**Intro to FFT**

**Background:**

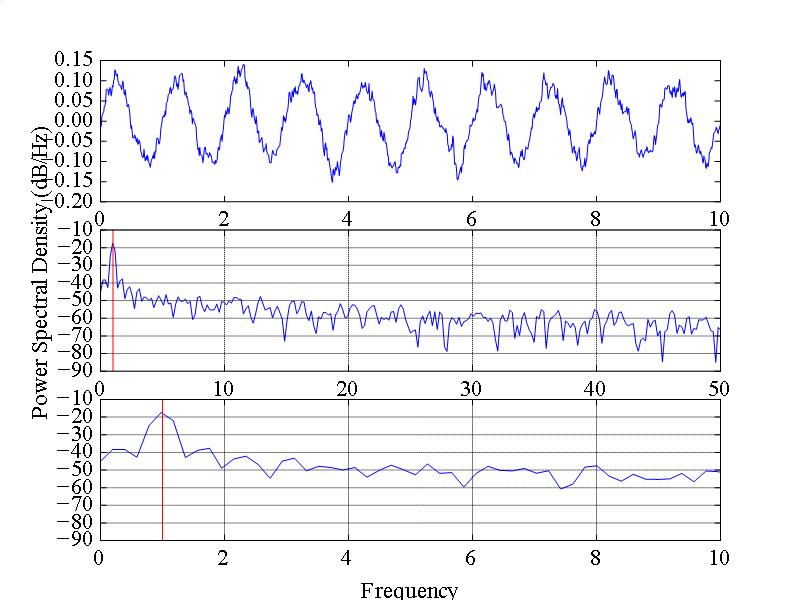
The Fourier Transform works by taking a function in time-space and converts it to frequency-space. It does this by approximating a function with a series of sine waves of all different frequencies such that when they are added together, they generate the original function. This implies that at each frequency, there is a corresponding amplitude for the sine wave. To convert to frequency-space, the amplitudes are plotted against the frequencies.

|  |  |
| --- | --- |
| Symbol | Definition |
| NFFT | Number of Fourier Transforms |
| Fs | Sampling Frequency |
| Fsig | Signal frequency |
| Nyquist Frequency | Upper limit for Fsig. Equal to Fs/2 |
| Bandwidth | How wide each frequency band is. Equal to Fs/NFFT |

**Example 1:**

The graph below is the sample FFT performed on matplotlib’s website: <http://matplotlib.org/api/pyplot_api.html>

It graphs a sine wave with equation 0.1\*sin(2pi\*t). Therefore, we can expect that after performing the Fourier transform, there should be a large spike at a frequency of 1 hz, as this is the main frequency of the actual function. The random other amplitudes of a considerably lower value (remember that dB is a logarithmic scale) come as a result of the randomness artificially generated by the program. This randomness is normal of actual data. A red line is drawn at x=1.



The third subplot is a zoomed in version of the second that shows that there is a peak at 1 Hz.

Below is the code that generates example 1:

from pylab import \*

fs = 100 #calculate sampling frequency

dt = 1.0/fs

t = arange(0,10,dt) #The domain of the function is from 0 to 10

nse = randn(len(t)) #adds noise

r = exp(-t/0.05)

Fsig = 1

cnse = convolve(nse, r)\*dt #The following creates a little bit of noise

cnse = cnse[:len(t)]

s = 0.1\*sin(2\*Fsig\*pi\*t) + cnse #This generates a sine wave on top of noise

subplot(311)

plot(t,s) #This plots s in the top panel.

subplot(312)

psd(s, 512, fs) #This runs the Fourier Transform.

axvline(x=Fsig, color = 'red') #Notice that the frequency of the graph sin(2\*pi\*k\*t) is k.

#This is why the vertical line is at x = k, as the FFT function will have a peak at x=k.

subplot(313)

psd(s,512,fs)

xlim([0,10]) #This zooms in on the figure.

axvline(x=Fsig, color = 'red')

ylabel('')

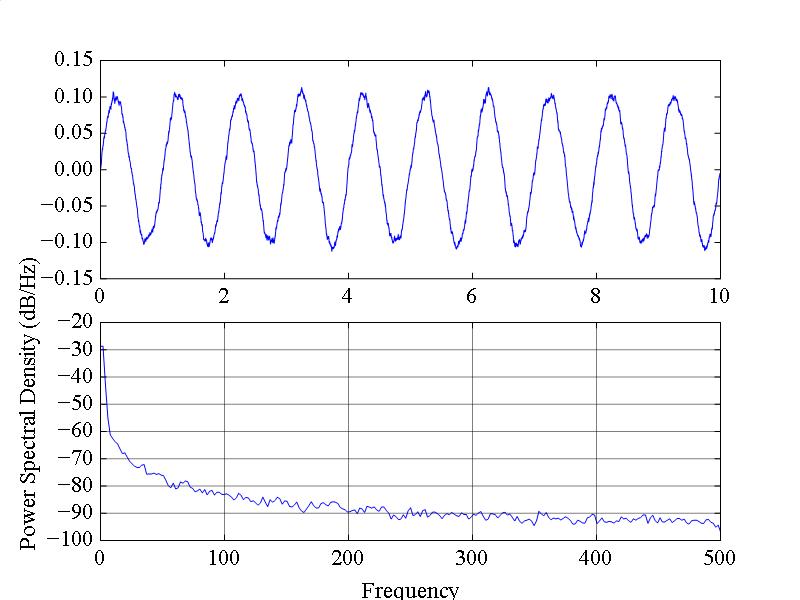
savefig('example1.jpg')

show()

Above is the python code that generated the plot. Dt represents 1/Fs where Fs represents the sampling frequency. Notice that the sampling frequency, 100, is twice the total number of frequencies plotted after the Fourier transform (The x-axis). That number, or half the sampling frequency, is known as the Nyquist frequency, or the limit of the Signal Frequency (Fsig). This is because the data starts to bounce back upon itself if frequencies greater than half the sampling frequency are tested.

**Example** 2:

Below is the graph if dt = 0.001 or Fs = 1000. Notice that frequencies up to 500 are graphed via the psd function. The psd function automatically sets the domain of the plot from 0 to the Nyquist frequency.

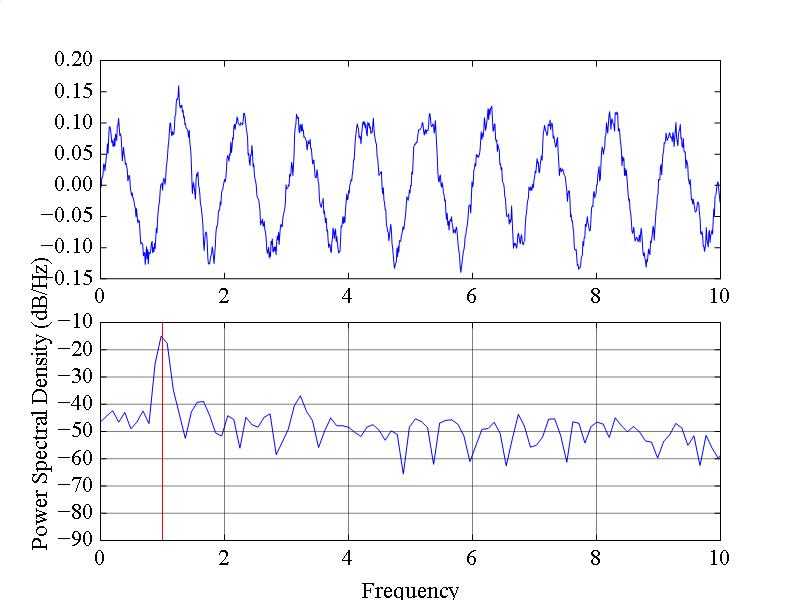


The actual ppd function according to matplotlib is matplotlib.pyplot.**psd**(x, NFFT=256, Fs=2, Fc=0, detrend=<function detrend\_none at 0x26a6ed8>, window=<function window\_hanning at 0x26aa140>, noverlap=0, pad\_to=None, sides='default', scale\_by\_freq=None, hold=None, \*\*kwargs)

Right now, in the program, the psd is used as psd(x, 512, 1/dt) We already discussed that 1/dt represents the frequency, but the meaning of NFFT is unclear. NFFT determines how many different frequency bins the Fourier Transform will test. Right now, NFFT = 512, but the default is 256. NFFT is most optimal if it is a power of 2.

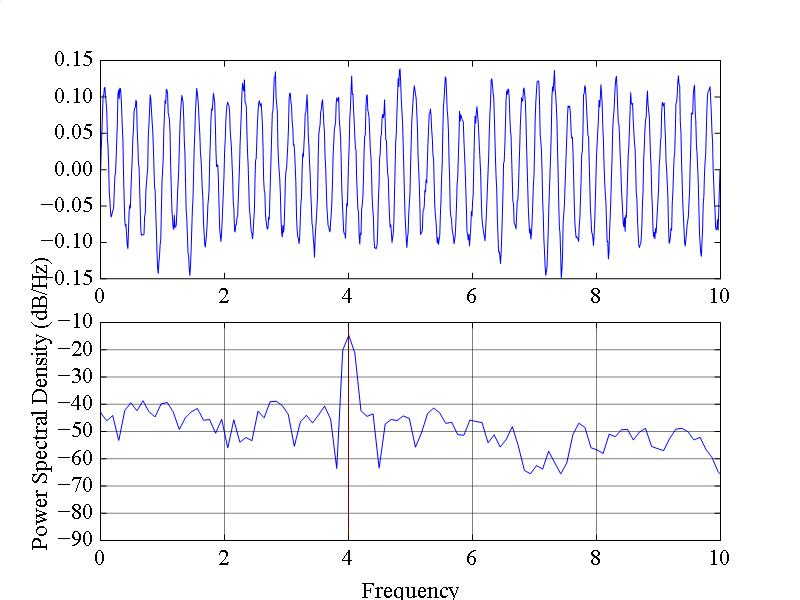
**Example 3:**

The number of frequencies that the Fourier transform will actually plot is NFFT/2. This is most clear by an example. dt = 0.01 and NFFT = 1024.



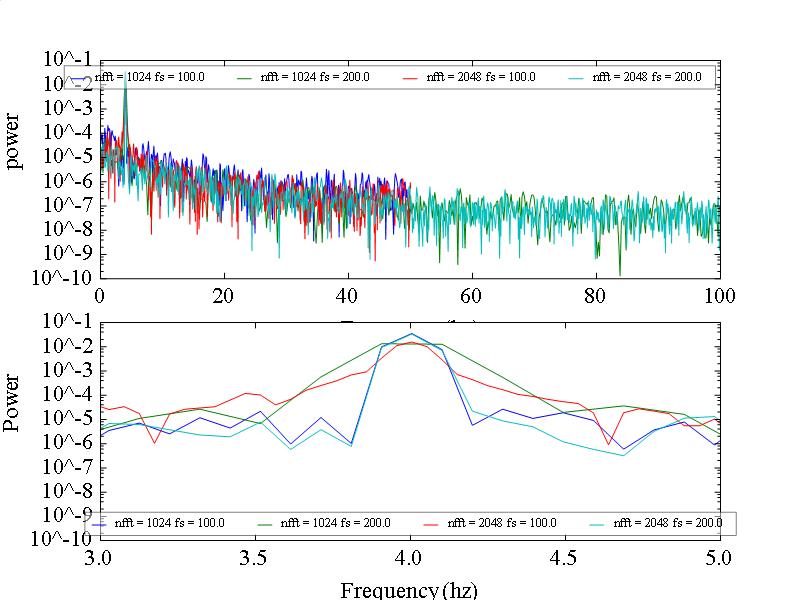
**Example 4:**

If we let the Signal Frequency be 4, or set Fsig in the program equal to 4, we can verify that a peek occurs at 4 Hz rather than 1, as this is the frequency of the sine wave. So long as Signal Frequency is less than the Nyquist Frequency (Fs/2) No problems occur.



**Example 5:**

The sampling frequency determines the highest frequency at which the FFT will test, and the NFFT determines how many bins are being generated. So, the bandwidth, or the distance between each frequency point on the x-axis is dependent on both of those. Since the max frequency is Fs/2, and the number of bins is NFFT/2, the bandwidth is given by Fs/NFFT. Below is a graph which demonstrates this.



**Example 6:**

The next example will demonstrate why the Signal Frequency must be lower than the Nyquest Frequency. Below are the FFTs of three functions, all with Nyquist Frequency 50. Below is the result of taking psd’s for Fsig= 49, 50, and 51 respectively. Notice that the result of the FFT implies that the Signal rate of the third graph is 49 when it is actually 51. Also notice that there is absolutely no peak for the 50.

