SIGNAL ANALYSIS CONCEPTS

By
John Ehlers

INTRODUCTION

This is the first of a series of articles where I will teach you the skills and secrets of applying signal analysis. I learned many of the principles of signal analysis in my Electrical Engineering training, and have applied these over the years in practicing the art of technical analysis of the market. As an engineer, signal analysis is an everyday obsession to me. Technical Analysis is what I do – practical advice is what I am giving. Practical advice that you can use right now.

If you have a computer and if you are playing the market you cannot avoid technical analysis. You can either do this analysis poorly or you can do it well. It is simply a matter of study and applying techniques that are proven in the high-tech world every day. My objectives for you in this series of articles are fourfold:

1) To learn the skills I have acquired through years of research.
2) To discover the secrets for successful application of technical analysis.
3) To use these skills and secrets for yourself.
4) To win by making better trades and having more fun.

What I have learned and discovered are the inputs. What I will share with you as outputs are the creative strategies and tangible tools that you can put to immediate use.

I know many of you will think the concepts are too technical. I can only respond that your intensity must amplify the strategies and structure that I provide. With dedication, I assure you that all the techniques can be grasped, absorbed, and applied by you. This will result in a larger perspective, a deeper insight, and expanded trading techniques. What I will be telling you is what works. What you will be using is what counts.

In this first article I make a few observation about moving averages, simply because averages are of universal interest to technical analysis. I also introduce a new concept of data streaming using Hilbert Transforms. Through the use of Hilbert Transforms I show how to measure signal-to-noise ratio and cycle period. So let's get to it.

MOVING AVERAGE LAG

Moving averages are ubiquitous to technical analysis. I would therefore be remiss if I didn't comment on some techniques to improve their application. Two general characteristics of all moving averages are that they smooth the input data and they lag that data. Their use and application is almost always a tradeoff between these characteristics. The smoothing function means that the higher frequency components of the input data are removed, so moving averages are also referred to as low pass filters. That is, they pass only the low frequency components while removing the high frequency components.

Moving average lag is probably the most important characteristic for traders to understand quantitatively. Figure 1 is a schematic of how a simple moving average is formed. Data within the observation window is averaged to produce a single point. The observation window is moved from bar to bar on the chart to form the continuous moving average. If the weighting of the data values within the observation window is uniform, the average value of the data is centered in the vertical dimension of the window and is also centered in the vertical dimension of the window. Since the simple moving average is plotted with respect to the right hand side of the observation window, the lag must be half the width of the observation window. When more complex weighting functions are applied to the data, such as a linearly weighted moving average, the lag will be the center of gravity (c.g.) of the weighting function. A linearly weighted moving average will therefore have a lag equal to one third the observation window width.

Since the exponential moving average (EMA) is commonly used in technical analysis, we will describe how to compute the averaging constant α in terms of the lag that the EMA produces. The mathematical expression for an EMA is:

$$f(z) = \alpha *g(z) + (1 - \alpha)*f(z - 1)$$

In this expression z is the counter for the sampled variable, f is the filtered output, and g is the input data. In terms of daily price bars this equation says that the filtered output today is equal to the alpha fraction times today's price plus the complement of the alpha fraction times yesterday's filtered output. Note that if the input data is a constant value, the filtered output will ultimately become this constant value also because α and $(1 - \alpha)$ sum to unity. With reference to Figure 2, assume the input price is a continuous trend and has a value "I" for the current bar. If the trend has a slope "S" and the EMA has a lag "L", then we can write the EMA as:

$$I - L*S = \alpha *I + (1 - \alpha)*(I -S*L - S)$$

$$I - L*S = \alpha* I + (I - S*L) - S - \alpha *I + \alpha *S*(L + 1)$$

So that

1) $\alpha = 1 / (L + 1)$

Equation 1 is an important equation. Once we know the lag we can tolerate, we can calculate the EMA alpha directly. For example, if we can stand a 3 bar lag with our EMA, we would use $\alpha = .25$. For completeness, we note that Jack Hutson[1] related the EMA alpha to the period of a simple moving average (P) as

2) $\alpha = 2 / (P + 1)$

Equation 2) is approximately the same as equation 1) because the lag of the simple moving average is half its period.

SIMPLE ZERO LAG MOVING AVERAGE

There are several ways to obtain a zero lag moving average. One method basically takes an average from left to right across the screen, accepting the lag. Next, a moving average of that average is taken from right to left across the screen. This way smoothing is doubled and lag is cancelled. The only problem with this approach is that a peak into the future is required to make this zero lag moving average work at the right hand edge of the screen.

[1] Jack Hutson, "Filtered Price Data: Moving Averages vs. Exponential Moving Averages", Technical Analysis of Stocks & Commodities, 1984, Vol 2, #3, p102.

In a later article we will discuss Kalman filters. A Kalman filter can approximate a zero lag moving average.

Here is a simple way to produce a zero lag moving average: With reference to Figure 3, note we have two moving averages. The first moving average has a lag "L" and the second moving average has a lag of "2L". This means the lateral separation from the price to the first moving average is exactly the same as the lateral separation between the two moving averages. In the case of the linear trend we can think in terms of moving the curves vertically to effect a lateral displacement. If we take the difference between the two moving averages and add this difference to the first moving average, we have exactly reconstructed the original price trend. We therefore have, in effect, a zero lag moving average. Since smoothing is related to the lag L, we must be judicious in our selection of the lag. For example, if we select a lag of 4 bars, we must recognize that cycle periods shorter than about 4 bars will be attenuated in the output of the bandpass filter. Figure 4 shows the zero lag output computed from EMAs having a 4 bar and an 8 bar lag (α = .2 and α = .111). This zero lag output is compared to the shorter average EMA. Please note that this filter DOES remove the pure trend but DOES NOT provide zero lag for cycle mode signals.

PHASOR NOTATION

It is convenient to think of signals for analysis in terms of phasors because this concept enables the mathematical formulation and solution of many problems. This analysis only considers the AC (Alternating Current) component of the signal, and so the signal must be first detrended before the analysis is performed. A phasor is described in Figure 5. Picture the phasor as a bicycle crank rotating counterclockwise. If we put a ballpoint pen at the end of the arrow and pulled a sheet of paper under the crank in the same way seismographs are created, the rotating phasor will plot out a sinewave in the time domain. Therefore, the phasor can be used to describe the frequency, amplitude, and phase of all the frequency components of the signal. Signal frequency is the rate at which the phasor rotates. Amplitude is the length of the phasor. Phase is the angle at which the phasor is pointing at any selected instant. At the beginning, we will consider the signal to be comprised of only one dominant cycle to minimize complexity. Our analysis assumes the phasor is rotating, so we are primarily interested in the relative phase differences between analysis components. We simplify the analysis by disregarding the rotation in our notation.

The names used in our analysis are almost whimsical. We will use the theory of Complex Variables, which are not really all that complex. In so doing, we will divide the signals into real and imaginary components. The imaginary components are not a figment of our imagination. Mathematicians tagged the square root of minus one as the imaginary operator "j", since this number cannot possibly exist. What is really done is to create a two dimensional number space with the "real" numbers going to plus and minus infinity in the horizontal dimension and the "imaginary" number going to plus and minus infinity in the vertical dimension. The imaginary operator "j" rotates numbers from the real axis to the imaginary axis. When we multiply two complex numbers we have:

U1 = x1 + jy1
U2 = x2 + jy2
U1*U2 = (x1*x2 – y1*y2) + j(x1*y2 + x2*y1)

The minus sign occurs because j*j = -1. It is often easier to consider numbers in the complex plane in polar coordinates; a radius at a given angle. This is written as $Re^{j\theta}$. The angle is written as an exponential because when two phasors are multiplied, the complex product is the product of the two amplitudes at an angle that is the sum of the two angles. This is a characteristic of summing in a product is the same as multiplying exponentials. This polar notation brings us back to our bicycle crank analogy of the phasor.

The objective of using complex variables is to establish two orthogonal components. The two directions in the complex plane are at right angles and therefore are orthogonal. Another mathematical definition of orthogonal is that the two components are completely uncorrelated. One way of viewing orthogonality is that motion in the imaginary direction has no impact at all on motion in the real direction. Another simple example illustrates this orthogonality. A Sine has zero amplitude at a phase angle of zero, rises to a maximum at a phase angle of 90 ^{o}, falls back to zero at 180 ^{o} phase, reaches a negative maximum at 270 ^{o}, and returns back to zero at 360 ^{o} phase. A Cosine is a Sine multiplied by the "j" operator. That is, a Cosine has maximum amplitude at zero degrees phase, decreases to zero at 90 ^{o} phase, goes to a maximum negative amplitude at 180 ^{o} phase, rises to zero amplitude at 270 ^{o} phase, and returns to maximum amplitude at 360 ^{o} phase. There is a 90 ^{o} rotational relationship between these two functions, and therefore they are orthogonal. One can see the Sine and Cosine are uncorrelated by computing

$$0 = \int_0^{2\pi} Sin(\theta)Cos(\theta)\, d\theta$$

You have just completed a crash course in complex variables. It was perhaps a difficult backgrounder, but a necessary one to understand some of the aspects of signal processing. Real and Imaginary components (or In Phase and Quadrature components as they are more often called) are used in all modern signal analysis ranging from your audio system to receiving signals from satellites. We traders don't have the luxury of In Phase and Quadrature components in the data. All we have is a stream of real sampled data. It turns out that In Phase and Quadrature components can be generated from the real data stream in an All-Pass Filter called a Hilbert Transform. How a Hilbert Transform works is well beyond the scope of this article, but interested readers can find a description by Charles Rader in the technical literature[2]. I find that the theoretical Hilbert Transform must be slightly modified for market data. With due respect to Mr. Rader, I find the equations for the Hilbert Transform outputs should be as indicated in the EasyLanguage code of sidebar 1.

****************************** SideBar One ***

EasyLanguage Code for Hilbert Transform

```
Inputs:     Price((H+L)/2);

Vars:       Imult(.635),
            Qmult(.338),
```

[2] Charles M. Rader, "A Simple Method for Sampling In-Phase and Quadrature Components", IEEE Transactions on Aerospace and Electronic Systems, Vol AES-20, No 6, November 1984, p 821

```
                InPhase(0),
                Quadrature(0);

If CurrentBar > 5 then begin

        {Detrend Price}
        Value1 = Price - Price[7];

        {Compute Hilbert Transform}
        Inphase = 1.25*(Value1[4]  - Imult*Value1[2]) + Imult*InPhase[3];
        Quadrature = Value1[2] - Qmult*Value1 + Qmult*Quadrature[2];

        {Plot the results}
        Plot1(Inphase, "I");
        Plot2(Quadrature, "Q");
end;

********************************************************************************************
```

The best way for you to test this code for yourself is to generate a theoretical sinewave in Excel. To do this, you only generate a counter (C) for each row. The value of the sinewave at each cell would be equal to Sin(6.28*C/Period). The period can be anything you choose, 20 for example. Then, all you have to do is to clean the data into a form that TradeStation will read and export this data to an ASCII file.

The plot of the InPhase and Quadrature components of a theoretically chirped (continuously varying period) sinewave is shown in Figure 6. It appears that the crossing of the quadrature and negative of the InPhase component would make an ideal indicator, signaling the cycle turning point before it occurs. However, I would caution you not to do this because the Hilbert Transform has a group delay of approximately four bars due to its construction. While this delay is of little consequence for longer cycles, this four bar delay is a half cycle of an eight bar cycle. Thus, in these shorter cycles you would be getting exactly the wrong trading signal. In a later article we will revisit an indicator that produces excellent cycle mode signals.

SIGNAL TO NOISE RATIO

Now let’s put the InPhase and Quadrature components to serious use. The signal amplitude is just the length of our phasor. With reference to Figure 7, and recalling the Pythagorean Theorem, the length of the phasor is the square root of the sum of the Squares of the In Phase and Quadrature components. We therefore have the signal amplitude on a bar-by-bar basis. We must smooth this with an EMA to avoid having the resultant appear very choppy.

The signal amplitude is not much use by itself. However, if we can estimate the signal amplitude relative to the market “noise”, then we have a tool that estimates the quality of our technical analysis. With the kind of data we have available to us, let us develop a unique definition of noise. A sampled signal is shown in Figure 8a as a sinewave with the sampling uncertainty as the high and low of each bar. The high and low is the uncertainty of each of our perfect sinewave sample points. We can make good trades as long as our Signal amplitude is much larger than the average daily range of the bars. When half the average daily range becomes equal to the signal amplitude making money on a trade becomes a crapshoot. Under this condition, shown in Figure 8c, we could make an entry at the low of the bar containing the signal high and make an exit at the high of the bar containing the signal low for zero profit. Therefore we will define the case where half the average daily trading range is equal to the signal amplitude as our Zero Decibel Signal to Noise (SNR) condition. We want the signal amplitude to be at least twice the noise amplitude (6 dB SNR) to have a reasonable chance to make a profit from our analysis.

The EasyLanguage code to calculate and plot the SNR is shown in SideBar 2.

************************************* SideBar Two ***
EasyLanguage Code for SNR Indicator

```
Inputs:         Price((H+L)/2);

Vars:           Imult (.635),
                Qmult (.338),
                InPhase(0),
                Quadrature(0),
                Amplitude(0),
                Range(0);

If CurrentBar > 8 then begin

        {Detrend Price}
        Value1 = Price - Price[7];

        {Compute "Noise" as the average range}
        Range = .2*(H - L) + .8*Range[1];

        {Compute Hilbert Transform outputs}
        Inphase = 1.25*(Value1[4]  - Imult*Value1[2]) + Imult*InPhase[3];
        Quadrature = Value1[2] - Qmult*Value1 + Qmult*Quadrature[2];

        {Compute smoothed signal amplitude}
        Value2 = .2*(InPhase*InPhase + Quadrature*Quadrature) + .8*Value2[1];

        {Compute smoothed SNR in Decibels, guarding against a divide by zero error,
and compensating for filter loss}
        If Value2 < .001 then Value2 = .001;
        If Range > 0 then Amplitude = .25*(10*Log(Value2/(Range*Range))/Log(10) +
1.9) + .75*Amplitude[1];

        {Plot Results}
        Plot1(Amplitude, "Amp");
        Plot2(6, "Ref");

end;
```

****

INSTANTANEOUS CYCLE PERIOD MEASUREMENT

One basic definition of a cycle is that the phase has a constant rate of change. For example, a 10 bar cycle changes phase at the rate of 36° per day so that 360 ° of phase is completed (one full cycle) every 10 bars.

In the case of our In Phase and Quadrature components, the phase angle for each bar is just the ArcTangent of the ratio of the Quadrature component to the In Phase component. All we have to do to determine frequency is compute the difference in phase from bar to bar and sum these differences backwards into history until the sum is equal to or greater than 360°. It is really that easy. Now aren't you glad you waded through all that Complex Variable stuff?

While the instantaneous measurement of frequency is a spectacular result of using the Hilbert Transform and it is vastly superior to FFT measurements, truly accurate cycle measurements can only be made by further reducing the effective noise bandwidth in a high resolution technique such as MESA[3]. MESA currently uses the instantaneous cycle period measurement to establish the length of data to be used for the MESA cycle measurement.

The EasyLanguage code to compute and plot the Instantaneous cycle period is shown in SideBar 3. To demonstrate that this code accurately measures frequency, its calculation for the cycle period is plotted below the Chirp theoretical waveform in Figure 9. You can check the measurement by counting the number of bars between successive highs (or lows) in the waveform.

********************************* SideBar Three **

EasyLanguage Code to Measure Cycle Period

```
Inputs:     Price((H+L)/2);

Vars:       Imult (.635),
            Qmult (.338),
            InPhase(0),
            Quadrature(0),
            Phase(0),
            DeltaPhase(0),
            count(0),
            InstPeriod(0),
            Period(0);
```

[3] Abhay Samant & Sam Shearman, "High-resolution frequency analysis with a small data record", IEEE Spectrum, September 1999, p82-86

```
If CurrentBar > 5 then begin

    {Detrend Price}
    Value3 = Price - Price[7];

    {Compute InPhase and Quadrature components}
    Inphase = 1.25*(Value3[4]  - Imult*Value3[2]) + Imult*InPhase[3];
    Quadrature = Value3[2] - Qmult*Value3 + Qmult*Quadrature[2];

    {Use ArcTangent to compute the current phase}
    If AbsValue(InPhase +InPhase[1]) > 0 then Phase =
ArcTangent(AbsValue((Quadrature+Quadrature[1]) / (InPhase+InPhase[1])));

    {Resolve the ArcTangent ambiguity}
    If InPhase < 0 and Quadrature > 0 then Phase = 180 - Phase;
    If InPhase < 0 and Quadrature < 0 then Phase = 180 + Phase;
    If InPhase > 0 and Quadrature < 0 then Phase = 360 - Phase;

    {Compute a differential phase, resolve phase wraparound, and limit delta phase
errors}
    DeltaPhase = Phase[1] - Phase;
    If Phase[1] < 90 and Phase > 270 then DeltaPhase = 360 + Phase[1] - Phase;
    If DeltaPhase < 1 then DeltaPhase = 1;
    If DeltaPhase > 60  then Deltaphase = 60;

    {Sum DeltaPhases to reach 360 degrees.  The sum is the instantaneous period.}
    InstPeriod = 0;
    Value4 = 0;
    For count = 0 to 50 begin
        Value4 = Value4 + DeltaPhase[count];
        If Value4 > 360 and InstPeriod = 0 then begin
            InstPeriod = count;
        end;
    end;

    {Resolve Instantaneous Period errors and smooth}
    If InstPeriod = 0 then InstPeriod = InstPeriod[1];
    Period = .25*(InstPeriod) + .75*Period[1];

    Plot1(Period, "DC");
end;

************************************************************************************************
```

PUTTING IT ALL TOGETHER

The topics we have described in this article are used in two EasyLanguage Indicators, the Hilbert SNR and the Hilbert Period. An example of their application to real world data is shown in Figure 10. We want to trade only when the SNR is greater than 6 dB. Knowing the instantaneous frequency and the location of the last significant low (or high), we know the best time to make a market entry. We will let you examine the charts to determine if these new trading tools are of value to you.

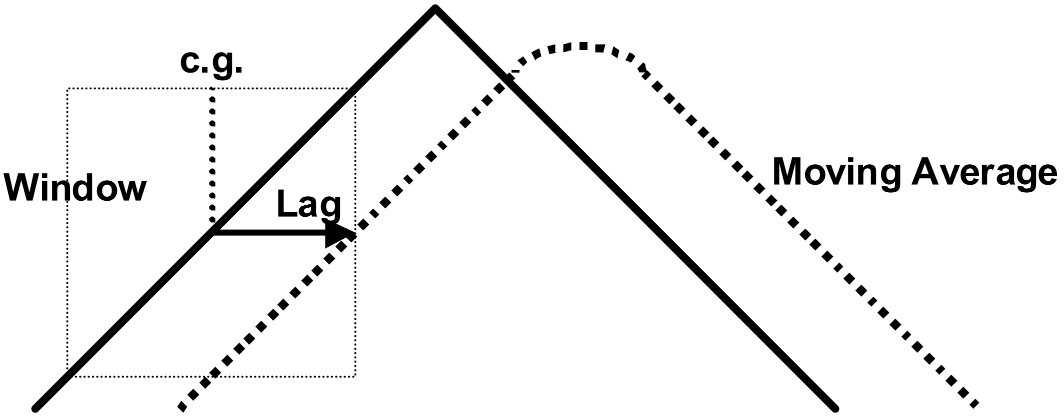


Figure1. An average formed over the width of an observation window is plotted at the right hand side of the window to produce lag. The observation window is moved along the data set to produce a moving average.

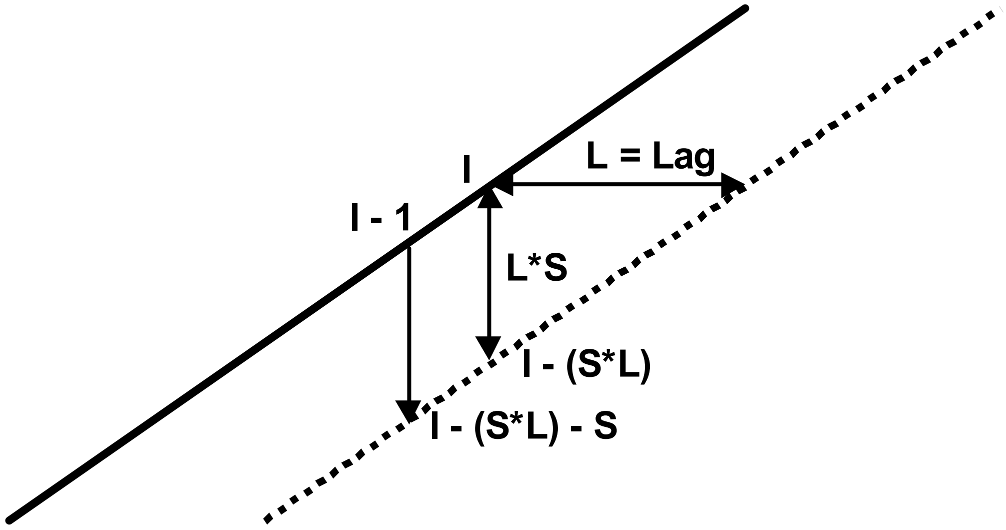


Figure 2. Calculating the α of an Exponential Moving Average

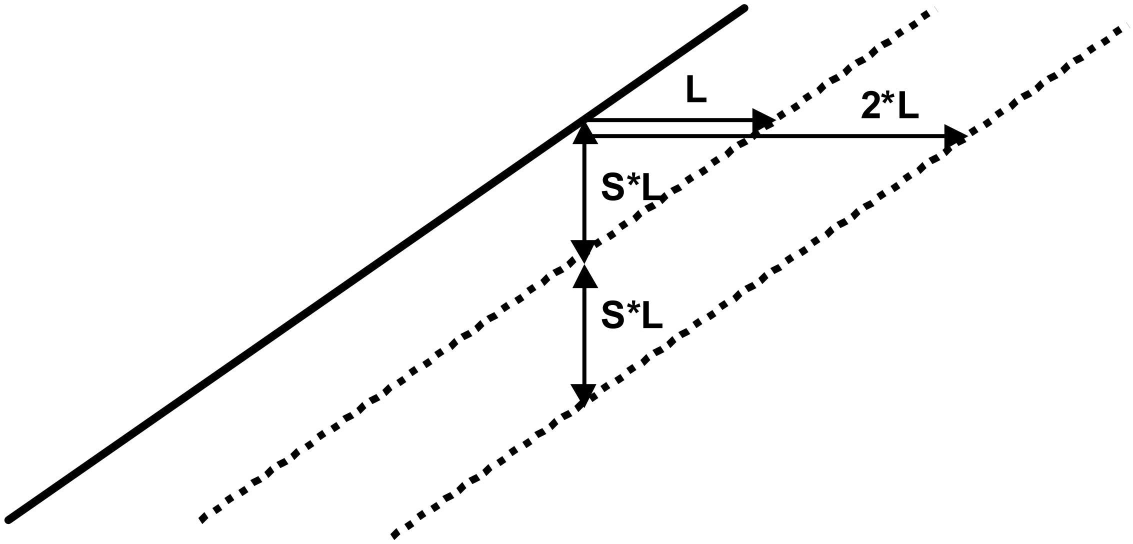


Figure3. Constructing a Simple Zero Lag Moving Average

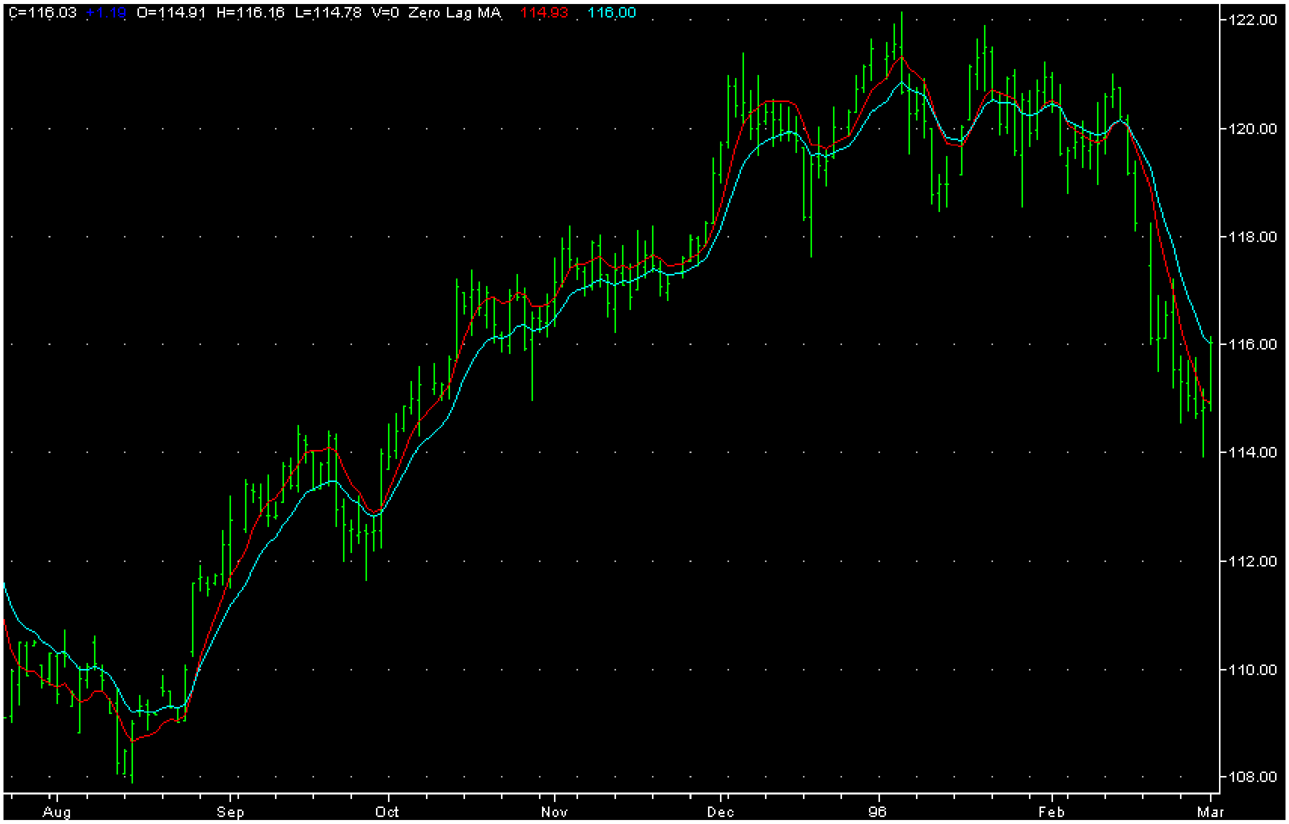


Figure 4. A Simple Zero Lag Moving Average Compared with an EMA

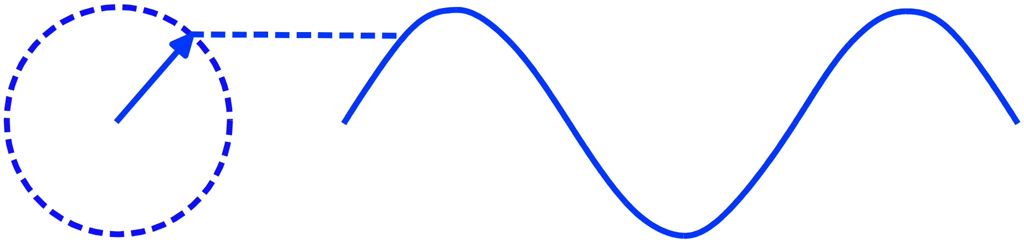


Figure 5. Generation of a Sinewave from a Phasor

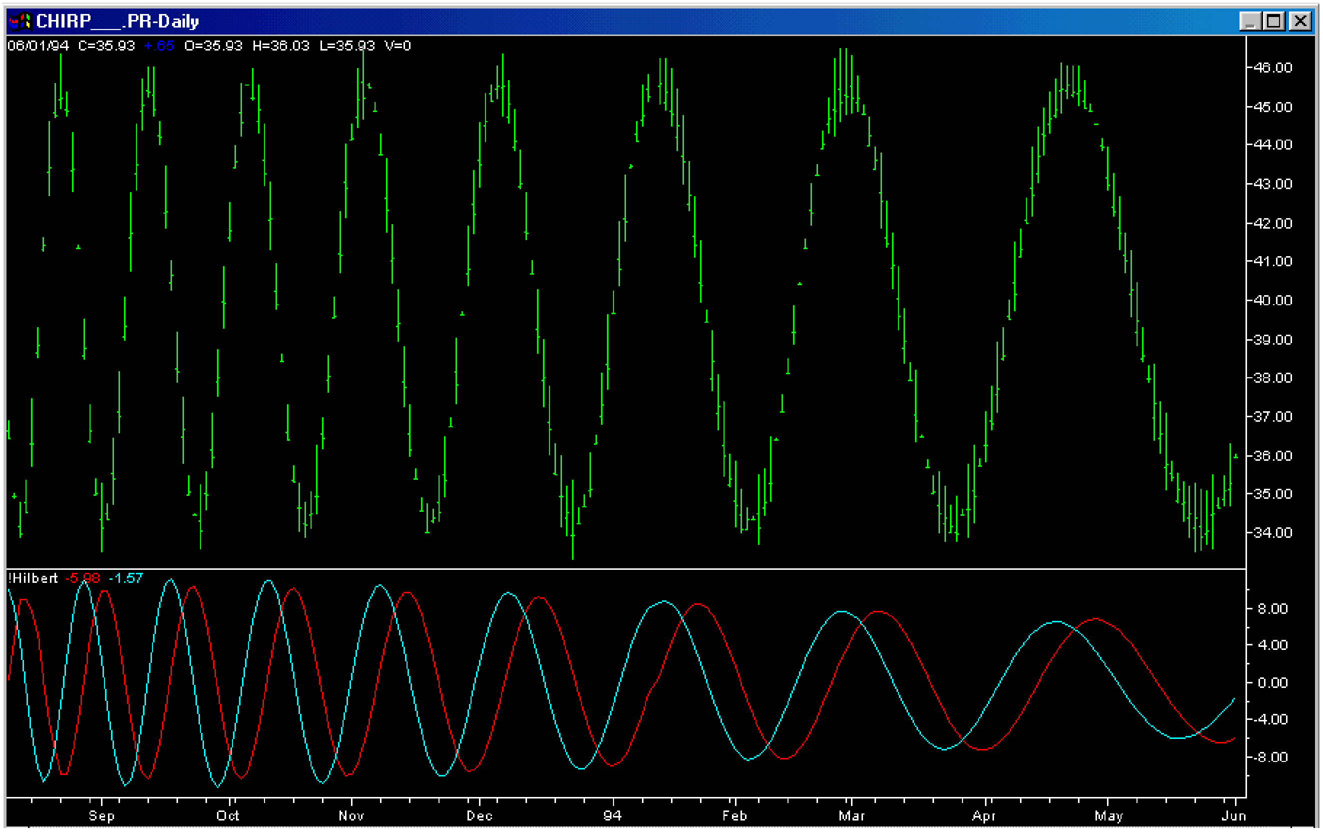


Figure 6. Hilbert Transforms of a Theoretical Chirped Signal

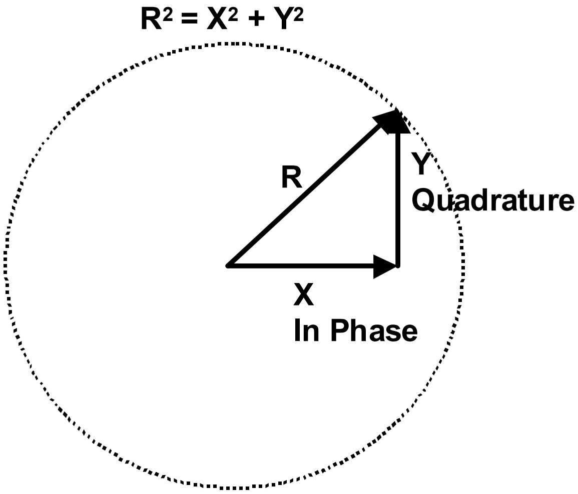
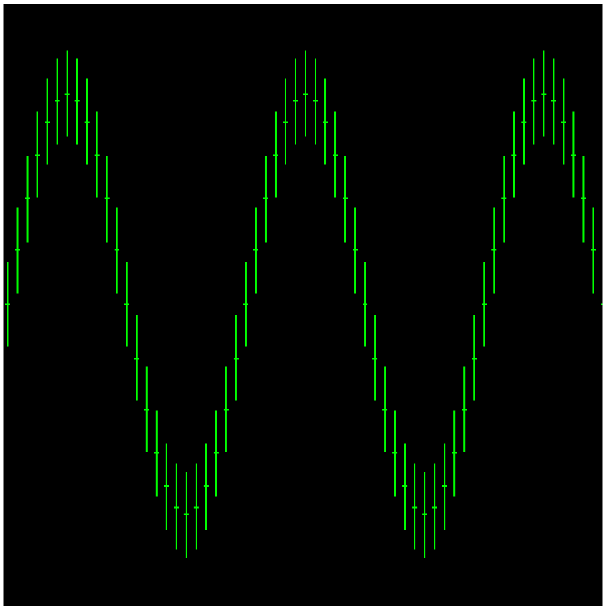


Figure 7. Computing the Phasor Amplitude



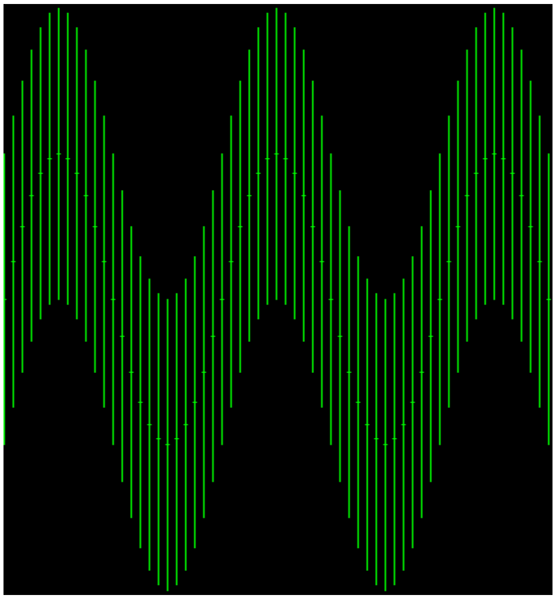


Figure 8a. Sampled Data

Figure 8b. Zero dB SNR Data

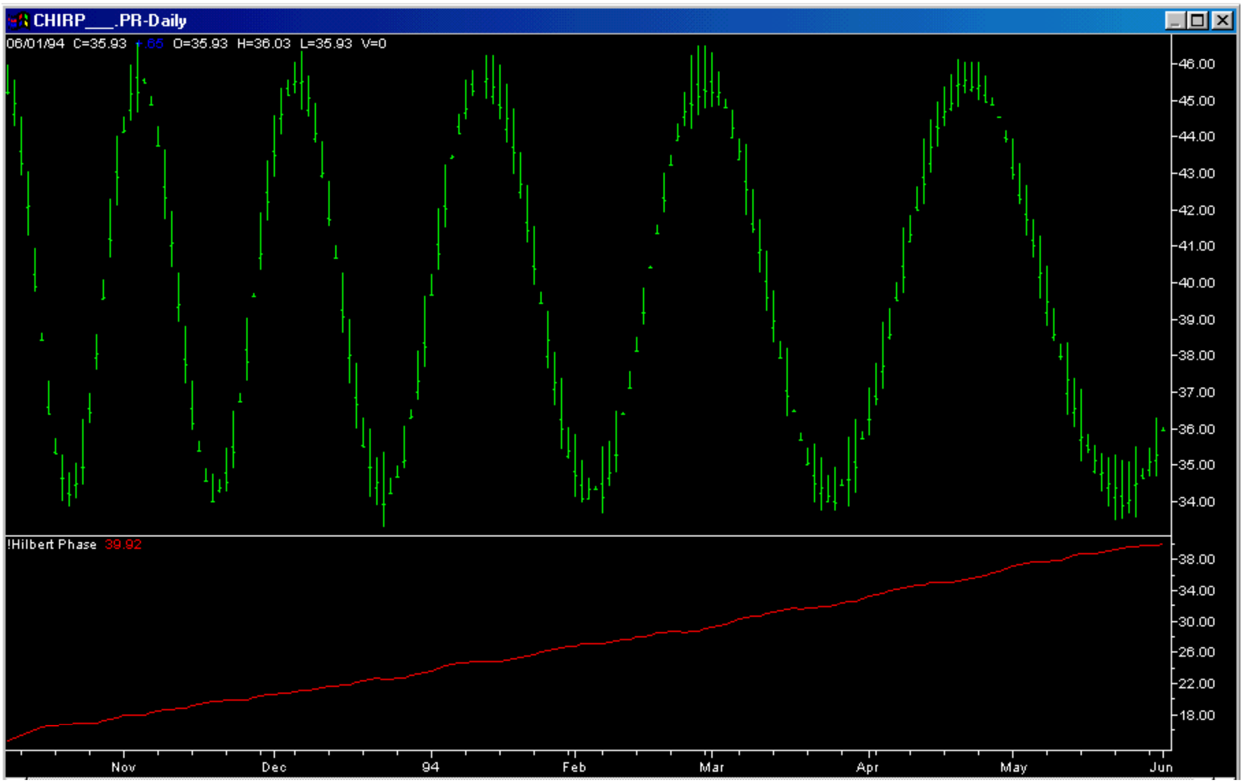


Figure 9. Cycle Measurement of a Theoretical Chirped Signal

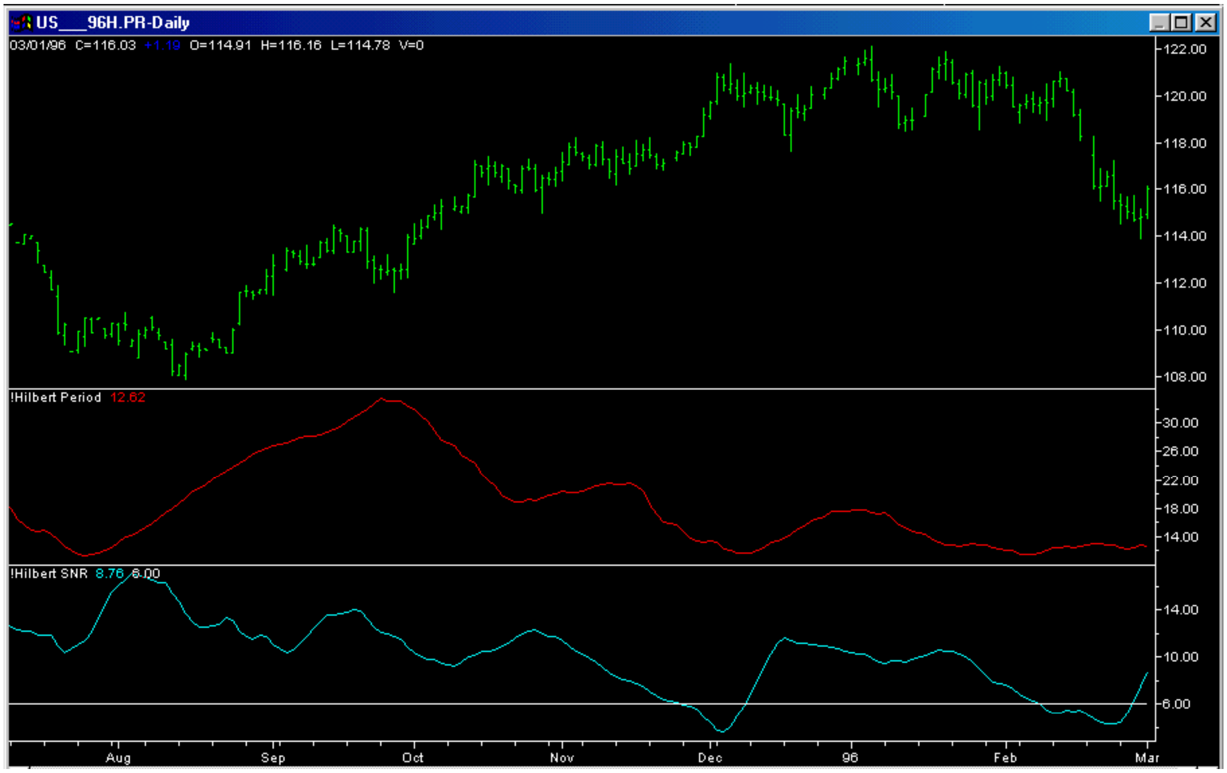


Figure 10. Hilbert Period and Hilbert SNR on Real-World Data