Assignment No 8

Shivanshu Shekhar

April 14, 2022

1 Introduction

- We learnt to analyze signals in frequency domain.
- We learnt about the `fft` module of `numpy`.
- We approximated the CTFT(Continuous time Fourier Transform) of a Gaussian using FFT(Fast Fourier transform).

2 Problems and Solutions

2.1 Question 1

2.1.1 Random data

We find the Fourier Transform of a random signal and then we find the inverse Fourier Transform of the function, we then finally compare those values with the actual values generated, we find that the max error we get is of the order of -16.

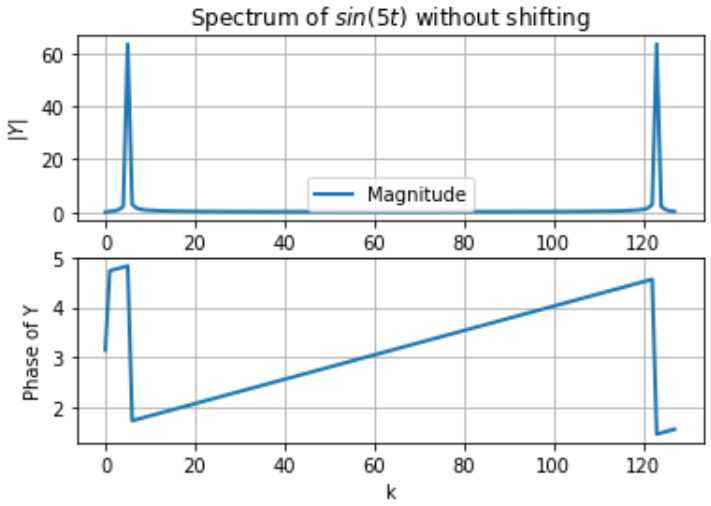
```
x = np.random.rand(100)
X = fft.fft(x)
y = fft.ifft(X)
print(abs(x-y).max())
```

2.1.2 Spectrum of $\sin 5t$

Next we wanted to find the spectrum of $\sin 5t$:

```
x = np.linspace(0, 2*np.pi, 128)
y = np.sin(5*x)
Y = fft.fft(y)
plt.figure()
plt.subplot(2,1,1)
plt.title(r"Spectrum of $sin(5t)$ without shifting")
plt.plot(abs(Y), lw = 2, label = "Magnitude")
```

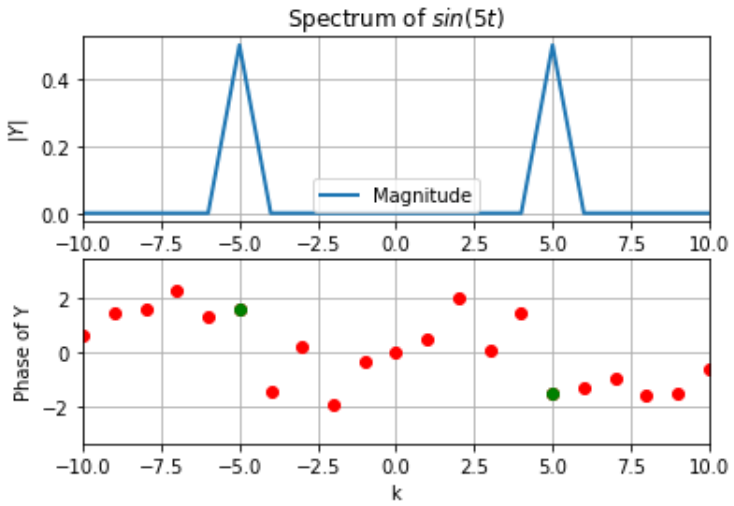
```
plt.xlabel("k");plt.ylabel(r"$|Y|$")
plt.grid();plt.legend()
plt.subplot(2,1,2)
plt.xlabel("k");plt.ylabel("Phase of Y")
plt.plot(np.unwrap(np.angle(Y)), lw = 2)
plt.grid()
plt.show()
```



We need to shift the phase plot so that it goes from $-\pi$ to π, as 0 and $2pi$ means the same thing.

```
x = np.linspace(0, 2*np.pi, 128, endpoint = False)
y = np.sin(5*x)
Y = fft.fftshift(fft.fft(y))/128
w = np.linspace(-64,63,128)
plt.figure()
plt.subplot(2,1,1)
plt.title(r"Spectrum of $sin(5t)$")
plt.plot(w,abs(Y), lw = 2, label = "Magnitude")
plt.xlim([-10,10])
plt.xlabel("k");plt.ylabel(r"$|Y|$")
plt.grid();plt.legend()
plt.subplot(2,1,2)
ii = np.where(abs(Y)>1e-3)
plt.xlabel("k");plt.ylabel("Phase of Y")
plt.plot(w, np.angle(Y), "ro", lw = 2)
plt.plot(w[ii], np.angle(Y[ii]), "go", lw = 2)
plt.xlim([-10,10])
```

```
plt.grid()
plt.show()
```



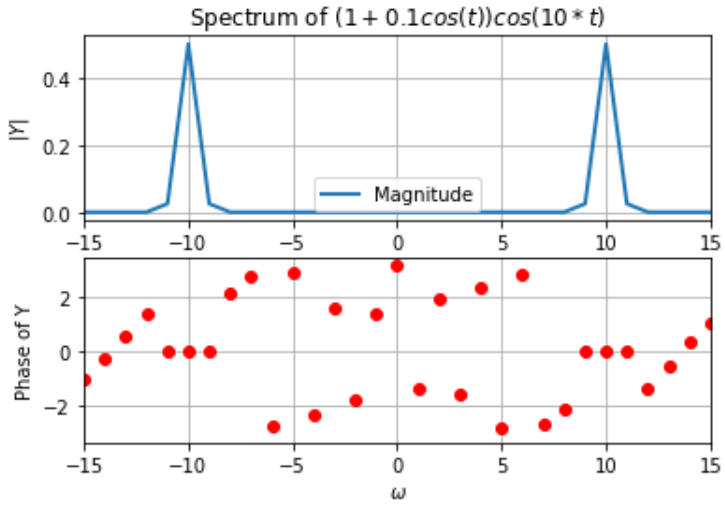
2.1.3 Spectrum of $(1 + 0.1\cos(t))\cos(10t)$

We considered the following signal for analysis:

$$f(t) = (1 + 0.1\cos(t))\cos(10t)$$

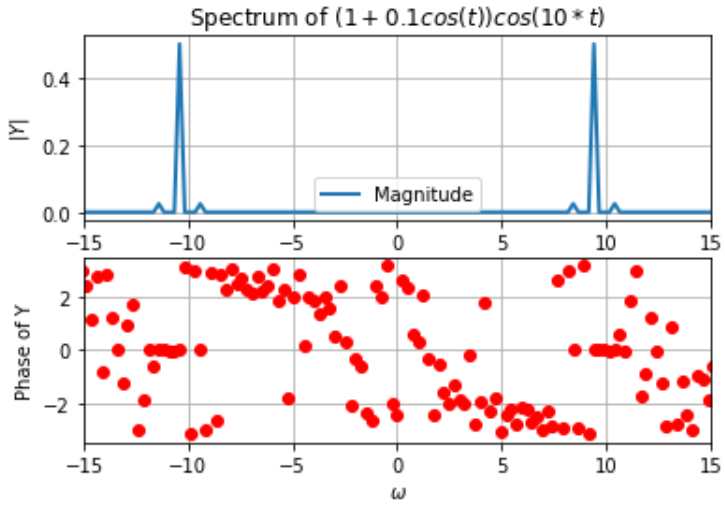
We accomplished this using:

```
t = np.linspace(0, 2*np.pi, 128, endpoint = False)
y = (1+0.1*np.cos(t))*np.cos(10*t)
Y = fft.fftshift(fft.fft(y))/128
w = np.linspace(-64,63,128)
plt.figure()
plt.subplot(2,1,1)
plt.title(r"Spectrum of $(1+0.1cos(t))cos(10*t)$")
plt.plot(w,abs(Y), lw = 2, label = "Magnitude")
plt.xlim([-15,15])
plt.ylabel(r"$|Y|$")
plt.grid();plt.legend()
plt.subplot(2,1,2)
plt.xlabel(r"$\omega$");plt.ylabel("Phase of Y")
plt.plot(w, np.angle(Y), "ro", lw = 2)
plt.xlim([-15,15])
plt.grid()
plt.show()
```



The number of sample considered is small to realize all the peaks so we plot again using more number of sample points.

```
t = np.linspace(-4*np.pi, 4*np.pi, 512, endpoint = False)
y = (1+0.1*np.cos(t))*np.cos(10*t)
Y = fft.fftshift(fft.fft(y))/512
w = np.linspace(-64,63,512, False)
plt.figure()
plt.subplot(2,1,1)
plt.title(r"Spectrum of $(1+0.1cos(t))cos(10*t)$")
plt.plot(w,abs(Y), lw = 2, label = "Magnitude")
plt.xlim([-15,15])
plt.ylabel(r"$|Y|$")
plt.grid();plt.legend()
plt.subplot(2,1,2)
plt.xlabel(r"$\omega$");plt.ylabel("Phase of Y")
plt.plot(w, np.angle(Y), "ro", lw = 2)
plt.xlim([-15,15])
plt.grid()
plt.show()
```



2.2 Question 2

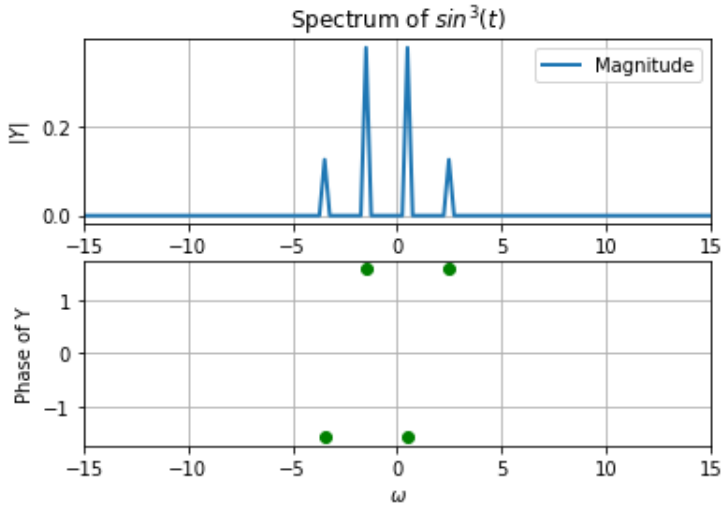
2.2.1 Spectrum of $\sin^3(t)$

We can write $\sin^3(t)$ as:

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

So we should get 2 peaks at 1 and 3 with —phase— being $\frac{\pi}{2}$.

```
t = np.linspace(-4*np.pi, 4*np.pi, 512, endpoint = False)
y = np.sin(t)**3
Y = fft.fftshift(fft.fft(y))/512
w = np.linspace(-64,63,512, False)
plt.figure()
plt.subplot(2,1,1)
plt.title(r"Spectrum of $sin^3(t)$")
plt.plot(w,abs(Y), lw = 2, label = "Magnitude")
plt.xlim([-15,15])
plt.ylabel(r"$|Y|$")
plt.grid(); plt.legend()
plt.subplot(2,1,2)
plt.xlabel(r"$\omega$"); plt.ylabel("Phase of Y")
ii=np.where(abs(Y)>1e-3)
plt.plot(w[ii], np.angle(Y[ii]), "go", lw = 2)
plt.xlim([-15,15])
plt.grid()
plt.show()
```



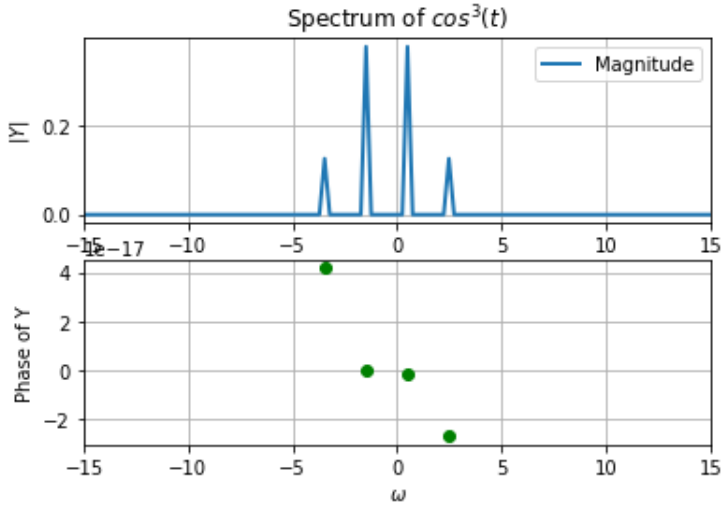
2.2.2 Spectrum of $\cos^3(t)$

We can write $\cos^3(t)$ as:

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

So we should get 2 peaks at 1 and 3 with —phase— being 0.

```
t = np.linspace(-4*np.pi, 4*np.pi, 512, endpoint = False)
y = np.cos(t)**3
Y = fft.fftshift(fft.fft(y))/512
w = np.linspace(-64,63,512, False)
plt.figure()
plt.subplot(2,1,1)
plt.title(r"Spectrum of $cos^3(t)$")
plt.plot(w,abs(Y), lw = 2, label = "Magnitude")
plt.xlim([-15,15])
plt.ylabel(r"$|Y|$")
plt.grid();plt.legend()
plt.subplot(2,1,2)
plt.xlabel(r"$\omega$");plt.ylabel("Phase of Y")
ii=np.where(abs(Y)>1e-3)
plt.plot(w[ii], np.angle(Y[ii]), "go", lw = 2)
plt.xlim([-15,15])
plt.grid()
plt.show()
```

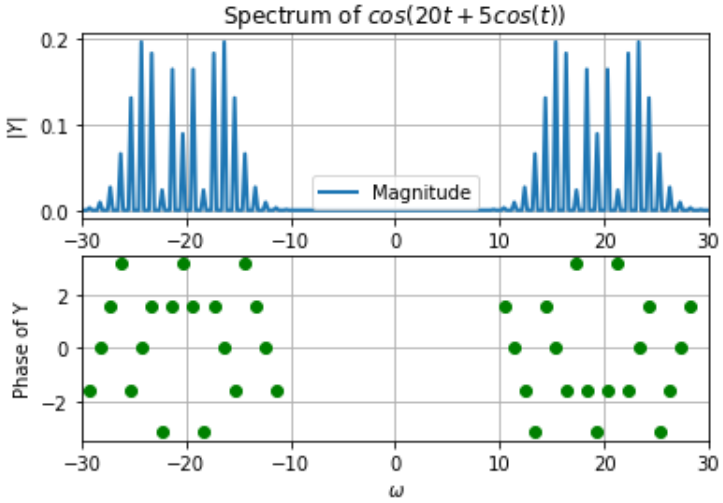


2.3 Question 3

We find the spectrum of the following function:

$$f(t) = \cos(20t + 5\cos(t))$$

```
t = np.linspace(-4*np.pi, 4*np.pi, 512, endpoint = False)
y = np.cos(20*t + 5*np.cos(t))
Y = fft.fftshift(fft.fft(y))/512
w = np.linspace(-64,63,512, False)
plt.figure()
plt.subplot(2,1,1)
plt.title(r"Spectrum of $cos(20t + 5cos(t))$")
plt.plot(w,abs(Y), lw = 2, label = "Magnitude")
plt.xlim([-30,30])
plt.ylabel(r"$|Y|$")
plt.grid(); plt.legend()
plt.subplot(2,1,2)
plt.xlabel(r"$\omega$"); plt.ylabel("Phase of Y")
ii=np.where(abs(Y)>1e-3)
plt.plot(w[ii], np.angle(Y[ii]), "go", lw = 2)
plt.xlim([-30,30])
plt.grid()
plt.show()
```



It is obvious from the plots that no longer a single peak carries all the energy and that the energy is distributed between many peaks hence the signal is modulated.

2.4 Question 4

The Fourier Transform of a signal is defined as:

$$X(\omega) = \frac{1}{2\pi} \int_{-\infty}^{\infty} x(t) e^{-j\omega t} dt$$

We can approximate the Gaussian using a sufficiently large window as the Gaussian tends to zero for large values of x, so we can approximate the integral for a window size of T(Let) as:

$$X(\omega) = \frac{1}{2\pi} \int_{-T/2}^{T/2} x(t) e^{-j\omega t} dt$$

Writing the integral as a Riemann summation of N terms, we get:

$$X(\omega) \approx \frac{T}{2\pi N} \sum_{n=-N/2}^{N/2-1} x(nT/N) e^{-j\omega nT/N}$$

Where T/N is the time step. Then, let $\omega = 2\pi k/T$:

$$X(2\pi k/T) \approx \frac{T}{2\pi N} \sum_{n=-N/2}^{N/2-1} x(nT/N) e^{-j2\pi kn/N}$$

We see that the summation is the DFT(Discrete Fourier Transform) of the signal. Thus, we get:

$$X(2\pi k/T) \approx \frac{T}{2\pi N} DFT\{x(nT/N)\}$$

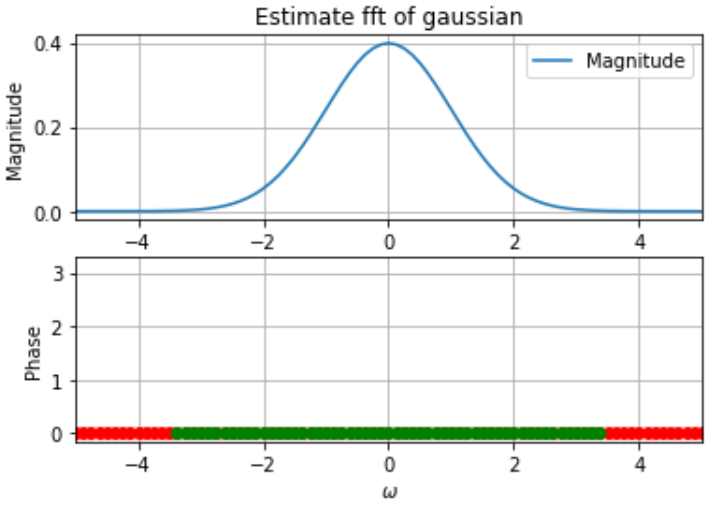
We can improve the accuracy of our obtained approximation by choosing a larger window size while keeping the sampling frequency constant. We do this iteratively until our error is below a certain threshold.

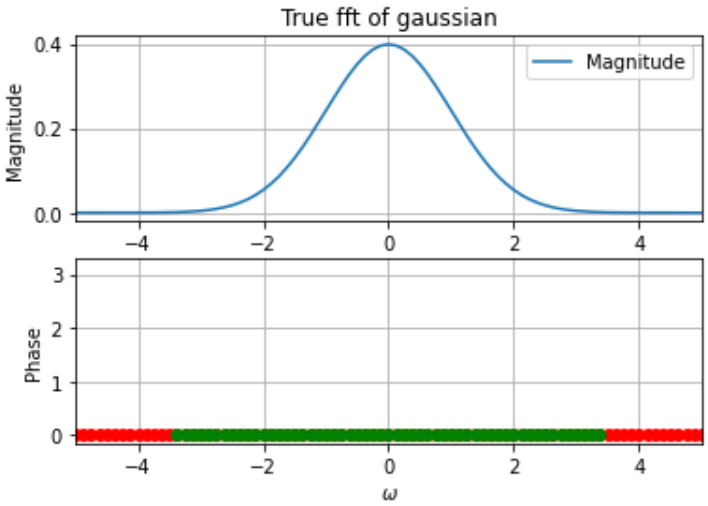
We compare our approximation to the actual CTFT(Continuous Time Fourier Transform) of the Gaussian:

$$\mathscr{F}(e^{-\frac{t^2}{2}}) = \frac{1}{\sqrt{2\pi}} e^{-\frac{\omega^2}{2}}$$

```
Yold = 0; iters=0;N=128;T=8*np.pi
err = 1e-6+1
while err>1e-6:
    x = np.linspace(-T/2, T/2, N, False)
    w = np.linspace(-N*np.pi/T, N*np.pi/T, N, False)
    y = gauss(x)
    Y=fft.fftshift(fft.fft(fft.ifftshift(y)))*T/(2*np.pi*N)
    err = sum(abs(Y[::2] - Yold))
    Yold = Y
    iters+=1
    T*=2
    N*=2

true_err = sum(abs(Y-expectedfn(w)))
print(f"True Error = {true_err}")

mag = abs(Y)
phi = np.angle(Y)
phi[np.where(mag<1e-6)]=0
plt.figure()
plt.subplot(2,1,1)
plt.plot(w,abs(Y), label = "Magnitude")
plt.xlim([-5,5])
plt.ylabel('Magnitude')
plt.title("Estimate fft of gaussian")
plt.grid(); plt.legend()
plt.subplot(2,1,2)
plt.plot(w,np.angle(Y),'ro')
ii=np.where(abs(Y)>1e-3)
```

```
plt.plot(w[ii],np.angle(Y[ii]),'go')
plt.xlim([-5,5])
plt.ylabel("Phase")
plt.xlabel(r"$\omega$")
plt.grid()
plt.show()

# Plotting expected output

Y_ = expectedfn(w)

mag = abs(Y_)
phi = np.angle(Y_)
phi[np.where(mag<1e-6)]=0
plt.figure()
plt.subplot(2,1,1)
plt.plot(w,abs(Y), label = "Magnitude")
plt.xlim([-5,5])
plt.ylabel('Magnitude')
plt.title("True fft of gaussian")
plt.grid();plt.legend()
plt.subplot(2,1,2)
plt.plot(w,np.angle(Y),'ro')
ii=np.where(abs(Y)>1e-3)
plt.plot(w[ii],np.angle(Y[ii]),'go')
plt.xlim([-5,5])
plt.xlabel(r"$\omega$")
plt.ylabel("Phase")
plt.grid()
plt.show()
```





3 Conclusion

We explored the `numpy.fft` module by using it to find the Fourier and Inverse Fourier Transforms of various signals, we also approximated the CTFT of Gaussian using DFT with appropriate window size.