Assignment No 9:The Discrete Fourier Transform

ee20b074,rahul

Introduction

This week's assignment is based on Discrete Fourier Transform (DFT) of signals. We examine the DFT of various functions using the fft library in numpy. We also attempt to approximate the CTFT of a gaussian by changing the window size and number of samples until the error is below a threshold.

FFT and IFFT

There are two commands in Python, one to compute the forward fourier transform and the other to compute the inverse transform. They are np.fft.fft and np.fft.ifft respectively.
We find the Fourier transform and invert it back to the time domain for a random signal, find maximum error to test the reconstruction.

```
x=np.random.rand(100)
X=np.fft.fft(x)
y=np.fft.ifft(X)
print ("Absolute Maximum Error = ",abs(x-y).max())
```

Doing this , we get the output:

```
Absolute Maximum Error =  3.413079373551675e-16
```

Printing out x and y side-by-side, we observe that the vectors are not exactly the same. x is purely real whereas y is slightly complex. This is due to the finite accuracy of the CPU so that the ifft could not exactly undo what fft did.

Spectrum of sin(5t)

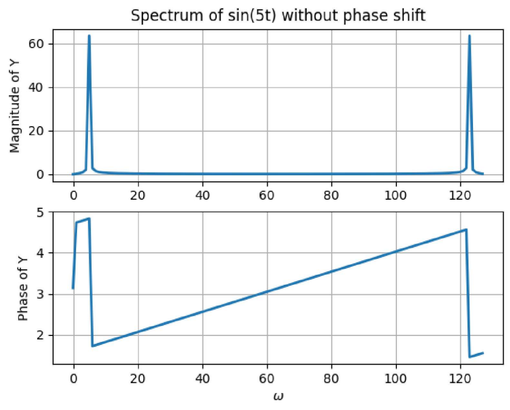
Here, we try to estimate the DFT of a sinusoidal function. For example,

$$y = sin(x) = (exp(jx) - exp(-jx))/2j$$

The expected spectrum is,

$$Y = (\delta(w-1) - \delta(w+1))/2j$$

In a similar process , we try to plot the spectrum of sin(5t).

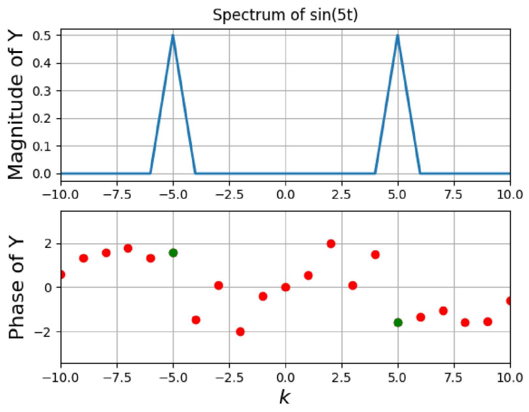


But, there are multiple issues to be fixed here:

- The peaks are not where we expect them to be. This can be corrected using fft shift.
- The spikes have a height of 64, not 0.5. This should be rectified by dividing by the sample rate.
- The frequency axis is not in place. This is due to the duplicacy of 0 and 2π.
- The actual phase at the spikes is near but not exactly correct.

After doing these modifications , we get a much neater spectrum:

```
x=np.linspace(0,2*pi,129);x=x[:-1]
y=np.sin(5*x)
Y=np.fft.fftshift(np.fft.fft(y))/128.0
plt.figure()
plt.subplot(2,1,1)
plt.plot(w,abs(Y),lw=2)
plt.xlim([-10,10])
plt.subplot(2,1,2)
plt.plot(w,np.angle(Y),'ro',lw=2)
ii=np.where(abs(Y)>1e-3)
plt.plot(w[ii],np.angle(Y[ii]),'go',lw=2)
plt.xlim([-10,10])
```

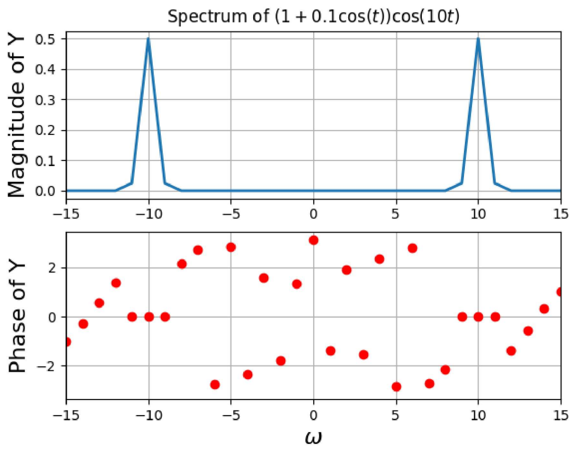


Spectrum of (1+0.1cos(t))cos(10t)

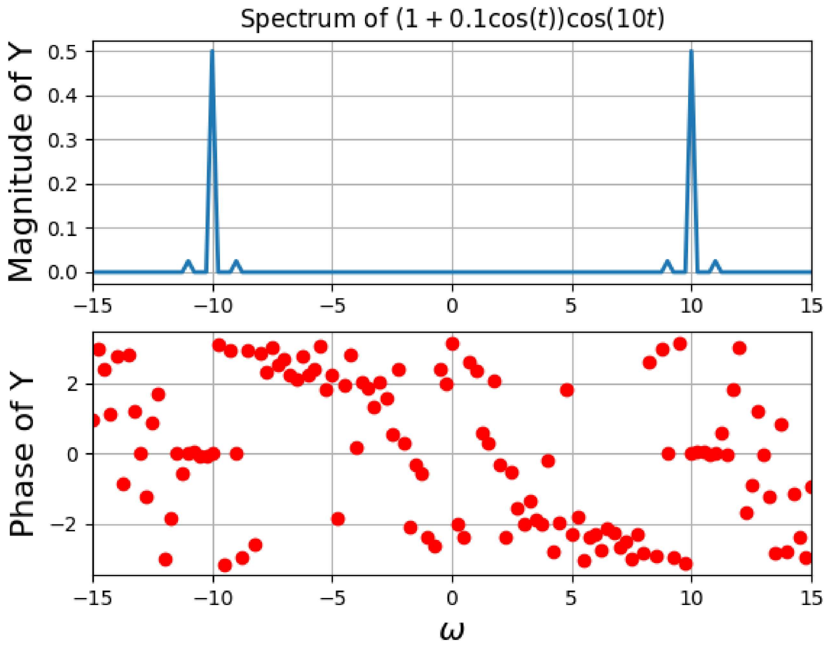
We consider the amplitude modulated spectrum:
y=(1+0.1cos(t))cos(10t)
We expect a shifted set of spikes, with a main impulse and two side impulses on each side. This is because,

$$0.1cos(10t)cos(t) = 0.05(cos11t+cos9t) = 0.025(exp(9jt)+exp(11jt)+exp(-9jt)+exp(-11jt))$$

Again, we use 128 points and generate the spectrum.



We see only a single broader spike. In order to see even the side peaks, the frequency resolution has to be improved. We can do so by keeping the number of samples constant and increasing the range in the time domain. The following spectrum is obtained:



At all 3 peaks, the phase is 0 as expected for a cosine wave.

Assignment Questions

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

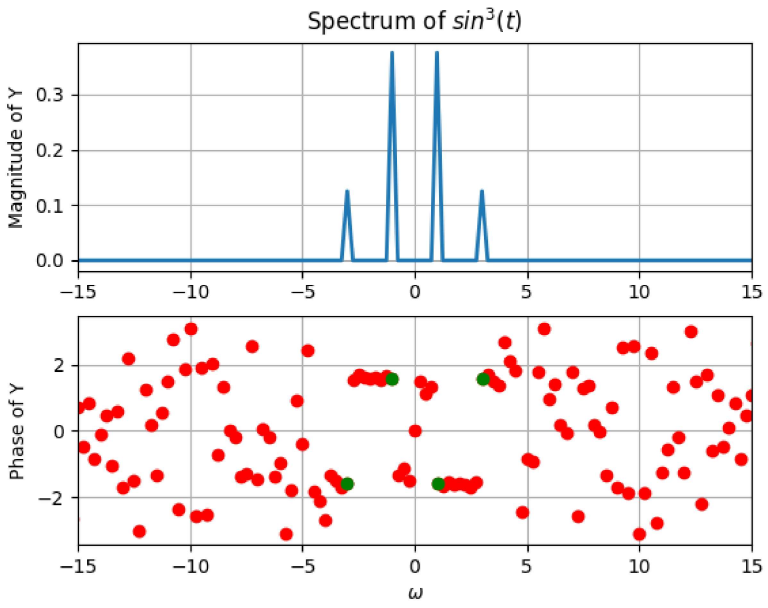
Consider the sinusoids $sin^3(t)$ and $cos^3(t)$:

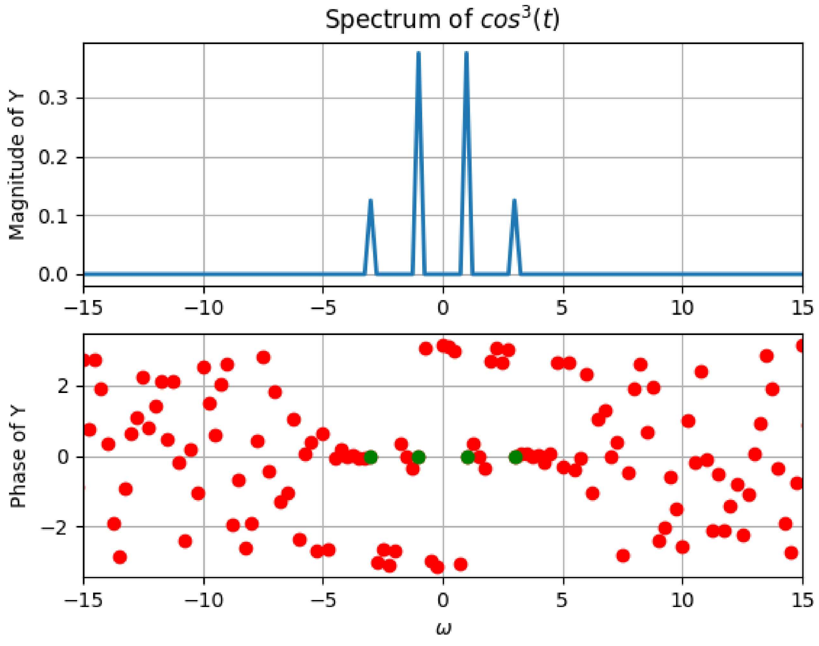
$$sin^3(t) = (3sin(t) - sin(3t))/4$$

$$cos^3(t) = (cos(3t) + 3cos(t))/4$$

Thus, in the frequency spectrum, there will be 4 impulses at frequencies 1 and 3.
Taking the same precautionary measures as done previously, we obtain the following approximate spectrum :

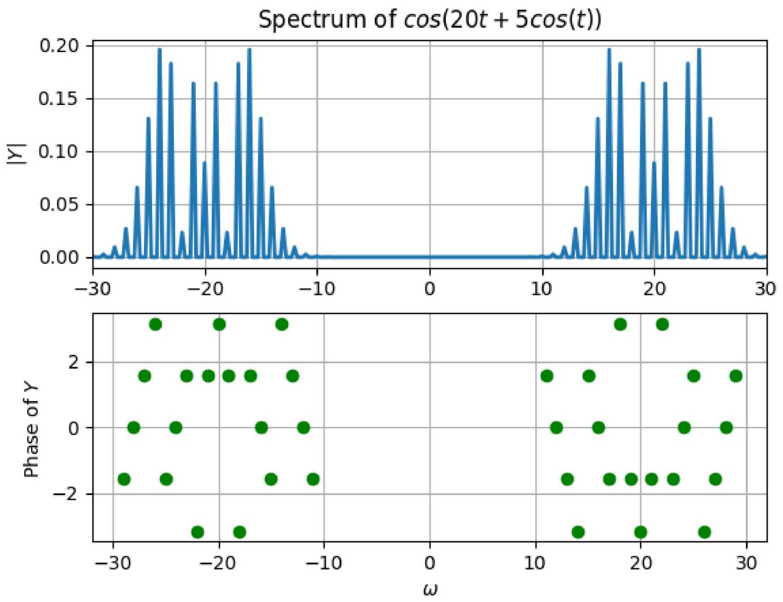
We observe that:
For $sin^3(t)$, the phase is similar to that of sum of 2 sinusoids.
For $cos^3(t)$, phase =0 at the peaks


Spectrum of $sin^3(t)$
Magnitude of Y
Phase of Y
ω


Spectrum of $cos^3(t)$
Magnitude of Y
Phase of Y
ω

Spectrum of frequency modulated wave

Consider the signal:
f(t) = cos(20t + 5cos(t))
Because the argument of the cosine function itself consists of a cosine term, the wave can be expressed using Bessel functions. Thus, the frequency spectrum consists of impulses at the carrier frequency ($w_c = 20$ here) and impulses at multiple side band frequencies, that are close to the carrier frequency.
Following the process same as above , we get:



Spectrum of the Gaussian function

The Gaussian function f(x) = $exp(-x^2/2)$ is not band-limited,in the sense that the frequency spectrum has non zero values even for very large frequencies. The Continuous Time Fourier Transform for the Gaussian is given by:

$$F(jw) = \sqrt(2\pi)exp(-w^2/2)$$

Thus, the phase is zero for all w while the magnitude is a Gaussian function. We can make approximations by making the window size T larger, and by decreasing the time domain sampling period or increasing the number of samples N . We find the appropriate values for these iterative keeping the sampling frequency constant.

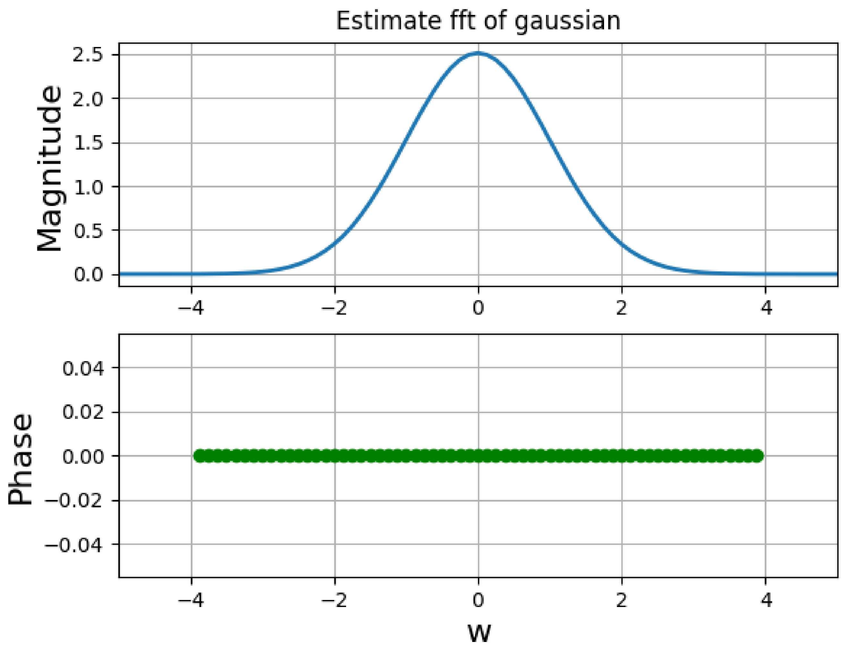
```
wlim=5
tolerance=1e-6
T = 8*pi
N = 128
Yold=0
err=1+tolerance
iters = 0
error=[]

#To find the optimum time range to get an accurate frequency domain.

while err>tolerance:
x = np.linspace(-T/2,T/2,N+1)[:-1]
w = np.linspace(-N*pi/T,N*pi/T,N+1)[:-1]
y = gauss(x)
Y=np.fft.fftshift(np.fft.fft(np.fft.ifftshift(y)))*T/N
Y_exp=expectedgauss(w)
err = np.max(np.abs(Y[::2]-Yold))
error.append(np.max(np.abs(Y-Y_exp)))
Yold=Y
iters+=1
T*=2
N*=2

error = np.max(error)
print("True Error: ",error)
print("Samples, N = "+str(N)+"\n"+"Time Period,T = "+str(int(T/pi))+"pi")

#Estimate DFT of gaussian function. Plot its magnitude and phase.

mag = np.abs(Y)
phi = np.angle(Y)

plt.figure()
plt.subplot(2,1,1)
plt.plot(w,mag,lw=2)
plt.xlim([-wlim,wlim])
plt.subplot(2,1,2)
ii=np.where(mag>1e-3)
plt.plot(w[ii],phi[ii],'go',lw=2)
plt.show()

#Expected DFT of Gaussian is a Gaussian in w. Hence, we plot the magnitude and phase
```

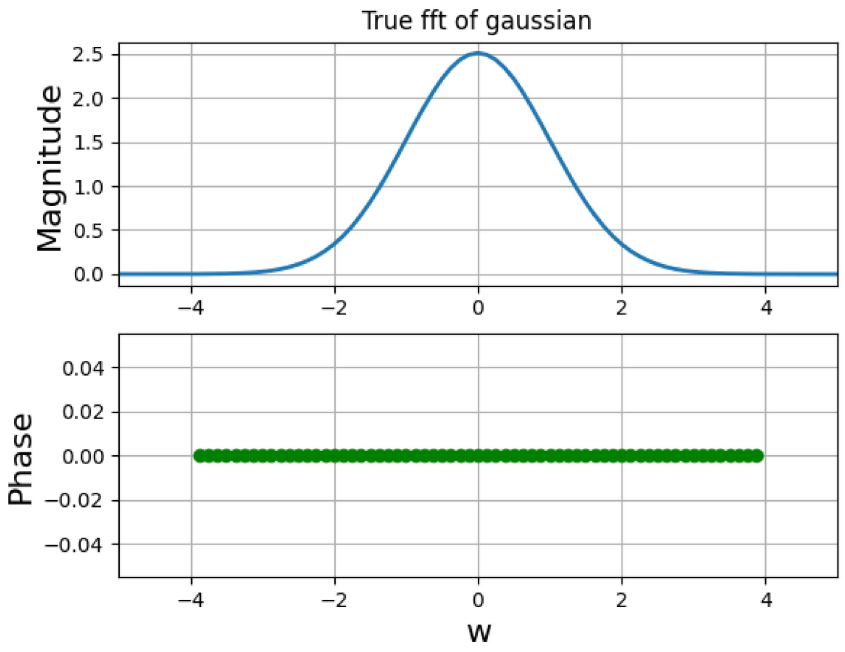
```
Y_exp = expectedgauss(w)
mag_exp = np.abs(Y_exp)
phi_exp = np.angle(Y_exp)

plt.figure()
plt.subplot(2,1,1)
plt.plot(w,mag_exp,lw=2)
plt.xlim([-wlim,wlim])
plt.subplot(2,1,2)
ii=np.where(mag_exp>1e-3)
plt.plot(w[ii],phi_exp[ii],'go',lw=2)
plt.show()
```

After the iteration, we find out the values of the number of samples and the Time Period as follows:

```
Samples, N = 512
Time Period,T = 32pi
```

For these values , we get the error value in the order of 1e-15 .
Plotting the fft of estimated and expected gaussian functions , we get :





Conclusion

The fft library in python provides a useful toolkit for analysis of DFT of signals. The Discrete Fourier Transforms of sinusoids, amplitude modulate signals, frequency modulated signals were analysed. In the case of pure sinusoids, the DFT contained impulses at the sinusoid frequencies. The amplitude modulated wave had a frequency spectrum with impulses at the carrier and the side band frequencies. The frequency moduated wave, having an infinite number of side band frequencies, gave rise a DFT with non zero values for a broader range of frequencies. The DFT of a gaussian is also a gaussian and the spectrum was found to sharpen for higher sampling rates, while broaden for greater time ranges.