tag{3}$$

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

So, we expect peaks $\omega = \pm 1\ rad/sec$ and $\omega = \pm 3\ rad/sec$.

6 Frequency Modulation with $cos(20t + 5cos(t))$:

```
# Spectrum of cos(20t + 5cos(t))
x = np.linspace(-4*PI, 4*PI, 513)
x = x[:-1]
y = np.cos(20*x + 5*np.cos(x))
Y = fft.fftshift(fft.fft(y))/512.0
fig5 = plt.figure(5)
fig5.suptitle(r'Spectrum of $cos(20t + 5cos(t))$')
YMag = np.abs(Y)
YPhase = np.angle(Y)
presentFreqs = np.where(YMag > 1e-3)
w = np.linspace(-40, 40, 513)
w = w[:-1]
plt.subplot(211)
plt.plot(w, YMag, lw=2)
plt.xlim([-50, 50])
plt.ylabel(r'$\|Y\|$')
plt.grid()
plt.subplot(212)
plt.plot(w[presentFreqs], YPhase[presentFreqs], 'ro', lw=2)
plt.xlim([-50, 50])
plt.ylim([-np.pi, np.pi])
plt.ylabel(r'$\angle Y$')
plt.xlabel(r'$k\ \to$')
plt.grid()
```

The DFT of $cos(20t + 5cos(t))$ can be seen below:

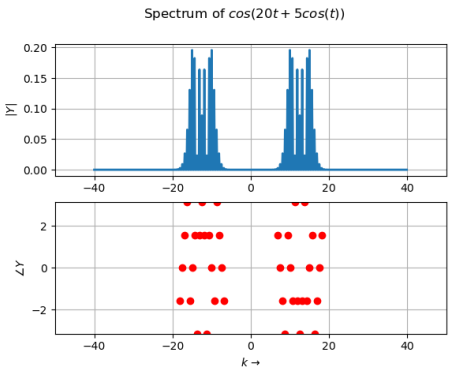

Figure 6: DFT of $cos(20t + 5cos(t))$

When we compare this result with that of the Amplitude Modulation as seen in Fig (3), we see that there are more side bands, and some of them have even higher energy than $\omega = \pm 20\ rad/sec$.

7 DFT of a Gaussian:

```
# Spectrum of Gaussian:

# Phase and Magnitude of estimated Gaussian Spectrum
                #considering a window from [-8pi, 8pi] with 512 points

t =  np.linspace(-8*PI, 8*PI, 513)
t = t[:-1]
xTrueGaussian = np.exp(-(t**2)/2)
Y = fft.fftshift(fft.fft(fft.ifftshift(xTrueGaussian)))*8/512.0
fig6 = plt.figure(6)
fig6.suptitle(r'Spectrum of Gaussian')
YMag = np.abs(Y)
YPhase = np.angle(Y)
absentFreqs = np.where(YMag < 1e-3)
YPhase[absentFreqs] = 0
w = np.linspace(-40, 40, 513)
w = w[:-1]
plt.subplot(221)
plt.plot(w, YMag, lw=2)
plt.xlim([-10, 10])
plt.ylabel(r'$\|Y\|$')
plt.title("Estimated")
plt.grid()
plt.subplot(223)
plt.plot(w, YPhase, 'ro', lw=2)
plt.xlim([-10, 10])
plt.ylim([-np.pi, np.pi])
plt.ylabel(r'$\angle Y$')
plt.xlabel(r'$k\ \to$')
plt.grid()

# Phase and Magnitude of true Gaussian spectrum
trueY = np.exp(-(w**2)/2)/np.sqrt(2*PI)
trueYMag = np.abs(trueY)
trueYPhase = np.angle(trueY)
plt.subplot(222)
```

```
plt.plot(w, trueYMag)
plt.xlim([-10, 10])
plt.title("True")
plt.grid()
plt.subplot(224)
plt.plot(w, trueYPhase, 'ro')
plt.xlim([-10, 10])
plt.ylim([-np.pi, np.pi])
plt.xlabel(r'$k\ \to$')
plt.grid()

meanError = np.mean(np.abs(trueY - Y))
print(r'Magnitude of mean error between actual and computed values of the Gaussian: ', meanError)
↪  #


plt.show()
```

The DFT of a gaussian is also a gaussian, as shown below:

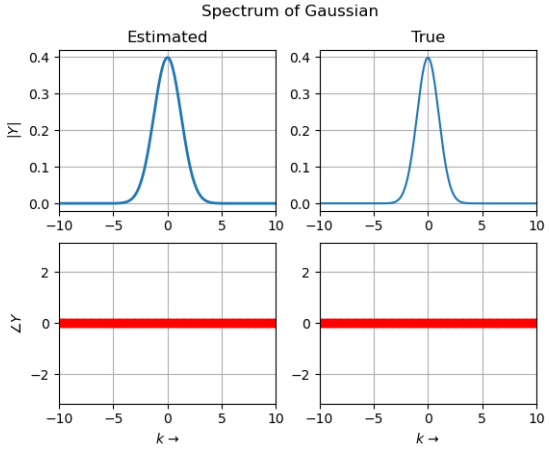

Figure 7: Gaussian Spectrum

8 Conclusion:

We have thus found out the DFT's of various signals using the `numpy.fft` package. We used the `numpy.fft.fftshift()` and `numpy.fft.ifftshift()` methods to fix distortions in the phase response.