Noise and The Discrete Fourier Transform

The Fourier Transform is a mathematical technique named after the famed French mathematician Jean Baptiste Joseph Fourier 1768-1830. Today, the Fourier Transform is widely used in science and engineering in digital signal processing. The application of Fourier mathematical techniques is prevalent in our everyday life from everything from television to wireless telephones to satellite communications.

What is the Discrete Fourier Transform (DFT)?

The mathematical technique called the DFT takes a discrete time series of n equally spaced prices, and transforms or converts this time series through a mathematical operation into a set of n complex numbers defined in what is called the *frequency domain*.

Why bother to do this? Well it turns out that one can do all kinds of neat analysis tricks in the frequency domain which are just to hard to do, computationally wise, with the original time series in the time domain. If the assumption is made that the time series that is examined is made up of oscillating signals of various frequencies plus noise, than in the frequency domain we can filter out the frequencies we have no interest in and also minimize the noise content of the price data.

A simple example will demonstrate.

Define the following signal that has two oscillating components, a trend and constant value:

$$\mathbf{signal(i) = 2*sin(\pi f_1 i) + cos(\pi f_2 i) + 1 + 0.005*i \quad for\ i=0\ to\ 511}$$

where f_1 = 9/512 and f_2 = 4/512

A graph of this signal would look like:

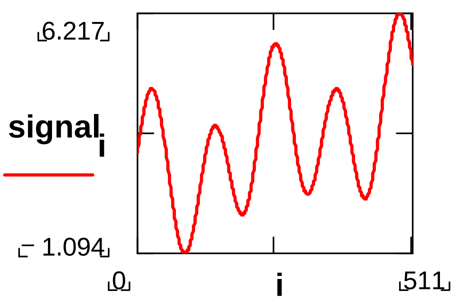


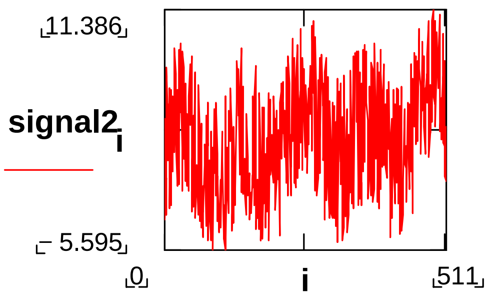
Figure 1, Constructed Signal vs time (i)

Add some noise to the signal:

signal2(i) = signal(i) + rnd(12) – 6   for each i, i=0 to 511

where **rnd(12)** is a random number between 0 and 12. We subtract 6 from this number to make the random number be between +6 and –6.

A graph of the signal plus added noise would look like this:



Signal with noise added

With the added noise, the signal has all but disappeared.

Let's take the Fast Fourier Transform of Signal + Noise and see what it looks like in the *frequency domain*. The Fast Fourier Transform (FFT) is a mathematical algorithm that computes the DFT very fast. Let **f** be the **FFT** of the signal **signal2** and let $|\mathbf{f}_j|$ be the magnitude the individual frequency components.

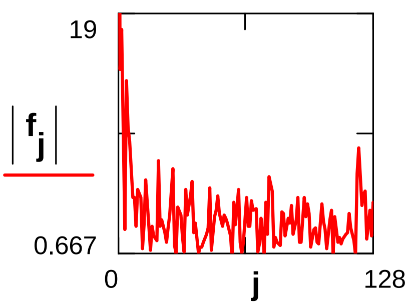


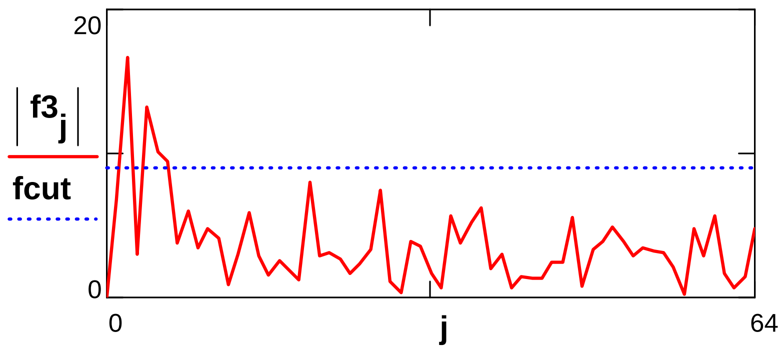
Figure 3, Frequency Magnitude of Signal+Noise

This is not helpful graph! The frequency magnitudes are all about equal accept for the magnitude at j=0 which is very large. The magnitude at j=0 corresponds to a constant in the time domain. What happened? This is an example of what happens when the trend and series average are not taken out of the time series before the **FFT** is done. The trend and the average completely swamp the frequency domain such that none of the characteristics we are looking for can be found.

Taking out the trend and the average,

signal3(i) = signal2(i) –1 – 0.025*i for each i, i=0 to 511.

Taking the **FFT** of **Signal3** . Let **f3** be the **FFT** of the signal **signal3** and let **$|f3_j|$** be the magnitude the individual frequency components.

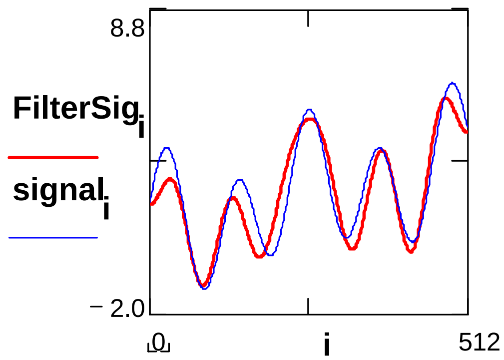


Frequency Magnitude chart of Signal+Noise minus trend and average

We can see the two clear frequency peaks in the FFT frequency magnitude chart.

We can use what is called a threshold noise filter and filter the noise out by only accepting those frequencies whose magnitudes exceed the threshold of 9 .

Zero out all frequencies below 9 in magnitude and call the resultant frequency spectrum **f3cut.** Take the inverse **FFT** of **f3cut**, and call it **FilterSig**. Add back the trend and constant and **FilterSig** becomes the following graph.



Inverse FFT of the Noise Filtered Signal **f3cut** with trend and constant added back. The Blue line represents the original signal before we added noise and the red line represents the noise filtered signal.

It can be seen that this method has successfully filtered out the added noise and retrieved almost all of the original noiseless signal.