Eur. J. Phys. **30** (2009) 1049–1062 doi:10.1088/0143-0807/30/5/013

Wavelet analyses and applications

Cristian C Bordeianu[1], **Rubin H Landau**[2] and **Manuel J Paez**[3]

[1] Faculty of Physics, University of Bucharest, Bucharest, RO 077125, Romania
[2] Department of Physics, Oregon State University, Corvallis, OR 97331, USA
[3] Department of Physics, University of Antioquia, Medellin, Colombia

E-mail: cristian.bordeianu@brahms.fizica.unibuc.ro, rubin@science.oregonstate.edu and mpaez@fisica.udea.edu.co

Received 27 May 2009, in final form 16 June 2009
Published 20 July 2009
Online at stacks.iop.org/EJP/30/1049

Abstract
It is shown how a modern extension of Fourier analysis known as wavelet analysis is applied to signals containing multiscale information. First, a continuous wavelet transform is used to analyse the spectrum of a nonstationary signal (one whose form changes in time). The spectral analysis of such a signal gives the strength of the signal in each frequency as a function of time. Next, the theory is specialized to discrete values of time and frequency, and the resulting discrete wavelet transform is shown to be useful for data compression. This paper is addressed to a broad community, from undergraduate to graduate students to general physicists and to specialists in other fields than wavelets.

1. Introduction

Fourier decomposition is a standard tool in physics and is excellent for analysing periodic signals whose functional forms do not change in time (*stationary* signals). However, if the signal is nonstationary, such as that in figure 1,

$$y(t) = \sin 2\pi t, \qquad \text{for} \quad 0 \leqslant t \leqslant 2, \tag{1}$$

$$= 5\sin 2\pi t + 10\sin 4\pi t, \qquad \text{for} \quad 2 \leqslant t \leqslant 8, \tag{2}$$

$$= 2.5\sin 2\pi t + 6\sin 4\pi t + 10\sin 6\pi t, \qquad \text{for} \quad 8 \leqslant t \leqslant 12, \tag{3}$$

then the standard Fourier analysis may tell us which frequencies are present, but it will not tell *when* the different frequencies were present. Clearly, if we want to obtain *time resolution* as well as frequency resolution, then we need to introduce another variable into the analysis, for otherwise the standard Fourier analysis has all of the constituent frequencies occurring simultaneously. In addition, because the basis set $e^{i\omega t}/\sqrt{2\pi}$ extends over an infinite time

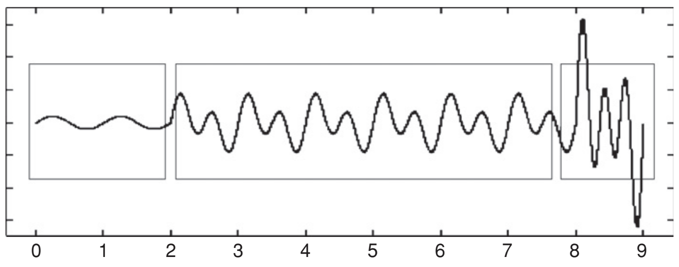

Figure 1. The time signal (1) containing an increasing number of frequencies as time increases. The boxes are possible placements of windows for short-time Fourier transforms.

domain with a constant amplitude, the deduction of the Fourier transform $Y(\omega)$, the amount of $e^{i\omega t}$ in a signal, requires integration over infinite time:

$$Y(\omega) = \int_{-\infty}^{+\infty} dt \frac{e^{-i\omega t}}{\sqrt{2\pi}} y(t) = \langle \omega | y \rangle. \tag{4}$$

We recognize the transform as the overlap of the basis function with signal y or the ω representation of the signal (the negative exponential appears because $\langle \omega |$ occurs and not $| \omega \rangle$). Accordingly, the value of $Y(\omega)$ at any one ω is correlated to its value at other ω's. This latter fact often makes it difficult to reconstruct a complicated signal without including a very large number of Fourier components, which in turn requires the lengthy computation of many components and the efficient storage of very large data files.

In this paper, we outline an extension of Fourier analysis known as *wavelet analysis*. The purpose of the paper is not to present new research results, but rather to introduce the basic ideas of wavelet analysis to a broader community and indicate how it overcomes some of the shortcomings of Fourier analysis by adding another variable to the theory. Indeed, wavelets have found value in areas as diverse as time-dependent quantum mechanics, brain waves and gravitational waves. Basically, rather than expanding a signal in terms of basis waves $e^{i\omega t}/\sqrt{2\pi}$ with infinite extent, we expand in terms of basis wave packets of finite extent, the so-called wavelets (examples in figure 2).

By centring different wavelets at different times (figure 3 (left)), we are able to deduce the time resolution of each frequency in the input signal. By adjusting the wavelets at each time to contain different frequencies (figure 3 (right)), we are able to deduce the frequency dependence of the signal at each time. And because the deduced wavelet components are sorted according to time and frequency without high correlation, for a given level of resolution required in the reconstructed signal, it is possible to eliminate the components that contribute only for higher resolution; this permits significant data compression (it is the mathematics behind JPEG-2000 standard). The subject of this paper is how this is done. In section 2, we provide some background information on short-time Fourier transforms, wave packets and filters. In section 3, we formulate the wavelet transform for continuous times and frequencies (CWT), and in section 4 we formulate the wavelet transform for discrete times and frequencies (DWT). In section 5, we explain the pyramid scheme implementation and in section 6 we calculate the Daub4 filter coefficients. While the CWT displays the basic principles of wavelet analysis in a straightforward way, it is not optimized for computation. The DWT is optimized for computation, but at the price of technical complications that some reader may find challenging on first reading.

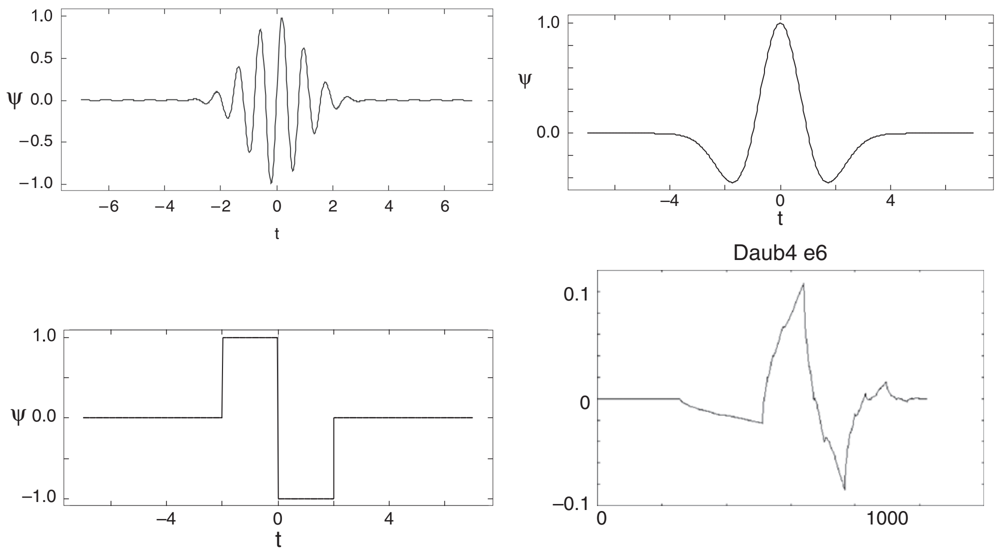

Figure 2. Four sample mother wavelets. Clockwise from top: Morlet (real part), Mexican hat, Daub4 e6 (explained later) and Haar. Wavelet basis functions are generated by scaling and translating these mother wavelets.

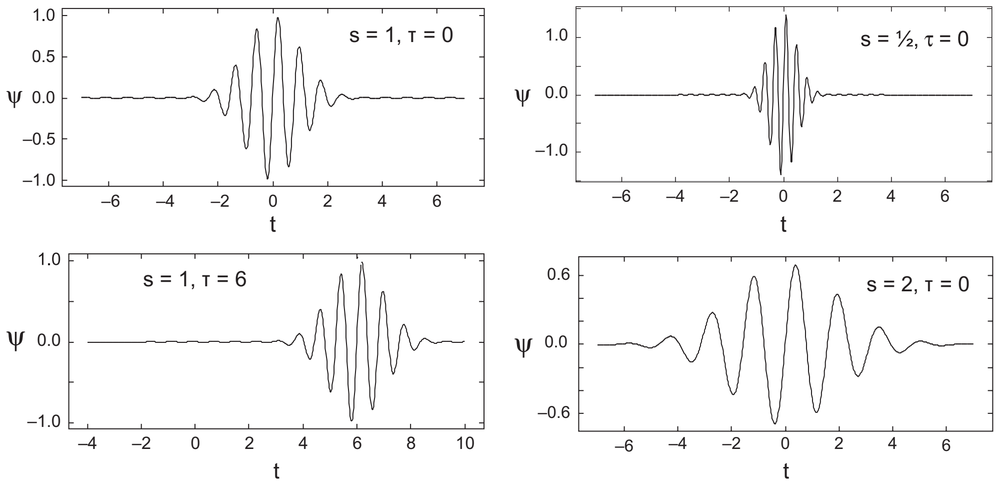

Figure 3. Four wavelet basis functions generated by scaling (s) and translating (τ) the oscillating Gaussian mother wavelet. Note how $s < 1$ is a wavelet with higher frequency, while $s > 1$ has a lower frequency than the $s = 1$ mother. Likewise, the $\tau = 6$ wavelet is just a translated version of the $\tau = 0$ one directly above it.

2. Short-time transforms and wavelets

As we have said, because the Fourier transform (4) extends over infinite time with a constant amplitude for the basis function $e^{i\omega t}/\sqrt{2\pi}$, if we apply it to a signal such as that in figure 1, in which there are different frequencies present at different times, we lose all information as to when in time a particular frequency occurred. An obvious solution to this problem, known as the *short-time Fourier transform*, was a precursor to wavelet analysis. An example of it would be to perform a separate transform for each of the boxes or windows shown in figure 1, and thus obtain a different spectrum for each time interval. Rather than 'chop up' a signal by hand for each case, we introduce a *window function* $w(t)$ that differs from 1 for only a short period of time or lifetime Δt around $t = 0$ (imagine a box or a Gaussian). Then the function $w(t - \tau)$ would be a window centred at time τ. Using this window function, we define the short-time Fourier transform as

$$Y^{(\mathrm{ST})}(\omega, \tau) = \int_{-\infty}^{+\infty} \mathrm{d}t \frac{\mathrm{e}^{-\mathrm{i}\omega t}}{\sqrt{2\pi}} w(t - \tau) y(t), \tag{5}$$

where the exact form of the window function is not essential as long as it has a short lifetime and so can be thought of as a transparent box on an opaque background. In (5), the value of the translation time τ corresponds to the location of the window w over the signal, so that any signal within the window width is transformed, while the signal lying outside the window does not enter. Evidently, the short-time transform is a function of two variables, and so a surface or 3D plot is needed to view it as a function of both ω and τ.

In the short-term Fourier transform (5), we multiply a window function $w(t-\tau)$, which is finite in time about time τ, by the exponential $\mathrm{e}^{-\mathrm{i}\omega t}$, which is oscillatory in time with frequency ω. In wavelet analysis, we combine both types of behaviour into a wavelet that has finite width in time Δt and also oscillates, but may have a variety of forms [1]. For example, a wavelet may be an oscillating Gaussian (Morlet: top left in figure 3),

$$\Psi(t) = \mathrm{e}^{2\pi \mathrm{i}t}\, \mathrm{e}^{-t^2} = (\cos 2\pi t + \mathrm{i} \sin 2\pi t)\, \mathrm{e}^{-t^2}, \tag{6}$$

the second derivative of a Gaussian (Mexican hat, top right in figure 3),

$$\Psi(t) = -\frac{\mathrm{d}^2}{\mathrm{d}t^2}\, \mathrm{e}^{-t^2} = (1 - t^2)\, \mathrm{e}^{-t^2}, \tag{7}$$

an up-and-down step function (lower left in figure 3) or a fractal-like shape (bottom right).

Of course, oscillatory functions that are nonzero for finite lengths of time Δt are just what are commonly called *wave packets*, and are familiar from time-dependent quantum mechanics and signal processing. It is well known [2] that the Fourier transform of a wave packet is a pulse in the frequency domain of width $\Delta\omega$, with the time and frequency widths related by the so-called *Heisenberg uncertainty principle*,

$$\Delta t \Delta\omega \geqslant 2\pi. \tag{8}$$

Accordingly, as a wavelet is made more localized in time (smaller Δt), it becomes less localized in frequency (larger $\Delta\omega$), and vice versa. For example, the function $y(t) = \sin \omega_0 t$ is completely localized in frequency, $\Delta\omega = 0$, and so has an infinite extent in time, $\Delta t \simeq \infty$, while the wavelets in figure 2 have finite $\Delta\omega$ and Δt.

Let us now take this one step further. We want to use wavelet basis functions that oscillate in time and also contain a range of frequencies. While the example functions in (6) and (7) are evidently such functions, it is not clear how to use them as a set of basis functions. More concretely, the basis functions for the short-term Fourier transform in (5) $\exp(-\mathrm{i}\omega t) w(t - \tau)/\sqrt{2\pi}$ contain both a frequency ω and time variable τ that also appear on the lhs, while the wavelets $\Psi(t)$ we have given so far are just functions of time. Actually, the wavelets $\Psi(t)$ we have given as examples so far are what are called *mother wavelets* or *analysing functions*, and each is used to generate an entire set of basis functions or daughter wavelets $\psi_{s,\tau}(t)$ via a simple, yet clever, change of variable that scales and translates the mother wavelet:

$$\psi_{s,\tau}(t) = \frac{1}{\sqrt{s}} \Psi\left(\frac{t - \tau}{s}\right). \tag{9}$$

For example,

$$\Psi(t) = \sin(8t)\,\mathrm{e}^{-t^2/2} \quad \Rightarrow \quad \psi_{s,\tau}(t) = \frac{1}{\sqrt{s}} \sin\left[\frac{8(t-\tau)}{s}\right] \mathrm{e}^{-(t-\tau)^2/2s^2}. \tag{10}$$

Translating and scaling one of these *mother wavelets* generate an entire set of *child wavelet* basis functions, with each individual function covering a different frequency range at a different time. We see that larger or smaller values of s expand or contract the mother wavelet, while different values of τ shift the centre of the wavelet. Because the wavelets are inherently oscillatory, the scaling leads to the same number of oscillations occurring in different time spans, which is equivalent to having basis states with differing frequencies. Here the division by $\sqrt{s}$ is made to ensure that there is equal 'power' (or energy or intensity) in each region of s, although other normalizations can also be found in the literature.

Note that we have made a change in notation that is standard for wavelets, but somewhat confusing at first. Rather than using the frequency variable ω we have switched to the *scale* variable s, which has the dimension of time. You may think of the two as simply inversely related:

$$\omega = \frac{2\pi}{s}, \qquad s = \frac{2\pi}{\omega} \quad \text{(scale–frequency relation)}. \tag{11}$$

If we are interested in the time *details* of a signal, we would say that we are interested in what is happening at small values of the scale s; equation (11) indicates that small values of s correspond to high-frequency components of the signal. That being the case, *the time details of the signal are in the high-frequency, or low-scale, components*.

3. The continuous wavelet transform

The wavelet transform $Y(s,t)$ of a time signal $y(t)$ is defined much like the Fourier transform (4), but with a wavelet basis rather than an exponential:

$$Y(s,\tau) = \int_{-\infty}^{+\infty} \mathrm{d}t\, \psi^*_{s,\tau}(t) y(t) = \int_{-\infty}^{+\infty} \mathrm{d}t \frac{1}{\sqrt{s}} \Psi\left(\frac{t-\tau}{s}\right) y(t) = \langle s, t|y\rangle. \tag{12}$$

Because each wavelet is localized in time, each acts as its own window function. Because each wavelet is oscillatory, each contains its own small range of frequencies or scale s values. Equation (12) says that the wavelet transform $Y(s,\tau)$ is a measure of the amount of basis function $\psi_{s,\tau}(t)$ present in signal $y(t)$. The τ variable indicates the time portion of the signal being decomposed, while the scale variable s is equivalent to the frequency present during that time. Although we do not derive it here, the inverse transform is

$$y(t) = \frac{1}{C} \int_{-\infty}^{+\infty} \mathrm{d}\tau \int_0^{+\infty} \mathrm{d}s \frac{\psi^*_{s,\tau}(t)}{s^{3/2}} Y(s,\tau), \tag{13}$$

where the normalization constant C depends on the specific wavelet class used. In figure 4, we show a 2D and a 3D representation of the results of evaluating numerically the wavelet transform (12) for the sample signal (1). The Java code used for the transform is from our textbook [4]. Recall that this signal contains one frequency during the first window of time, two frequencies during the second time window and three frequencies for the last window. Indeed, at short times τ we see a single hump at high scale $s \simeq 0.8$ (low frequency); at latter times this is joined by another hump at smaller s (higher frequency), with a third peak entering at the latest times with a yet smaller s value (higher frequency). Although it is hard to evaluate relative heights from this plot, the relative strengths of the three frequency components appear to be in agreement with (1).

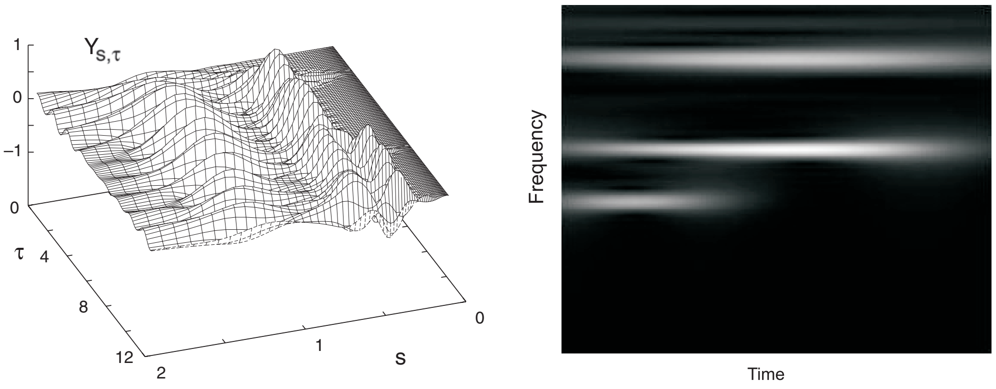

Figure 4. Left: the continuous wavelet spectrum 3D obtained by analysing the input signal with Morlet wavelets. Observe how at small values of time τ there is predominantly one frequency present; how a second, higher frequency (smaller-scale) component enters at intermediate times and how at larger times, a still higher frequency component enters. Right: the spectrogram 2D of the input signal makes it clear how there is one, two and then three frequencies present.

4. The discrete wavelet transform

As is true for discrete Fourier transforms, if a time signal y is measured at only N discrete times,

$$y(t_m) \equiv y_m, \qquad m = 1, \ldots, N, \tag{14}$$

then we can determine only N independent components of the transform Y. The *discrete wavelet transform* (DWT) is a clever way to compute only the N independent components required to reproduce the input signal, consistent with the uncertainty principle, by evaluating the transforms at discrete values for the scaling parameter s and the time translation parameter τ:

$$\psi_{j,k}(t) = \frac{\Psi[(t - k2^j)/2^j]}{\sqrt{2^j}} \equiv \frac{\Psi(t/2^j - k)}{\sqrt{2^j}}, \tag{15}$$

$$s = 2^j, \qquad \tau = \frac{k}{2^j}, \qquad k, j = 0, 1, \ldots. \tag{16}$$

Note that in contrast to the continuous wavelet transform (12), the DWT (15) employs a discrete set of basis wavelets denoted by the integer subscripts j and k, whose maximum values are yet to be determined. Also, we have assumed that the total time interval $T = 1$, so that time is always measured in integer values. This choice of s and τ based on powers of 2 is called a *dyadic grid* arrangement and is seen to automatically perform the scalings and translations at the different timescales that are at the heart of wavelet analysis (some references scale down with increasing j, in contrast to our scaling up). We now use the trapezoid integration rule to express the DWT as the simple sum

$$Y_{j,k} = \int_{-\infty}^{+\infty} \mathrm{d}t\, \psi_{j,k}(t) y(t) \simeq \sum_m \psi_{j,k}(t_m) y(t_m) h \quad \text{(DWT)}. \tag{17}$$

For an orthonormal wavelet basis, then the inverse discrete transform is [1]

$$y(t) = \sum_{j,k=-\infty}^{+\infty} Y_{j,k} \psi_{j,k}(t) \quad \text{(inverse DWT)}. \tag{18}$$

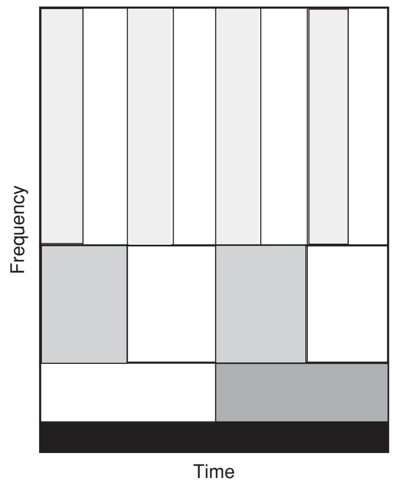

Figure 5. Time and frequency resolutions. Each box represents an equal portion of the time–frequency plane but with different proportions of time and frequency.

It is interesting to note that this inversion will exactly reproduce the input signal at the N input points if we sum over an infinite number of wavelets, but will be less exact in practical calculations.

Note in (15) and (17) that up until this point we have kept the time variable t in the wavelet basis functions continuous, even though s and τ are discrete. This is useful in establishing the orthonormality of the basis functions,

$$\int_{-\infty}^{+\infty} \mathrm{d}t\, \psi^*_{j,k}(t)\psi_{j',k'}(t) = \delta_{jj'}\delta_{kk'}, \tag{19}$$

where $\delta_{m,n}$ is the Kronecker delta function. Having $\psi_{j,k}(t)$'s normalized to 1 means that each wavelet basis has 'unit energy'; being orthogonal means that each basis function is independent of the others. Because wavelets are localized in time, the different transform components have low levels of correlation with each other. Altogether, this leads to efficient and flexible data storage.

The use of a discrete wavelet basis makes it clear that we need only sample the input signal at the discrete values of time determined by the integers j and k. In general, you want time steps that sample the signal at enough times in each interval to obtain a reconstructed signal with a predetermined level of precision. A rule of thumb is to start with 100 steps to cover each major feature. Ideally, the needed times correspond to the times at which the signal was sampled, although this may require some forethought.

Consider figure 5. We measure a signal at a number of discrete times within the intervals (k or τ values) corresponding to the vertical columns of fixed width along the time axis. For each time interval, we want to sample the signal at a number of scales (frequencies or j values). However, the basic mathematics of Fourier transforms indicates that the width Δt of a wave packet $\psi(t)$ and the width $\Delta\omega$ of its Fourier transform $Y(\omega)$ are related by an uncertainty principle

$$\Delta\omega\Delta t \geqslant 2\pi. \tag{20}$$

This relation constrains the number of times we can meaningfully sample a signal in order to determine a number of Fourier components. So while we may want a high-resolution

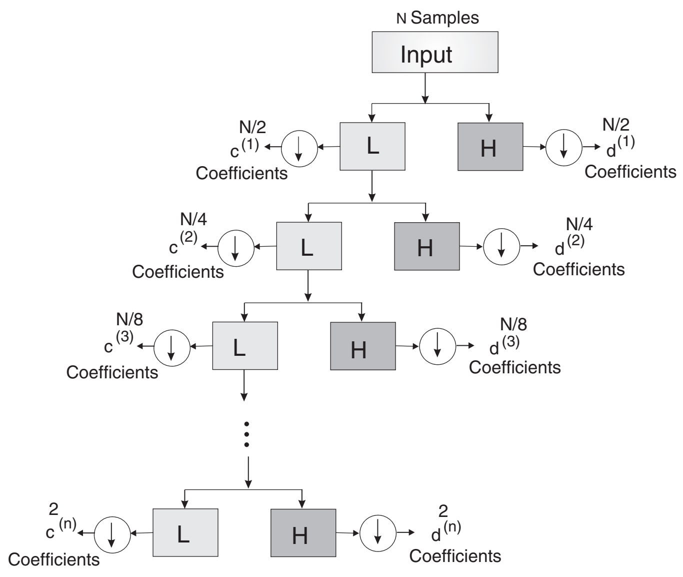

Figure 6. A signal is processed by high (*H*) and low (*L*) band filters, and the outputs are downsampled (reduced by a factor of 2) by keeping every other point. The filtering continues until there are only two *H* points and two *L* points. The total number of output data equals the total number of signal points.

reproduction of our signal, we do not want to store more data than are needed to obtain that reproduction. If we sample the signal for times centred about some τ in an interval of width $\Delta\tau$ (figure 5) and compute the transform at a number of scales s or frequencies $\omega = 2\pi/s$ covering a range of height $\Delta\omega$, then the relation between the height and width is restricted by the uncertainty relation, which means that each of the rectangles in figure 5 has the same area $\Delta\omega\Delta t = 2\pi$. The increasing heights of the rectangles at higher frequencies mean that a larger range of frequencies should be sampled at each time as the frequency increases. The inherent assumption in wavelet analysis is that the low-frequency components provide the gross or *smooth* outline of the signal which, being smooth, does not require much detail, while the high-frequency components give the details of the signal over a short time interval and so require many components in order to record these details with high resolution.

Industrial-strength wavelet analyses do not compute explicit integrals but instead apply a filtering technique known as *multiresolution analysis* (figure 6). It is a type of *pyramid algorithm* that takes the N sample values of a signal and passes it successively through a number of *filters*. Since each filter is equivalent to a DWT, the final output is the transform at all scales and times. Here we define a filter as a device that converts an input signal $f(t)$ to an output signal $g(t)$ via an integration over the signal:

$$g(t) = \int_{-\infty}^{+\infty} \mathrm{d}\tau\, f(\tau)h(t-\tau) = f(t) * h(t). \tag{21}$$

Here, the function $h(t)$ is called the *response* or *transfer function* of the filter and the operation in (21) is called a *convolution* and is denoted by an asterisk *. Equation (21) states that the output of a filter equals the input convoluted with the transfer function, while the *convolution* *theorem* states that the Fourier transform of the convolution $g(t)$ is proportional to the product of the transforms of $f(t)$ and $h(t)$:

$$G(\omega) = \sqrt{2\pi}\, F(\omega) H(\omega). \tag{22}$$

A comparison of the definition of a filter to the definition of a wavelet transform (12) shows that transforming a signal is equivalent to filtering it, with each filter outputting a different scale. The explicit output is the sum of the products of integration weights with the value of the wavelets at the integration points, and each product is called a filter coefficient c_i. Therefore, rather than tabulating explicit wavelet functions, a set of filter coefficients is all that is needed to perform discrete wavelet transforms.

5. Pyramid scheme implementation

The pyramid algorithm decomposes a signal into low-frequency or *smooth* components s_i and high-frequency or *detailed* components d_i. Because high-resolution reproduction requires more information about the details in a signal than about its gross shape, most of the smooth components can be discarded and thus the data compressed. Furthermore, because the components of different resolutions are independent of each other, it is possible to adjust the resolution by changing the number of high resolution components stored. So when we look at figure 6 we can think of each L or H filter as separating out the smooth and detailed components of the signal respectively, with repeated filterings effectively lowering the scale to get more detailed information (the reason this technique is called multiresolution analysis). Each $\downarrow$ filters out half of the signal, a process known as *factor-of-2 decimation*, and this produces the data compression. The end result is a vector with the components of the wavelet transform.

In practice, the filters L and H are filter matrices that multiply the input vector (we discuss some in the following section). This is followed by a reduction of the output by one-half, in which the low-frequency parts are discarded, and then a reordering of the output so that it may be filtered again. For example, let us consider a signal vector of length $N = 8$, which would also correspond to the last two steps in the pyramid algorithm:

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \\ y_8 \end{pmatrix} \xrightarrow{\text{filter}} \begin{pmatrix} s_1^{(1)} \\ d_1^{(1)} \\ s_2^{(1)} \\ d_2^{(1)} \\ s_3^{(1)} \\ d_3^{(1)} \\ s_4^{(1)} \\ d_4^{(1)} \end{pmatrix} \xrightarrow{\text{order}} \begin{matrix} \begin{pmatrix} s_1^{(1)} \\ s_2^{(1)} \\ s_3^{(1)} \\ s_4^{(1)} \end{pmatrix} \\ \begin{pmatrix} d_1^{(1)} \\ d_2^{(1)} \\ d_3^{(1)} \\ d_4^{(1)} \end{pmatrix} \end{matrix} \begin{matrix} \xrightarrow{\text{filter}} \\ \\ \\ \\ \end{matrix} \begin{matrix} \begin{pmatrix} s_1^{(2)} \\ d_1^{(2)} \\ s_2^{(2)} \\ d_2^{(2)} \end{pmatrix} \\ \begin{pmatrix} d_1^{(1)} \\ d_2^{(1)} \\ d_3^{(1)} \\ d_4^{(1)} \end{pmatrix} \end{matrix} \begin{matrix} \xrightarrow{\text{order}} \\ \\ \\ \\ \end{matrix} \begin{matrix} \begin{pmatrix} s_1^{(2)} \\ s_2^{(2)} \\ d_1^{(2)} \\ d_2^{(2)} \end{pmatrix} \\ \begin{pmatrix} d_1^{(1)} \\ d_2^{(1)} \\ d_3^{(1)} \\ d_4^{(1)} \end{pmatrix} \end{matrix}. \tag{23}$$

The first filtering leads to an output vector with the smooth and detailed components intermixed. It gets reordered into the lower vector of just detailed components, which is saved, and a smaller vector of first-order smoothed components, which then get filtered again. This second filtering leads to intermixed second-order filtered components, which then get reordered into just smooth and detailed components. The process ends when there are just two smooth components left, as we have here.

As a realistic illustration, imagine that we have sampled the chirp signal $y(t) = \sin(60t^2)$ at 1024 times. The successive filtering of this signal is illustrated schematically as a passage from the top to the bottom of figure 6, or with actual data in figure 7. First, the original 1024 signal samples are filtered and ordered into 512 detailed and 512 smooth components. All detailed components $\{d_i^{(1)}\}$ are saved, and the remaining 512 once-filtered smooth components $\{s_i^{(1)}\}$ are processed further. This step in the process is known as *downsampling*. We see in figure 7 that at this stage the smooth components still contain many high-frequency parts, while the detailed components are smaller in magnitude. These two sets are shown below the original signal. In the next level down, the 512 smooth components $\{s_i^{(1)}\}$ are filtered and ordered. The resulting 256, twice-filtered detailed components $\{d_i^{(2)}\}$ are saved, while the 256 smooth components $\{s_i^{(2)}\}$ are further processed, with these components displayed lower down in figure 6. The resulting output from each successive step is similar, but with coarser features for the smooth coefficients and larger values for the details. Note that in the upper graphs we have connected the points to make the output look continuous, while in the lower graphs, with fewer points, we have plotted the output as histograms to make the points more evident. The process continues until there are only two smooth and two detailed coefficients left. Since this last filtering is done with the broadest wavelet, it is of the lowest resolution and therefore requires the least information. When we are done, there will be a total of 512 + 256 + 128 + 64 + 32 + 16 + 4 = 1024 transform values saved, the same number as the number of time signals measured.

To reconstruct the original signal from the 1024 transform values (called signal *synthesis*), a reversed process is followed. We essentially follow figure 6; only now the directions of all the arrows are reversed and the inverses to the filter matrices are used. We begin with the last sequence of four coefficients, upsample them, pass them through inverse filters to obtain new levels of coefficients and repeat until all of the $N = 1024$ values of the original signal are recovered. If the data were compressed by eliminating some of the higher order detailed components, then we would start the process partially up the pyramid.

6. Daubechies wavelets filtering

We have spoken about how the discrete wavelet transformation can be implemented as digital filtering, with the filtering being simply a matrix multiplication of the signal vector by an appropriate filter matrix. We now indicate how one of the standard filter matrices, corresponding to the Daubechies wavelet bases, was determined by the Belgian mathematician Ingrid Daubechies in 1988 [3]. To keep things simple, we discuss just the Daub4 class containing the four coefficients c_0, c_1, c_2 and c_3. We deduce how these coefficients can be used to filter just four input signal values $\{y_1, y_2, y_3, y_4\}$. We have already seen that the pyramid algorithm for determining a DWT is based on a series of high- and low-pass filters, so we start by trying to express these two operations in terms of the filter coefficients. The basic idea is that replacing each signal value by the average of the values surrounding it tends to smooth the signal, and thus acts as the low-pass filter L. Likewise, replacing each signal value by the differences from surrounding values tends to emphasize the variation or details in the signal, and thus acts as the high-pass filter H. For example, if the signal were a sawtooth function with sharp points, the average would be rounded. Likewise, if the signal were constant, the differences would all vanish, as there are no details in the signal. A choice of filter coefficients that do this is

$$L = (+c_0 \quad +c_1 \quad +c_2 \quad +c_3) \tag{24}$$

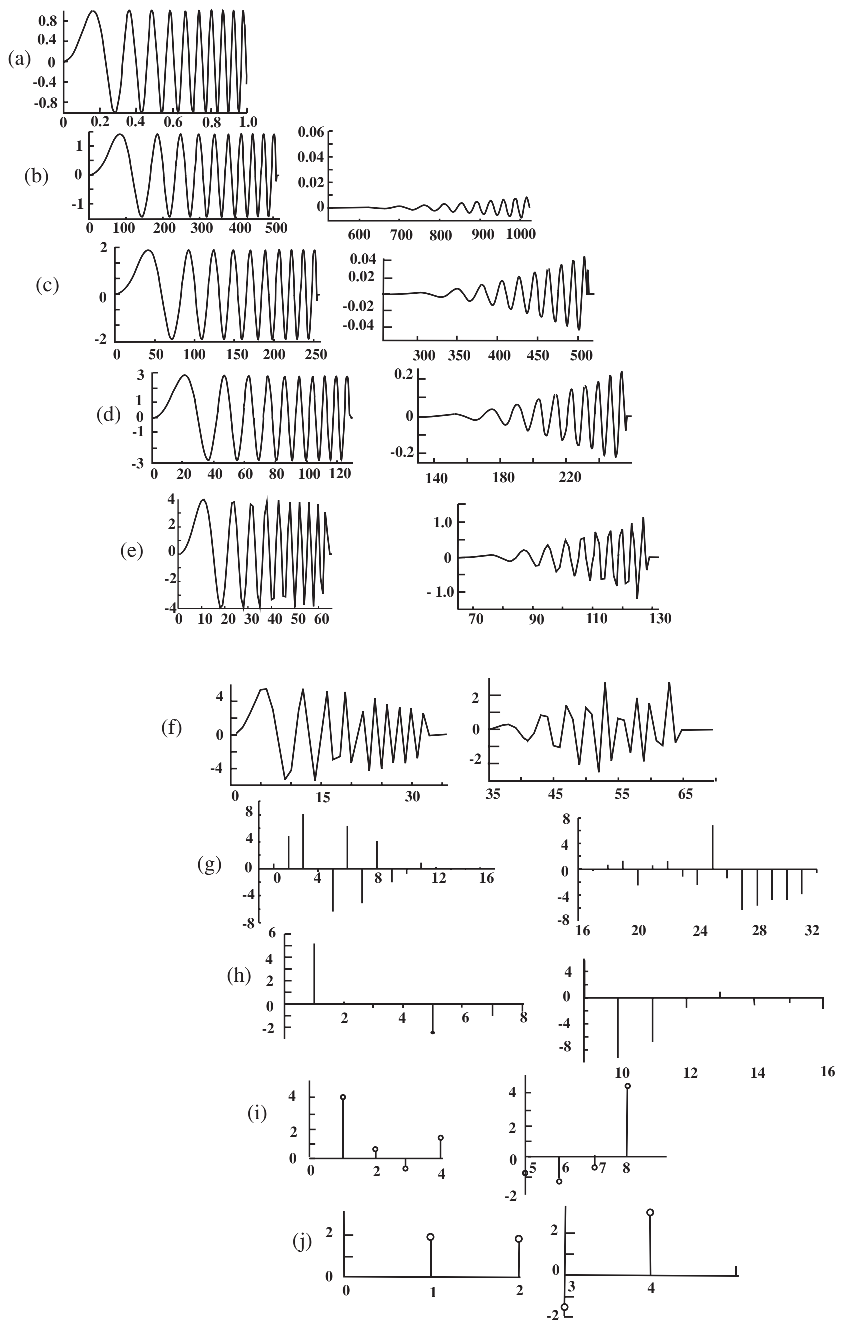

Figure 7. The filtering of the original signal at the top goes through the pyramid algorithm and produces the outputs shown, in successive passes. The sampling is reduced by a factor of 2 in each step. Note that in the upper graphs we have connected the points to make the output look continuous, while in the lower graphs, with fewer points, we have plotted the output as histograms to make the points more evident. The subimages labelled by (a)–(j) correspond to the original figure, 1024, 512, 256,128, 64, 32, 16, 8 and 4 samples.

$$H = (+c_3 \quad -c_2 \quad +c_1 \quad -c_0). \tag{25}$$

To see how this works, we form an input vector by placing the four signal elements in a column and then multiply the input by L and H:

$$L\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix} = (+c_0 \quad +c_1 \quad +c_2 \quad +c_3)\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix} = c_0y_0 + c_1y_1 + c_2y_2 + c_3y_3,$$

$$H\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix} = (+c_3 \quad -c_2 \quad +c_1 \quad -c_0)\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix} = c_3y_0 - c_2y_1 + c_1y_2 - c_0y_3.$$

Thus L acting on the signal vector is a single number that is the desired weighted average of the four input signal elements, while H acting on the signal vector would be small if the signal were constant. To determine the values of the filter coefficients, we need to set up an entire filter matrix and impose further requirements. If we want the output of the filtering process Y to contain the same number of elements as the input (four y's in this case), we just stack the L and H filters together:

$$\begin{pmatrix} Y_0 \\ Y_1 \\ Y_2 \\ Y_3 \end{pmatrix} = \begin{pmatrix} L \\ H \\ L \\ H \end{pmatrix}\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} c_0 & c_1 & c_2 & c_3 \\ c_3 & -c_2 & c_1 & -c_0 \\ c_2 & c_3 & c_0 & c_1 \\ c_1 & -c_0 & c_3 & -c_2 \end{pmatrix}\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix}. \tag{26}$$

Next we observed that a transform is equivalent to a rotation from the time domain to the frequency domain, yet rotations are described by orthogonal matrices, that is, matrices whose inverses are equal to their transposes. Accordingly, we require the filter matrix to be orthogonal, which means that the 4×4 filter matrix times its transpose equals the identity matrix:

$$\begin{pmatrix} c_0 & c_1 & c_2 & c_3 \\ c_3 & -c_2 & c_1 & -c_0 \\ c_2 & c_3 & c_0 & c_1 \\ c_1 & -c_0 & c_3 & -c_2 \end{pmatrix}\begin{pmatrix} c_0 & c_3 & c_2 & c_1 \\ c_1 & -c_2 & c_3 & -c_0 \\ c_2 & c_1 & c_0 & c_3 \\ c_3 & -c_0 & c_1 & -c_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$\Rightarrow \quad c_0^2 + c_1^2 + c_2^2 + c_3^2 = 1, \qquad c_2c_0 + c_3c_1 = 0. \tag{27}$$

Two equations in four unknowns are not enough for a unique solution, so we now include the further requirement that the detailed filter H must output a zero if the input is constant or linearly increasing vectors:

$$(y_0 \quad y_1 \quad y_2 \quad y_3) = (1 \quad 1 \quad 1 \quad 1) \quad \text{or} \quad (0 \quad 1 \quad 2 \quad 3). \tag{28}$$

This is equivalent to demanding that the moments up to order p are zero, that is, we have an 'approximation of order p'. Explicitly,

$$H(y_0 \quad y_1 \quad y_2 \quad y_3) = H(1 \quad 1 \quad 1 \quad 1) = H(0 \quad 1 \quad 2 \quad 3) = 0,$$

$$\Rightarrow \quad c_3 - c_2 + c_1 - c_0 = 0, \quad 0 \times c_3 - 1 \times c_2 + 2 \times c_1 - 3 \times c_0 = 0,$$

$$\Rightarrow \quad c_0 = \frac{1+\sqrt{3}}{4\sqrt{2}} \simeq 0.483, \qquad c_1 = \frac{3+\sqrt{3}}{4\sqrt{2}} \simeq 0.836, \tag{29}$$

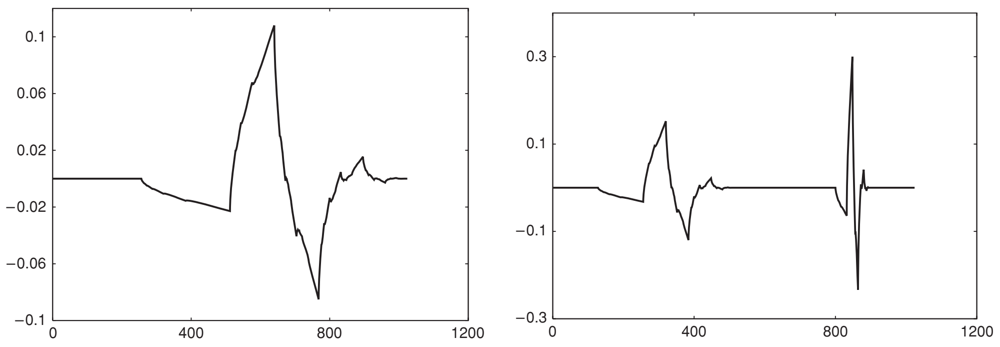

Figure 8. Left: the Daub4 e6 wavelet constructed by inverse transformation of the wavelet coefficients. Right: the sum of Daub4 e10 and Daub4 1e58 wavelets of different scales and time displacements.

$$c_2 = \frac{3-\sqrt{3}}{4\sqrt{2}} \simeq 0.224, \qquad c_3 = \frac{1-\sqrt{3}}{4\sqrt{2}} \simeq -0.129. \tag{30}$$

We now have our basic Daub4 filter coefficients. They can be used to process signals with more than four elements by creating a square filter matrix of the needed dimension that repeats these elements by placing the row versions of L and H along the diagonal, with successive pairs displaced two columns to the right. For example, for eight elements,

$$\begin{pmatrix} Y_0 \\ Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \\ Y_5 \\ Y_6 \\ Y_7 \end{pmatrix} = \begin{pmatrix} c_0 & c_1 & c_2 & c_3 & 0 & 0 & 0 & 0 \\ c_3 & -c_2 & c_1 & -c_0 & 0 & 0 & 0 & 0 \\ 0 & 0 & c_0 & c_1 & c_2 & c_3 & 0 & 0 \\ 0 & 0 & c_3 & -c_2 & c_1 & -c_0 & 0 & 0 \\ 0 & 0 & 0 & 0 & c_0 & c_1 & c_2 & c_3 \\ 0 & 0 & 0 & 0 & c_3 & -c_2 & c_1 & -c_0 \\ c_2 & c_3 & 0 & 0 & 0 & 0 & c_0 & c_1 \\ c_1 & -c_0 & 0 & 0 & 0 & 0 & c_3 & -c_2 \end{pmatrix} \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{pmatrix}. \tag{31}$$

Note that in order not to lose any information, the last pair on the bottom two rows is wrapped over to the left. If you perform the actual multiplications indicated in (31), you will note that the output has successive *smooth* and *detailed* information.

The time dependences of two Daub4 wavelets are displayed in figure 8. On the left is the wavelet for coefficient 6 (thus the e6 notation), and on the right is the sum of two wavelets corresponding to the coefficients 10 and 58. We see that the two wavelets have differing levels of scale as well as differing time positions. So even though the time dependence of the wavelets is not evident when wavelet (filter) coefficients are used, it is there. To obtain the time dependences from our filter coefficients, first imagine that an elementary wavelet $y_{1,1}(t) \equiv \psi_{1,1}(t)$ is input into the filter. This should result in a transform $Y_{1,1} = 1$. Inversely, we obtain $y_{1,1}(t)$ by applying the inverse transform to a Y vector with a 1 in the first position and zeros in all the other positions. Likewise, the ith member of the Daubechies class is obtained by applying the inverse transform to a Y vector with a 1 in the ith position and zeros in all the other positions. Programmes to perform DWTs and their inverse are available from our text [4] and on the Web[4].

7. Summary and conclusion

Physics textbooks often include significant coverage of Fourier analysis, but little if any coverage of the other analysis tools that are now used for digital signal processing, such as principal component analysis, linear time-invariant system theory and wavelet analysis. In this paper we have discussed continuous and discrete wavelet analyses, and have given an example of each. We have shown how to expand a time signal in terms of a wavelet basis set, where each wavelet is an oscillatory wave packet centred at a particular time. The expansion sums over all times at which the wavelet is centred as well as over all scales or frequencies for each wavelet. The resulting expansion provides information as to what frequencies are present in the signal at different times, and is particularly appropriate for analysing the multiscale phenomenon.

Because wavelet decomposition analyses a signal in terms of two independent variables, scale (frequency) and time, it is more complicated and requires more computation than Fourier analysis. However, it provides more information about the signal and in its discrete form it is quite fast to compute. In addition, the DWT uses concepts of digital filtering and the uncertainty principle to provide a high degree of data compression, lossless signal reconstruction and independent components that can be eliminated if lower than maximum resolution is required. Indeed, the multiresolution analysis form of the DWT is the basis of the JPEG-2000 storage scheme for digital photography, and finds use in modern formulations of quantum mechanics and optics.

References

[1] Addison P S 2002 *The Illustrated Wavelet Transform Handbook* (Bristol: Institute of Physics Publishing)
[2] Arfken G B and Weber H J 2001 *Mathematical Methods for Physicists* (San Diego, CA: Academic)
[3] Daubechies I 1995 *Proc. Int. Congress of Mathematicians* (1994) vols 1 and 2 (Basel: Birkhäuser) p 56
[4] Landau R H, Paez M J and Bordeianu C C 2005 *A Survey of Computational Physics. Introductory Computational Science* (Princeton, NJ: Princeton University Press) http://press.princeton.edu/titles/8704.html

[4] http://www.physics.oregonstate.edu/rubin/Books/Survey/WaveletCodes.