5. Convolution, filters and the discrete Fourier transfom

6. Spectra and spectrograms

Kristian Beedholm, Biology, SDU, Odense, Denmark

The complete and unabridged guide to FIR filters, filter banks, spectrograms and the Fourier transform in 8 pages and two lectures

About this document

In this document I have tried to bring across the understanding of basically two central tools, Fourier transformation and convolution. Convolution is introduced first, as the process is – as will be clear shortly – a familiar one, whereas Fourier transform is introduced in the end. Reading this should make you able to easily grasp anything presented in the two companion lectures. These will also deal with spectrograms and, if time allows, noise description.

Introduction

Signal analysis is concerned with the mathematical and statistical analysis of a not particularly well-delimited class of functions, collectively called signals. Another closely related discipline is signal processing, which deals in *changing* signals, storing them effectively and things like that. You may tell books on this subject from books on signal processing in that they introduce the Fourier transform only in later chapters, and tend to be focused on more general transforms the names of which I will not even relate, since this is supposed to be analysis …

In practice we only ever deal in digital, finite duration signals, whereas we most often only want to make statements about analog signals, and quite often these are of an (almost) infinite duration type such as the broadband background noise overlying the recordings. For this to be valid despite of the discrepancies, certain rules must be at least considered. It is assumed that all signal are sampled at a rate that is larger than twice the highest frequency in the signal to be analyzed.

The central tool in signal analysis is still, and will continue to be – at least for the remainder of the course – the Fourier transform, which generates "the complex spectrum" of a signal to be analyzed. Signal analysis tends to look slightly scary since the Fourier integral contains complex exponentials and is, well it's an integral, isn't it? On the other hand, once this hurdle is overcome everything just keeps getting easier - it's only difficult at a glance. Both this document and the lectures aim at graphically illustrating the principles of the discrete Fourier transform, so we will not even encounter an integral.

Convolution, a moving average filter example

As promised earlier I will introduce the concept of convolution first. This is not a very usual approach, but bear me with me on this.

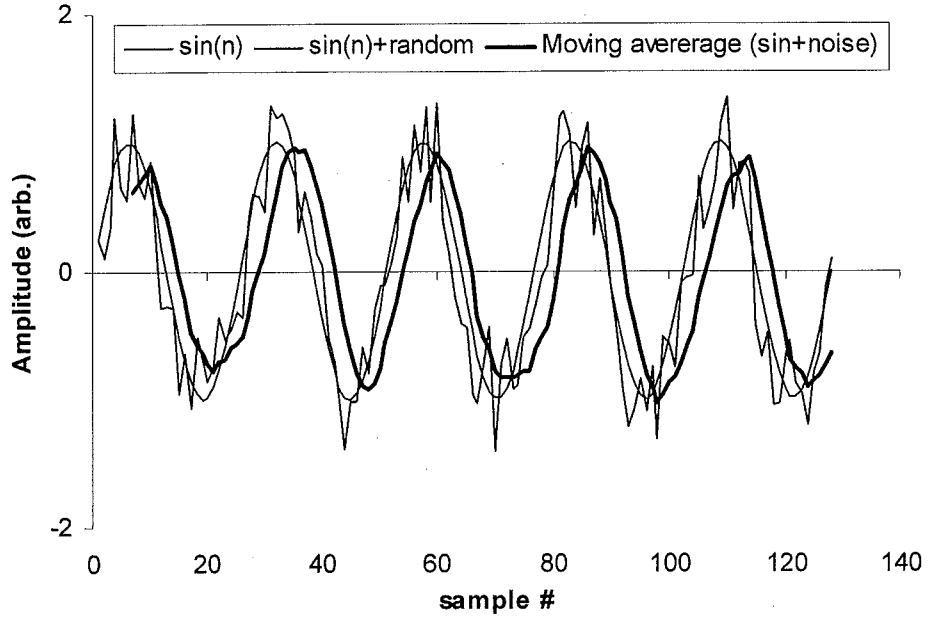

Fig 1 Excel example of low pass filtering through convolution. The red curve is the theoretical signal. The blue curve is the same data with some noise added to it. The black line is a 7 samples long moving average of the noisy data. This is (a part of) a convolution result, really. The delay of the partly denoised estimate with respect to the original is worth noting and explaining.

Consider therefore the signals in Fig. 1. The signal to be analyzed is a sinusoid with broadband noise, meaning there is a random number added to each sample. This could be a typical situation describing parts of a recording of a biological communication signal[1]. The noise is the unwanted part of the recording. To get an idea about the signal trend without the noise, Excel offers some standard interpolation curves to add to a graph such as this one. One of these is a moving average, which is the black curve in Fig. 1. In the example the moving average is performed over 7 samples. Each point on the black curve is therefore formed as the sumproduct between seven samples of the signal in noise and what we can from now on think of as a digital filter[2] with a filter coefficients vector, L = [1/7 1/7 1/7 1/7 1/7 1/7 1/7]. For each sample on the black curve, the filter waveform is displaced to the right relative to the data and a new sumproduct is formed, which gives the average of 7 new samples[3]. Excel does not show the first 6 samples of the moving average, presumably because these samples are the averages of the first 1, 2, 3 … 6 samples, which differs from the promised 7. The whole process can be acknowledged to last longer than the original data, but this is also not shown in the Excel graph. The moving average is not actually zero before the filter vector

---

[1] It isn't. It's just some numbers on a computer, but I hope to inspire more interest in the example by pretending that it could be from a blue whale, *Baleanopterus musculus*. This is an example of a completely superfluous footnote; it's just there to keep you awake.

[2] This type of filter is called a Finite Impulse Response (FIR) filter. Remember this when you meet this abbreviation in the literature. The concept of impulse response is presented below.

[3] or rather, 1 new and 6 completely the same

overhangs the data by more than 6 samples, but apart from these omissions, the process of forming a moving average is actually identical to what we like to call convolution. In this example it is a convolution between our vector of filter coefficient, L and the noisy Sine.

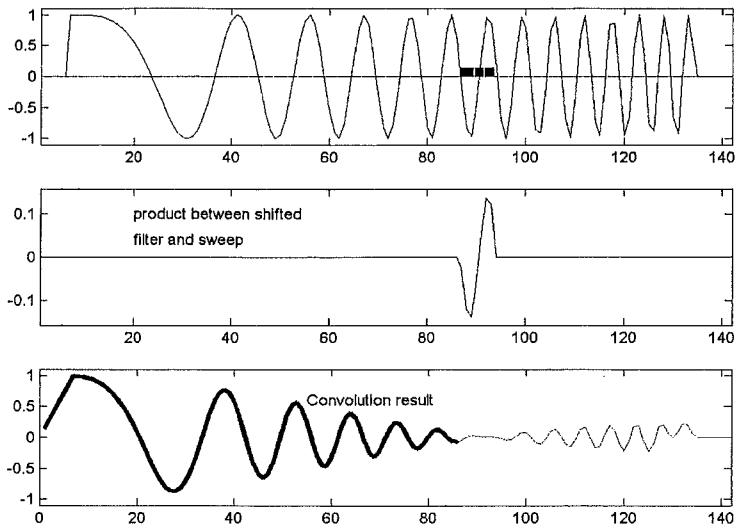

Fig 2 Step 86 in the running average process of a frequency sweep. The top panel shows the input waveforms, the sweep (blue line) and the digital filter (red bars). Middle panel shows the product between the two as it appears at this particular step in the process. The bottom trace then shows the sum of the products (blue dot) together with the history of such steps, which constitutes (a part of) what we should now start to get used to calling the convolution result (black + grey curves).

Refer now to Fig 2, which illustrates a single step in the same process, but with a frequency sweep as the filtered data instead of the Sine in Fig. 1. The swept signal starts with low frequencies and ends with higher frequencies so we can easily detect the effect of any filter operations performed on it. This is why we use this signal, here and in coming examples. It is apparent from the result that the running average is a low-pass filter; the lowest frequencies in the sweep pass the process unattenuated but the later parts of the sweep are reduced in the output as the frequency increases. If the data of interest, i.e. the biological (or other) signal within the noise has higher frequencies than the lowest part of the sweep shown, then this filter will not only remove noise but also parts of the signal[4].

That the moving average is a low-pass filter can also be acknowledging by noticing that the filter coefficient vector only contains positive values. If a part of the signal changes quickly (has a high frequency) above and below zero, the sumproduct between signal and filter coefficient

[4] If the period of the signal of interest is the same as the length of the averaging window, the filter removes ALL of it (in the example this happens around step 92). You may notice that in the later part of the filtered curve the amplitude goes back up a bit. This is because the properties of the moving average type of filter are not very good, and the rebound at frequencies higher than 1/7 is part of a sidelobe, which is phenomenon one tries to minimize by using filters with more cerebral effort invested in the design.

vector is close to zero[5].

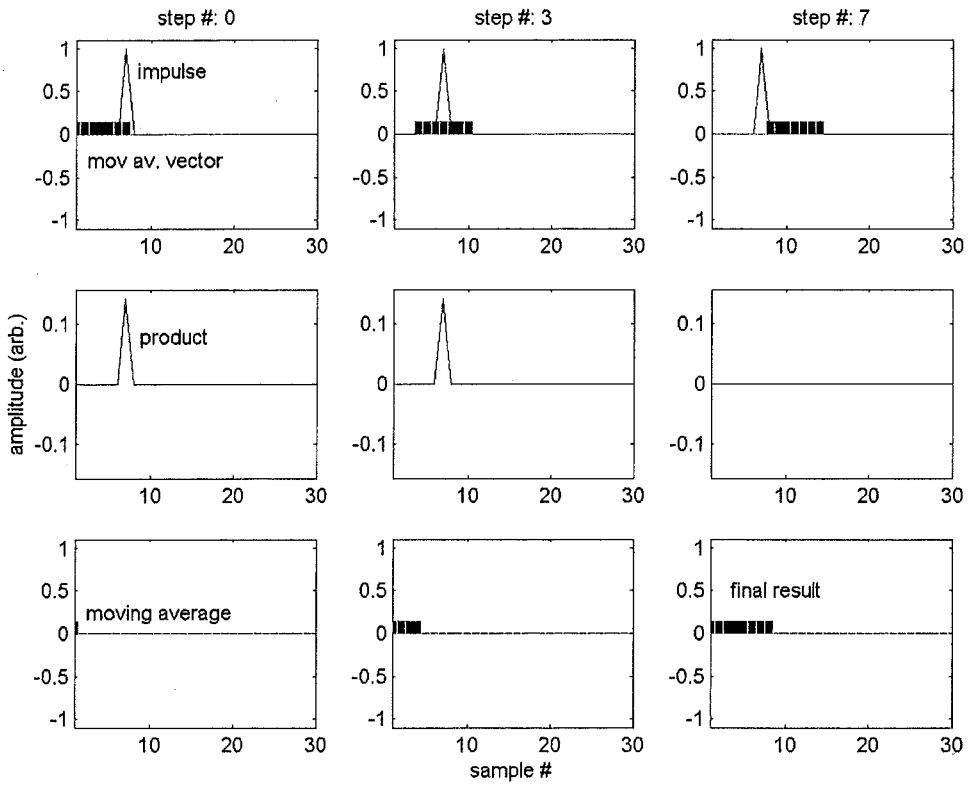

Fig 3 Stages in the convolution between an impulse (unit sample) and an averaging vector of filter coefficients. After seven steps of delaying, multiplying and summing, the output is identical to the filter coefficient vector that was used to make the moving average. This would of course also have been the case if the vector of filter coefficients were a more complex signal. The output is known as the impulse response of a system and for most systems we can use the output resulting from such an impulse to predict the response from any input by convolving the input signal with the impulse response.

Impulse response

Imagine owning a black box with an input and an output end. You may make the observation that it passes direct current unhindered but attenuates alternating current signals, and more so for higher frequencies. So you guess that it is a moving averager like the one in Figs 1 and 2. There is a simple way to find out if this is correct, namely by inputting a very short positive-going pulse[6]. By convolving with the impulse signal the output is simply the filter coefficient vector! We call this the impulse response[7]. Generally, if one wants to characterize a filter or other gadget, this is the way to go about it: input an impulse and collect the output. One can then predict the output resulting from any arbitrary input by convolving the input signals by the collected impulse response. The reaction of the moving average filter with filter coefficients, L to an impulse is shown in Fig. 3. The output is identical to L. You can often achieve the same by sampling the

---

[5] Example: Σ ([-1 1 -1 1 -1 1 -1 1] x [1/8 1/8 1/8 1/8 1/8 1/8 1/8 1/8]) = 0. Here I had to change the vector length to 8 from 7 to make this point. I can no longer recall the point of using the length 7 in the examples...

[6] The examples are all using notation only valid for digital signals, and really anal people will insist on calling a digital impulse a unit sample.

[7] Unit sample response for digital signals.

impulse response from an analog filter and convolve your sampled sound signal data with it.

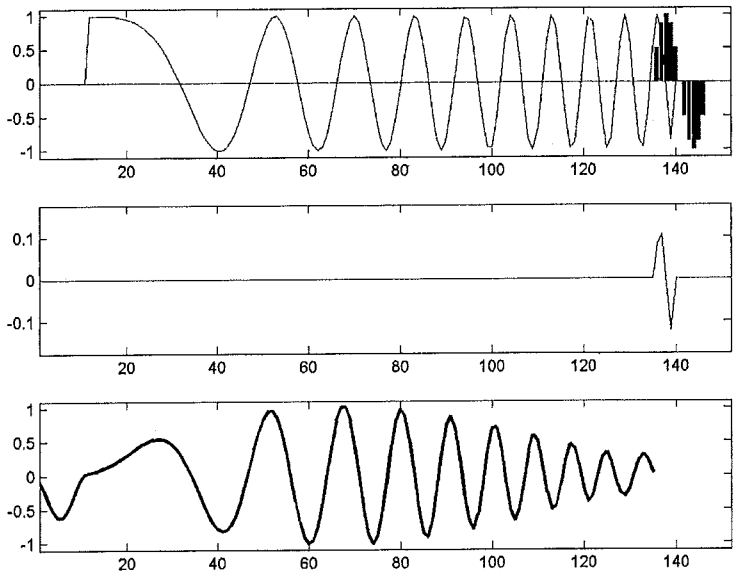

Fig 4 Late stage in the convolution between a sweep signal and a filter coefficient vector that forms a single cycle of Sine wave. The peak output amplitude occurs in the middle of the signal, so this is a band-pass filter, but not a particularly specific (narrow) one.

What happens if the filter coefficients are a more complex waveform than in the uniform example, L? In Fig. 4 the signal is the same sweep as in Fig. 2 but now the filter coefficient vector (=impulse response of the filter) is a single cycle of a sine, still 7 samples long. Notice that the amplitude of the output is maximal at a sample corresponding to some intermediate frequency within the sweep.

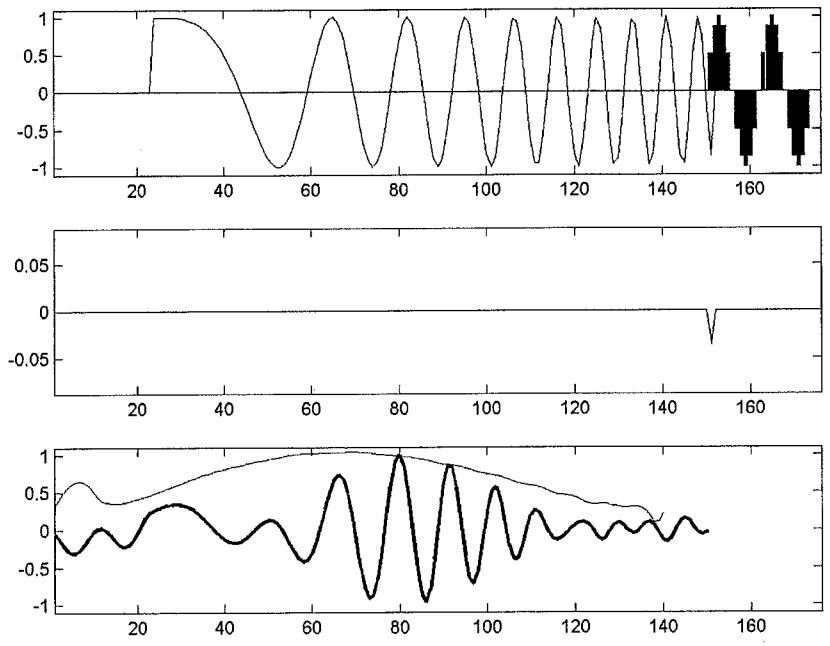

Fig 5 Like Fig. 4, but with the filter coefficients vector constaning two cycles of Sine, instead of one. This filter is more frequency specific. The blue line in the bottom trace shows the amplitude of the convolution result from the single cycle example in Fig. 4.

This is evidently a bandpass filter. In Fig. 5 is shown what happens if instead of one cycle we use two. Two effects of this change are evident: The output gets more time-local, but the overall process also lasts longer, because the filter coefficient vector gets longer. The convolution result is in fact the sum of the two input vectors – 1.

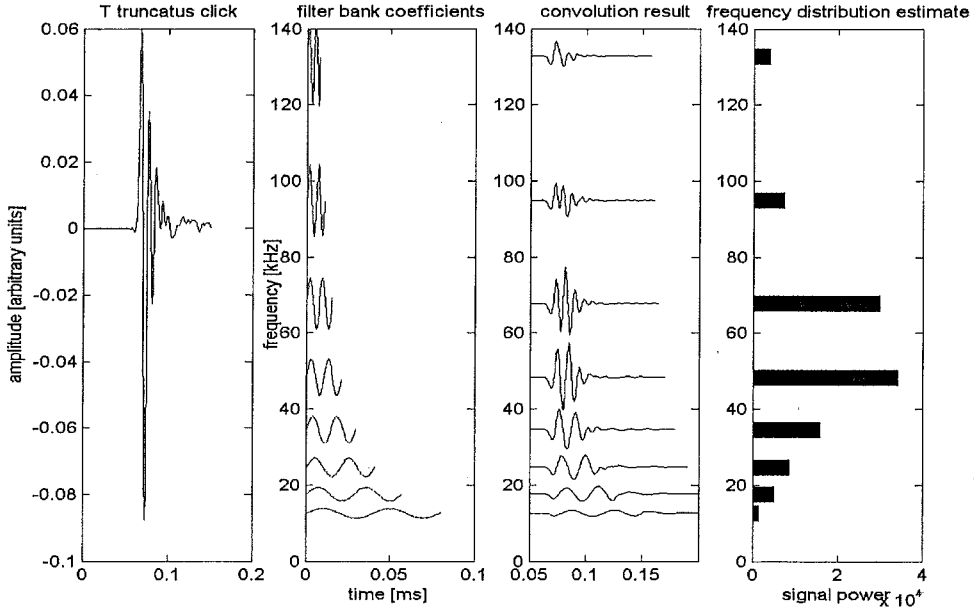

Fig 6 Many different frequencies of the two cycles that made up the filter in Fig. 5 constitute a filter bank. The bandpass filters are convolved with the input signal, in this case a bottle nosed dolphin echolocation click. By summing the squared convolution results, we can build a spectral density histogram, which helps us answer many questions. The center frequency of this click, for instance, was around 50 kHz.

If one wishes to analyze a given signal in terms of its frequency content such bandpass filters can come in quite handy, since they single out certain parts of the signal, depending of the frequencies in it, just as it happened with the sweep in Figs. 4 and 5. To break the signal down one must of course use several filters, each with a specific period (frequency) of the Sine cycles. This can then be used to make a systematic mapping of signal power as a function of frequency, frequency in the sense "the reciprocal of the period of the Sine wave making up the filter coefficient vector". For noise analysis this is a quite common approach[8].

Discrete Fourier transform

Taking the tendency with the more specific bandpass filters with more cycles in the filter coefficient vector to the extreme, we can see that only an infinitely long signal will produce an infinitely narrow filter, but luckily this is not a big problem, if we align the input signal and the a finite chunk of filter coefficient vector that is longer than the input signal. When we slide it for the convolution we let the filter vector disappear to the right as it enters on the left. The effect is the same as if the vector were indeed infinitely long - it bites its own tail!

[8] Although in that case the filters are more advanced than just two cycles of Sine wave.

What you can perhaps realize from this thought experiment is that nothing is actually going to happen to the amplitude of the convolution result as the two signals slide along each other and the sumproducts are computed for each step, see Fig. 7. The infinitely narrow filter only produces one frequency as the output. If instead of a Sine we had used a Cosine then the same would have happened but 90 degrees out of phase. In short we can skip the whole convolution story and simply do one sumproduct between signal and a Sine vector and a Cosine vector.

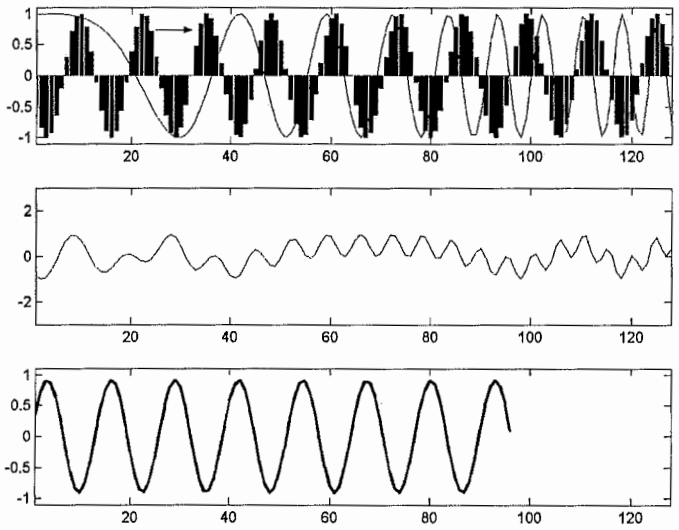

Fig 7 A single step in the convolution between an "infinite" Sine vector and the frequency sweep from Figs 3, 4 and 5. The output is a Sine with an amplitude that is dependent on the presence of that frequency in the signal. A Cosine will produce an identical result but 90 degrees out of phase. Since $Cos(x)^2+Sin(x)^2=1$ the signal's content of one frequency is completely summed up by these two values, the sumproduct with Cosine and the sumproduct with the Sine.

The frequency analysis is then almost complete. We usually define the Cosine sumproducts to be the real part of a Fourier transform for a given frequency and the sumproduct with the Sine to be the imaginary part in a complex number. This is just to make the notation shorter. The amplitude spectrum is then given by $(Real^2+Imag^2)^{½}$.

In Fig 8 is shown parts of the steps in forming the Fourier transform. It shows seven example Cosines to be multiplied with the input signal, after which the sum of the product is the real part of the Fourier transform at each of those frequencies. There is a Sine wave of the same frequency, which is not shown in the graph. The sumproduct of the signal with the Sines constitute the imaginary part. Here only the resulting amplitude is shown in the result.

Note that the spectrum is symmetrical around the frequency 0.5, the Nyquist frequency when using a sampling rate of 1. Often the frequencies above that frequency are not displayed as they are clearly redundant, but this data is handy (but not crucial) if the signal is to be transformed back again.

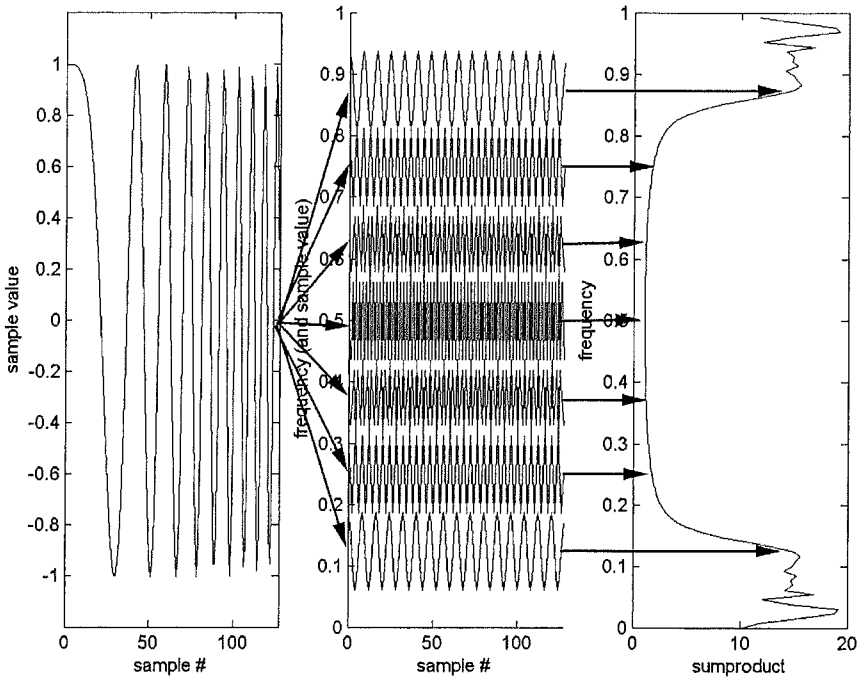

Fig 8. Elements of Fourier transform. Leftmost panel shows the sweep signal to be analyzed. The middle panel shows examples of the Cosines with which the signal is multiplied to find the real part of the transform. The rightmost panel shows the amplitude of the sums of the products between the example frequency waveforms. To find these values the sumproducts with Sine waves have been utilized too, which is not shown.

Inverse tranform

Usually one sees only the amplitude part of the spectrum, but the ratio of the imaginary part to the real part contains the information about where on the time axis the signal is. Often we need to change the signal in the frequency domain and then transform it back into the time domain. For this the actual values, not just the amplitudes are needed. The inverse transform is almost but not quite the same as the Fourier transform. One needs to change the sign of the Sine waves in the sumproducts for the imaginary part and there has to some amplitude scaling, but it is close enough for government work to say that the process of Fourier transformation is reciprocal. This is a really deep and fascinating fact. Just one example here at the end of how funny the reciprocity is: The faster the values of the signal wiggles up and down, the higher the frequency and the longer to the right on the frequency axis it is in the transform. But given the reciprocity the faster the real part of the Fourier transform wiggles up and down the farther to the right on the time axis it is, meaning it is has a longer delay!