Chapter 14
Wavelet Transform

Abstract The aim of this chapter is to present a time-resolved frequency analysis of a signal. This is more demanding than one might be inclined to think, due to the time-bandwidth product (a classical analogue of Heisenberg's uncertainty relationship). We have chosen a so-called continuous wavelet transform with Morlet wavelets, since it offers an extremely useful rationale for optimization. Morlet wavelets are presented and a brute force method of analysis is outlined, followed by a more elegant transform utilizing FFT repeatedly in a kind of frequency selection procedure. A computer program in Matlab or Python is given. The user must specify frequency range as well as frequency resolution (through a K parameter). We discuss in detail how to optimize frequency resolution vs time resolution in the analysis.

14.1 Time-Resolved Frequency Analysis

Fourier transformation, as described in Chap. 5, is well suited for "stationary" signals (whose character does not change appreciably over time). For signals that change over time, the temporal information is distributed over all frequency components, and it is a hopeless task to determine how the frequency spectrum varies over time. Therefore, the FFT of a signal that varies widely from one time interval to another within the same data string is of very little value indeed.

There are several methods to explore how the frequency spectrum changes over time. In frequency analysers and in analyses of long-lasting signals (very large data sets), a so-called short-time Fourier transformation or piecewise transformation is used (see Fig. 14.1). "Short-time FT" or "short-term FT" are shortened to STFT, and mathematically STFT can be stated as follows:

$$\text{STFT}\{x(t)\}(\tau, \omega) \equiv \mathscr{X}(\tau, \omega) = \int_{-\infty}^{\infty} x(t) w(t-\tau) \mathrm{e}^{-\mathrm{i}\omega t} \mathrm{d}\, t \; .$$

A. I. Vistnes, *Physics of Oscillations and Waves*, Undergraduate Texts in Physics,
https://doi.org/10.1007/978-3-319-72314-3_14

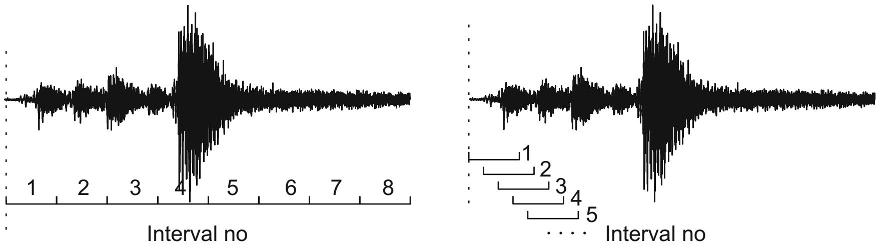

Fig. 14.1 To get temporal information by analysing a long string of time series data, we can break up the total observation into several intervals and perform Fourier transformation interval by interval. The intervals can be chosen so that they do not overlap each other (left part) or overlap each other (right part)

Here, $w(t - \tau)$ is a so-called window function that has a bell-like shape. In practice, the so-called Hanning or Gaussian forms are used, which have a significant value only for a limited period of time around the reference time τ and fall to zero outside the limited time period.

In practice, a fast Fourier transform (FFT) is used for analysing each time window. By choosing a narrow window function w, we get a high time resolution and a poorer time resolution with a wide window function.

In the analysis, we let τ slide through the entire data string to be analysed. We can often choose whether the next window function should overlap the previous one or not, and if so, how large the overlap should be. The result is often shown in a diagram where the intensity of Fourier components in each time window is plotted as a function of time. Intensity is usually indicated by colour coding. The result is usually called a "spectrogram" (see left part of Fig. 14.2).

The advantage of this method is that, if we wish, we can analyse continuous signals for weeks on end. There is no limitation on the length of the data string, since, in practice, we only pick out a limited segment for each round of analysis.

The downside is that we get a frequency resolution that is inversely proportional to the time analysed in each window (i.e. how long the time window lasts). This means that we get the exact same frequency resolution (and thus also time resolution) whether we analyse low-frequency or high-frequency signals. We must select the width of the window function for some typical frequency in the signal. However, if there are widely different frequencies at the same time in the signal, it is impossible to find an optimal window width suitable for all circumstances.

This is an important detail for STFT and for all time-resolved frequency analysers. Due to the time-bandwidth product with which we became acquainted when we studied Fourier transformation, it is impossible to get very precise information about time and frequency at the same time. If we use a window w that extends over a long period of time, we can get fairly accurate frequency information. However, we cannot get precise information about changes over time. It is the classic analogy to Heisenberg's uncertainty relationship that surfaces again.

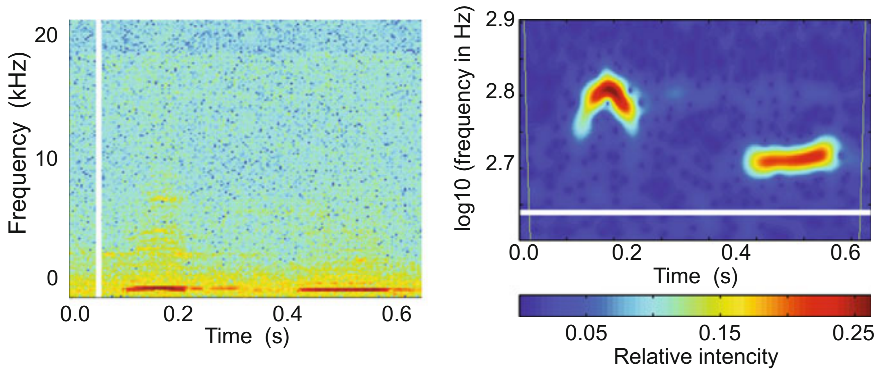

Fig. 14.2 Spectrograms of the sound of a cuckoo bird. To the left is a result of Matlab's built-in STFT function, based on FFT of 300 points at a time, and a so-called Hamming window with 400 points. The overlap from one analysis to the next is 200 points. The vertical white stripe indicates that this spectrum is built up line to line with vertical lines. To the right, a spectrogram is calculated using wavelet transformation with Morlet wavelets, with the computer program given later in this chapter. The horizontal white stripe indicates that such a chart is constructed line by line using horizontal lines. See text for other details

By a "sliding filter" method, as in STFT, where the window changes a data point at a time before new analysis, we avoid jumps in the results arising from randomness in how the intervals are chosen. The problem is, however, that we must carry out some apparently unnecessary calculations. This can be avoided by making a jump in the position of the window function from one analysis to another, but the appropriate jump length will depend on the frequency being analysed. With the STFT function in Matlab, we can choose both the width and the degree of overlap for the window function w, but it is difficult, on account of several reasons, to optimize the choice using STFT.

In this chapter, we will consider another method that can provide time-resolved frequency information, the so-called continuous wavelet analysis with Morlet wavelets. We end up with a chart showing how the frequency picture changes over time, just as we often do with STFT, but there are major differences in how the analysis is performed mathematically [see right part of Fig. 14.2)].

In STFT, we use FFT to analyse each segment of the time signal (picked out using the w window) and then switch to the next segment of the time signal. We therefore build up the "Fourier coefficient versus frequency and time" diagram ("STFT spectrogram") stripe by stripe, using *vertical* stripes. We get all frequencies from zero to half the sampling frequency, whether or not we are interested in the entire range.

In wavelet analysis, we get a "wavelet spectrogram", which may look quite similar to an STFT spectrogram in certain contexts. In wavelet analysis, we also build the spectrogram line by line, but now with *horizontal* lines (see white lines in Fig. 14.2). Therefore, we must choose the entire time interval we want to analyse before the analysis starts, which in some contexts is a disadvantage. The advantage is, however, that we can choose which frequency range will be studied (see Fig. 14.2). We can further select a logarithmic frequency axis if we wish to fully exploit the fact that the *relative* frequency resolution of this method is the same for all frequencies.

> "Continuous wavelet analysis" resembles a sliding, short-time Fourier transform (STFT), but the wavelet analysis with Morlet wavelets gives the same relative frequency resolution for all frequencies. The secret is to use different lengths of time, depending on the frequency to be analysed.

It may be mentioned that there are also many variants of wavelet transformation wherein we make as few transformations as possible with only a fairly limited loss of information. Such a transformation is much more efficient than the continuous variant and is used in technological contexts where speed is of paramount importance. The disadvantage of such a wavelet transformation is that the transformed signal is far more difficult to understand than the usual Fourier spectrum. This is the main reason why we do not go into that method here.

Wavelet analysis is a wide-ranging field of mathematics/informatics, and courses are offered on the subject at many universities. We will not go into details concerning the strictly mathematical or computational aspects of the subject. The purpose of including wavelets in this book is to point out that Fourier transformation is often unsuitable for nonstationary signals and to draw attention at the same time to a method of analysis that is preferable in such circumstances. In addition, work with wavelets can contribute to a deeper understanding of time-limited phenomena in general and the corresponding frequencies. Among other things, there are close analogues between Heisenberg's uncertainty relationship and wavelet analysis.

Some of you will probably use wavelet analysis in the master thesis or in a PhD project (and later employment). For this reason, we place emphasis on showing when wavelet analysis is useful and when the method does not have much to offer. Wavelets are used for analysing solar spot activity (and changes in the spot cycle over time), El Niño Southern Oscillations, glacial cycles, roughness, grain size analysers, analysis of, e.g. cancer cells vs. normal cells and much more.

Technologically, there is an extensive use of wavelets in, among other applications, JPEG compression of image files and in MP3 compression of sound.

14.2 Historical Glimpse

Let us recapitulate the story of Fourier transformation: the French mathematician Joseph Fourier (1768–1830) "discovered" Fourier transformation almost 200 years ago. (Fourier also worked with heat flow, and was probably the first to discover the greenhouse effect.)

Fourier transformation is largely used in analytical mathematics. In addition, the transformation gained enormous currency in the data world after J. W. Cooley and J. W. Tukey discovered in 1965 the so-called fast Fourier transform (FFT) that makes it possible to perform Fourier transformation much faster than before. In FFT, symmetries in the sine and cosine functions are used to reduce the number of multiplications in the calculation, but to get the most effective transformation, the number of data points must be an integer power of 2, i.e. $N = 2^n$.

It has been said that the Cooley–Tukey fast Fourier transform was actually discovered by Carl Friedrich Gauss around 1805, but forgotten and partially reinvented several times before 1965. The success of Cooley and Tukey's rediscovery is due to the emergence of computers at about the same time.

Wavelet analysis is of much later origin. Admittedly, wavelets were introduced already around 1909, but the method was first taken seriously around 1980. There is far greater scope for special variants of wavelet analysis than in Fourier transformation. It is both an advantage and a disadvantage. We can by far tailor-make a wavelet analysis to suit the data we wish to analyse. The downside is that the wide variety of possibilities causes us to use our head a little more in wavelet analysis than in Fourier transformation, both when the transformation is to be carried out and when we interpret the results. But the results are often the more interesting!

14.3 Brief Remark on Mathematical Underpinnings

14.3.1 Refresher on Fourier Transformation

We have gone through Fourier transformation in Chap. 5, but let us recall the mathematical expressions here too.

Let $x(t)$ be an integrable function of time. We can then calculate a new function $X(\omega)$, where ω denotes the frequency, in the following manner:

$$X(\omega) = \frac{1}{2\pi} \int_{-\infty}^{\infty} x(t) e^{-i\omega t} \, dt \; . \tag{14.1}$$

The interesting feature about this function is that we can make a corresponding inverse transformation:

$$x(t) = \int_{-\infty}^{\infty} X(\omega) e^{i\omega t} \, d\omega \tag{14.2}$$

and recover the original function. Note the change in the sign of the exponent in the exponential function.

We see from Eqs. (14.1) and (14.2) that when $x(t)$ is real, $X(\omega)$ will be complex. This is necessary in order that $X(\omega)$ should be able to indicate both how large the oscillations are at different frequencies, *and* the mutual phase of the different frequency components. (A symmetry in X ensures that x, given by the inverse transformation, will be real, as it was originally.)

It should also be mentioned that x and X (called conjugate variables) generally do not have to be time and frequency. In Chap. 8, we also used FT to analyse a spatial description of a wave. The result was a description in "spatial frequencies" which we may equally well have called "wavenumbers". Thus, position and wavenumbers are also conjugate variables. Even more conjugates variables are in use in physics.

The above expressions are used for analytical calculations. When we use a computer, we do not fully know how $x(t)$ varies in time. We only know x_n, the value of x at discrete times t_n, where n is an index that varies from 1 to N, where N is the number of measurements of x that have been made. We assume that the measurement times are equally spaced. The total time over which x is measured is then $T = N\delta t$ where δt is the interval between two successive times of measurement (details discussed in a previous chapter).

When Fourier transformation is performed on discrete data, a discrete transformation is used. This can be stated as follows:

$$X_k = \frac{1}{N}\sum_{n=1}^{N} x_n e^{-i2\pi f_k t_n} = \frac{1}{N}\sum_{n=1}^{N} x_n \exp\big[-i2\pi f_k t_n\big] \qquad (14.3)$$

where we have used the two common ways to write an exponential function, and $k = 0, 1, 2, \ldots, N-1$. Further, $f_k = 0, f_s/N, 2f_S/N, \ldots, f_s(N-1)/N$ where f_s is the sampling frequency. Finally, $t_n = 0, T/N, 2T/N, \ldots, T(N-1)/N$ with $N/T = f_s$.

It may not be easy to grasp the expression, but what we really do to determine the Fourier transform at a frequency f_k is to multiply (term by term) the digitized function x_n with a cosine function of frequency f_k and sum up all the terms that appear. (For the imaginary part of the Fourier transform, we multiply with a sine function of frequency f_k.)

The corresponding "inverse" transformation is given by:

$$x_n = \sum_{k=1}^{N} X_k e^{i2\pi f_k t_n} \qquad (14.4)$$

for $n = 1, 2, 3, \ldots, N$.

14.3.2 Formalism of Wavelet Transformation

Wavelet transformation can be stated in an apparently similar manner to a Fourier transformation:

Let $x(t)$ be an integrable function of time. We can then calculate a new function $\gamma_K(\omega_a, t)$ which provides information about frequency and time simultaneously.

ω_a can be termed "analysis angular frequency". K is a "sharpness" parameter (also known as "wavenumber") related to whether we want high precision in time (K small) or high precision in frequency (K large).

Individual values for the new wavelet transformed function can be found in the following way:

$$\gamma_K(\omega_a, t) = \int_{-\infty}^{\infty} x(t+\tau)\Psi^*_{\omega_a,K}(\tau)\mathrm{d}\tau \ . \qquad (14.5)$$

Here, $\Psi_{\omega_a,K}(\tau)$ is the wavelet itself, and the asterisk denotes complex conjugation. For the Morlet wavelet used here, $\Psi^*_{\omega_a,K}(\tau) = \Psi_{\omega_a,K}(-\tau)$ (see below).

The special thing about wavelet analysis is that we can choose from almost infinitely many different wavelets depending on what we want to get from the analysis. In our context, we only use Morlet wavelets, whose real part can be expressed as an *cosine* function (plus a small amount of constant correction) multiplied by a Gaussian envelope. The imaginary part is a *sine* function multiplied with the same Gaussian envelope as before (see Fig. 14.3).

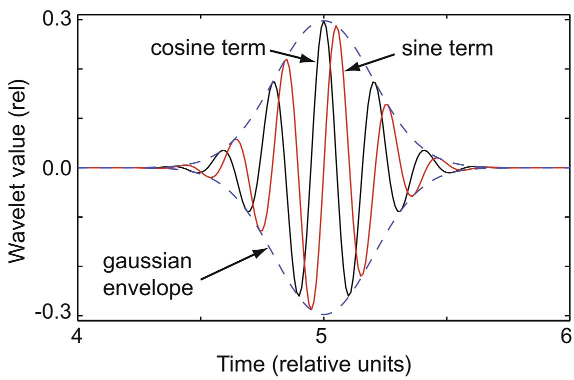

Fig. 14.3 Example of a Morlet wavelet for $K = 6$. Both the real part (the cosine term) and the imaginary part (sine term) are displayed

A Morlet wavelet can be described as:

$$\Psi_{\omega_a,K}(\tau) = C\{\exp(-\mathrm{i}\omega_a\tau) - \exp(-K^2)\}\exp\big[-\omega_a^2\tau^2/(2K)^2\big] \quad (14.6)$$

where C is a "normalization constant". When we describe $\Psi_{\omega_a,K}(\tau)$ numerically, it is advantageous to use the following expression for C:

$$C = \frac{0.798\ \omega_a}{f_s K} \quad (14.7)$$

where f_s is the sampling frequency.

Some remarks:
There is still no uniform description of wavelets. Different sources indicate formalism in different ways; including expressions such as "scaling parameter", "mother" and "daughter wavelets" are key concepts. We have chosen to use a presentation that is close to an article by Najmi and Sadowsky (see the bibliography) because its similarity to other formalism in this book. The "constant" C, I found through trial and error, after imposing the criterion that a wavelet transform of a pure sinusoidal signal should be close to the amplitude of the sinusoid regardless of the parameters ω_a, f_s and K (with a deviation usually not exceeding 1%). The actual expression for a Morlet wavelet we will not be used in practice, except as an illustrative example. For efficient wavelet transformation, we will be using the Fourier transform of the wavelet directly. Details are given in the text which follows.

In the wavelet analysis, we check if the signal we study contains different frequencies at different times. The angular frequency used for the analysis is ω_a. The τ parameter specifies the time at which a specific wavelet has a maximum and corresponds to the centre for the small time interval we investigate.

The parameter K, a real constant, can be called the "width" of the wavelet. Some call it "wavenumber" because it specifies the approximate number of waves under the Gaussian envelope for the wavelet, given by the last factor on the right-hand side of Eq. (14.6). It is recommended that K is 6.0 or larger.

Because of the second factor on the right-hand side of Eq. (14.6), we see that the wavelet Ψ is complex.

Figure 14.3 shows an example of a Morlet wavelet. We see that it bears the correct name, because "wavelet" means a "small wave". Be sure that you thoroughly understand how the wavelet is formed, namely, as the product of a complex harmonic function and a Gaussian envelope centred around time τ.

Note that the expression in Eq. (14.6) is a general description. When it is implemented in a computer program for analysing a specific signal, we must know the sampling frequency used. This enters the normalization constant C. If the specific signal is described in N equidistant points in time, then the total time for sampling equals $T = N/f_s$. We then choose to describe any Morlet wavelet used in the analysis by an array with the same sampling frequency and the same length as the specific signal to be analysed.

If we compare Eq. (14.1) with Eq. (14.5), we see that the expressions look similar to each other. We integrate the product of a function x and a wave. Both are thus linked to an "inner product" within mathematics, but as already said, we would not go into mathematical details here.

However, there are more differences than we might think at first. A significant difference is that the wavelet transformation leads to a three-dimensional description (value of γ as a function of both ω_a and t), while a description based on Fourier transformation is only two dimensional (value of X as function of frequency).

It is possible to make an "inverse" wavelet transformation similarly to an inverse Fourier transform. This is essential when wavelets are used in JPEG image compression and MP3 music file compression. However, we do not include details regarding this formalism in our context. (Those interested may consult the last reference in the list of wavelet resources at the end of the chapter.)

14.3.3 "Discrete Continuous" Wavelet Transformation

First, a few words about the use of the words "discrete" and "continuous". A digitized signal will be called discrete because we only have a finite number of measurement results (equidistant in time). However, we will designate the particular wavelet transformation described in this chapter as "continuous", meaning that the "sliding filter" (just the right part of Fig. 14.1) moves by one point across the digitized signal each time a new calculation is performed. An alternative would be to shift the wavelet by, for example, half the wavelet width.

Wavelet transformation is used almost exclusively on discrete signals, since the calculations are so extensive that they are virtually impossible to perform analytically (except in very simple model descriptions).

For digitized signals (discrete signals), the Morlet wavelet itself can be expressed as:

$$\Psi_{\omega_a,K,t_k}(t_n) = C\{\exp(-\mathrm{i}\omega_a(t_n - t_k)) - \exp(-K^2)\} \exp\big[-\omega_a^2(t_n - t_k)^2/(2K)^2\big] . \tag{14.8}$$

Here, it is assumed that the signal to be analysed is described in equidistant points using the number string x_n for $n = 1, 2, \ldots, N$. The time t_k indicates *the centre of the wavelet (!)*.

The wavelet transformation for one particular frequency and one particular instant will be:

$$\gamma_K(\omega_a, t_k) = \sum_{n=1}^{N} x_n \Psi^*_{\omega_a,K,t_k}(t_n) . \tag{14.9}$$

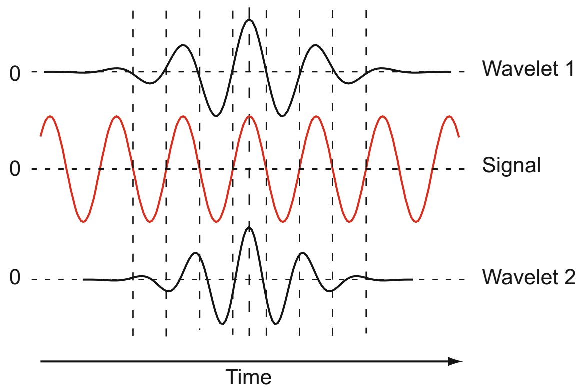

Fig. 14.4 A sinusoidal signal (in the middle) along with a (Morlet) wavelet with the same period of time (*top*) and a wavelet with a shorter period of time (*bottom*)

Let us visualize the process in order to acquire a better understanding of what it entails. In Fig. 14.4, we show a section of a time string along with two different choices of wavelets. Wavelet transformation consists in point-by-point multiplication of the signal with the analysing wavelet, and calculating the sum of all the products. The result is the wavelet transform of the signal for exactly the frequency the wavelet represents and for the exact point in the signal where the wavelet has its maximum value.

For wavelet 1, we see that the signal changes signs approximately at the same instant at which the wavelet changes its sign. That is, the product at any point becomes positive, and the sum of products is therefore quite large (since $\int \cos^2 \omega t \, dt$ is positive). For wavelet 2, the signal and the wavelet do not change sign at the same instant. Some of the products are therefore positive and some negative. The sum of products is significantly lower than for the first case (since $\int \cos \omega_1 t \cos \omega_2 t \, dt$ is often close to zero when $\omega_1 \neq \omega_2$).

We have thus attempted to show that the wavelet transformation of a regular sinusoidal wave will have a maximum when the "periodicity" (or frequency) of the analysing wavelet corresponds to the "periodicity" of the signal in the time interval where we perform the analysis.

To analyse the signal x_n for other periodicities, we need to change the wavelet, and this is done by using, for example, the ω_a parameter.

14.3.4 A Far More Efficient Algorithm

Wavelet transformation defined in Eq. (14.9) is, what is called, a convolution of the time signal $x(t)$ with the wavelet $\Psi^*_{\omega_a,K}$. The appearance of the convolution integral is of interest to us because it is easy to show that the Fourier transform of a convolution is similar to the inner product (pointwise product of two functions) of the Fourier transforms of each of the two functions that are included.

Let us denote the Fourier transform of x and the Fourier transform of the wavelet Ψ with, respectively, $\mathscr{F}(x)$ and $\mathscr{F}(\Psi)$. Let us also denote the Fourier transformed of $x * \Psi$ by $\mathscr{F}(x * \Psi)$. Then, the convolution statement states that

$$\mathscr{F}(x * \Psi) = \mathscr{F}(x)\ \mathscr{F}(\Psi) \tag{14.10}$$

where right-hand side is pointwise multiplication of the two Fourier transforms. But then we can make an inverse Fourier transform $\mathscr{F}^{-1}$ of the right- and left-hand side of this equation and get:

$$\mathscr{F}^{-1}\left(\mathscr{F}(x * \Psi)\right) = x * \Psi = \mathscr{F}^{-1}\left(\mathscr{F}(x)\ \mathscr{F}(\Psi)\right)\ . \tag{14.11}$$

The Fourier transform of the signal, $\mathscr{F}(x)$, can be calculated easily and can be used unchanged for the rest of the wavelet analysis. Fourier transformation of the wavelet Ψ itself must, in principle, be repeated every time we change the analysis frequency or wavenumber. However, we have an analytical expression of the Fourier transform of a Morlet wavelet (see below), which makes the calculation significantly faster.

When we then take an inverse Fourier transform of the product $\mathscr{F}(x)\ \mathscr{F}(\Psi)$, we get the entire $x * \Psi$ envelope in a jiffy. That is, we get the entire time variation in the convoluted signal for the selected analysis frequency (and K value) at one time.

To get a full wavelet analysis, we then have to carry out the procedure for an array of analysing frequencies (which we basically choose for ourselves). For example, if we choose to do the analysis at 1000 frequencies, it means that the calculations take about 1000 times longer than a simple Fourier transform. So, although the method based on the convolution theory is very effective compared to the brute force method, the calculation of continuous wavelet transformation with Morlet Wavelets takes a long computer time.

Now let us look at the Fourier transform of a Morlet wavelet defined in Eq. (14.6). The Morlet wavelet is complex, and it is very satisfying to find that when we calculate the FFT of this complex function, we get a purely real result; moreover, there is no mirroring in the spectrum!

The Fourier transform of a Morlet wavelet (Eq. 14.8) can be stated as follows:

$$\mathscr{F}(\Psi) \equiv \hat{\Psi}_{\omega_a,K}(\omega) = 2\{\exp\left(-[K(\omega-\omega_a)/\omega_a]^2\right) - \exp(-K^2)\exp\left(-[K\omega/\omega_a]^2\right)\}\ . \tag{14.12}$$

We see that this is a bell-shaped (Gaussian) feature (apart from a rather insignificant correction term for most selections of K). The peak of the Gaussian function is at the analysis frequency.

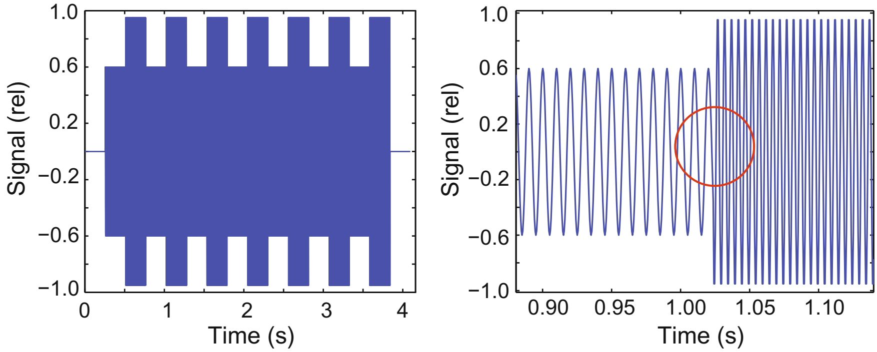

Fig. 14.5 Generated is used in our example. The left part shows the entire signal, while a detail of this is shown in the right part. The amplitude of the 100 Hz signal is 0.6, while the amplitude of the 200 Hz signal is 1.0. The signal is continuous everywhere, even at the instances where the frequency changes abruptly (*detail to the right*)

14.4 Example

We will now show in practice an example of wavelet transformation and start by generating a signal as a function of time.

The signal we generate changes between 100 and 200 Hz at fixed intervals (see Fig. 14.5). The outermost parts of the signal are set equal to zero. Note that when we generate a variable frequency signal during the time the signal exists, we will *insist*, for reasons that need not be spelled out here, that there is no discontinuity in the signal at the instant when the frequency changes. We will be able to meet this demand if we keep an eye on the *phase* of the signal throughout and upgrade the phase at each new time step. This feature of the signal is demonstrated in the expanded plot on the right half of Fig. 14.5.

We then implement the wavelet transform directly from Eqs. (14.9) and (14.8), or we can use the more efficient method described by Eqs. (14.11) and (14.12). The frequency of the analysing wavelet was selected as $\omega_a = 2\pi \times 100$ (which equals 100 Hz in the signal itself).

If we use Eqs. (14.9) and (14.8), we will shift the peak of the wavelet (along the time axis), for each new point in the wavelet transform calculation, from being completely at the outer left edge to being completely at the outer right edge. The result is shown in Fig. 14.6. We see that we get a value of about 0.6 for the times when the original signal had a frequency equal to the analysis frequency.

Figure 14.7 illustrates the more efficient method described by Eqs. (14.11) and (14.12). The Fourier transform of the signal itself is multiplied point by point with the Fourier transform of the wavelet (wavelet with analysing frequency 100 Hz and the given K value). The Fourier transform of the wavelet is a bell-shaped function (almost Gaussian form) with a position of 100 Hz in our case and has no “folded” component. The result of the pointwise multiplication is that only one of the four

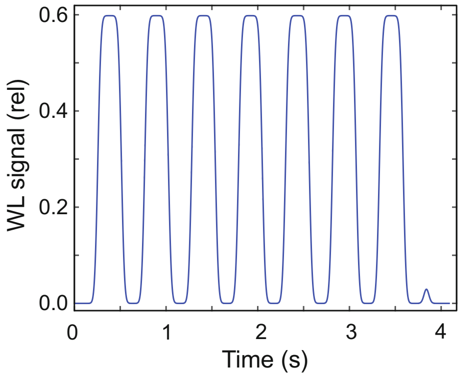

Fig. 14.6 Wavelet transform of the time signal in Fig. 14.5 for an analysis frequency of 100 Hz. The *K* parameter was 12, which means that the wavelet was roughly about $12 \times (1/100)\ \text{s} = 0.12\ \text{s}$ long. The width of the wavelet leads to rounding of sharp corners in the diagram

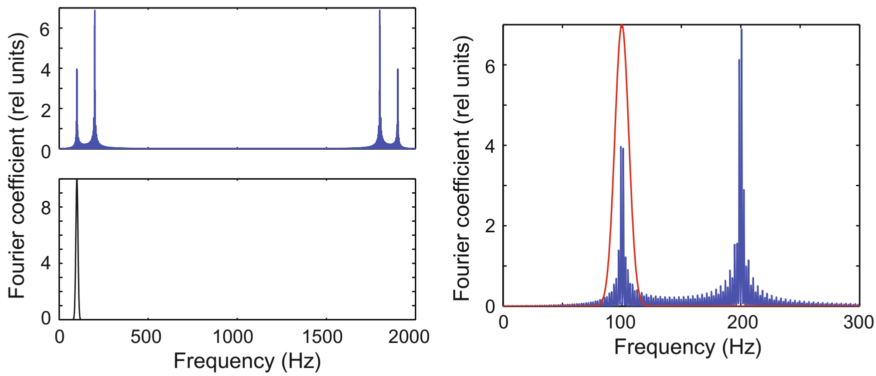

Fig. 14.7 Fourier transform of our time signal at the top left, together with the Fourier transform of the wavelet at the bottom left. The details on the right show that the Gaussian curve in a way functions as a filter and will pick out only parts of the frequency spectrum of the signal. See the text for details

"peaks" in the frequency range of the signal survives. An inverse Fourier transform of this signal is then calculated, and absolute values are taken. Plotting the result, we get exactly the same curve as shown in Fig. 14.6.

The wavelet diagram does not show any sign of the 200 Hz signal, but that is because we have only analysed the 100 Hz signal. To get a more complete wavelet diagram, one has to repeat the procedure for a whole set of frequencies. Then, the wavelet diagram becomes three dimensional with time along the x-axis, frequency along the y-axis and the intensity as a function of time and frequency indicated by colour.

A detail is worth noting here already: The curve in Fig. 14.6 has rounded corners. This is due to the fact that the wavelet has a definite extent in time and therefore will "detect" a 100 Hz sequence even before the wavelet peak is within the 100 Hz range. Similarly, the wavelet will "discover" areas with no 100 Hz, even when the peak of the wavelet is within the 100 Hz range. We come back to this effect in great detail since.

It seems appropriate to list the steps involved in the calculation of a wavelet diagram (using the most effective method):

- Calculate the Fourier transformed of the time signal we are going to analyse.
- Calculate directly the Fourier transformed of a Morlet wavelet with the analysis frequency ω_a and wavenumber K of interest.
- Multiply these with each other, point by point. (Note that there must be consistency between the frequencies f_k in the Fourier transforms of the signal and of the analysing wavelet.)
- Perform an inverse Fourier transform.
- The absolute value of this will then provide information about the time at which the original signal contained frequencies equal to the analysis frequency.
- By changing the Morlet wavelet to the next analysis frequency, we gradually build new horizontal lines in the wavelet diagram until we have covered as many analytical frequencies as we want.

Since fast Fourier transform is such an efficient operation, the method we just outlined is sufficiently fast to be useful.

In a program code a little later in this chapter, there is an example of how wavelet transformation with the effective algorithm can be implemented.

14.5 Important Details

14.5.1 Phase Information and Scaling of Amplitude

In the usual Fourier transformation, we effect in principle two transformations simultaneously, one of the type

$$X(\omega) = \int_{-\infty}^{\infty} x(t) \cos \omega t \, \mathrm{d}\, t \tag{14.13}$$

and the other of the type

$$X(\omega) = -\mathrm{i} \int_{-\infty}^{\infty} x(t) \sin \omega t \, \mathrm{d}t \ . \tag{14.14}$$

The reason is that we have both sine and cosine terms and that we must be able to capture, for example, a sinus signal in x no matter what phase it has.

In the usual Fourier transformation, we have *one* starting point for the analysis. This means that it is easy to find the relative phases of the different frequency components.

In continuous wavelet analysis, we have different starting points and lengths of the analysis window along the way in the calculations. That makes it much more difficult to keep track of phases. This is one of the reasons why we almost exclusively take the absolute value of the wavelet transformation in one or the other variant when the output of wavelet analysis is presented. (However, if we were to do an inverse wavelet transformation afterwards, we would have to take care of the phase information.)

There are several ways of specifying signal strength in a wavelet diagram. Often, the *square* of absolute values is used, which gives the *energy* of the signal.

Based on experience, I do not like to use the square of absolute value because the difference between the powerful and weak parts is often so great that we lose information about the weak parts. Then, it is often better to use absolute value directly ("amplitude level").

However, I often prefer to use *square root* of the absolute value. Then, the weak parts show up even better than when the absolute value is plotted.

We are free to choose how the results of the wavelet transformation are plotted, but we must not lose sight of our choice when we extract quantitative values from the diagrams.

14.5.2 Frequency Resolution Versus Time Resolution

We figured out from Fig. 14.4 that, when the wavelength of the signal x is exactly equal to the wavelength within the wavelet, wavelet transformation will give maximum value. If we make a *small* change in the wavelet by changing the analysis frequency, the transformation will give a lower value, but not zero value. In other words, a wavelet analysis will give rise not only to a different frequency from that corresponding to the signal but also to nearby frequencies.

The theme of this section is to know how far this "adulteration effect" goes.

Let us assume that a wavelet transformation involves a "digital filtering" of a signal, as illustrated in Fig. 14.7. The sharpness of this filtering is determined by the width of the Gaussian function used in the filtering. *We need to find the relation between the width of the frequency picture and the width of the wavelet in the time picture.*

In Fig. 14.8, on the left, there are three different choices of wavelets [(calculated from Eq. (14.8)] and to the right is the Fourier transform of the wavelet [calculated from Eq. (14.12)].

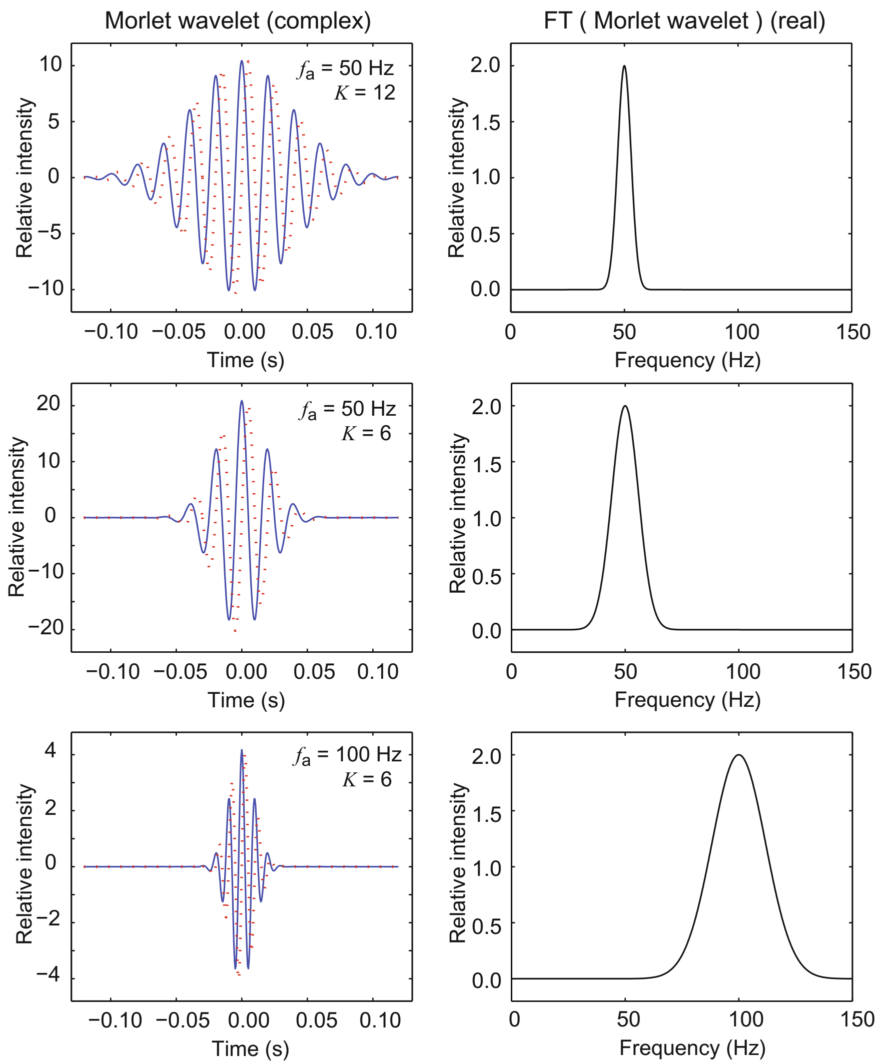

Fig. 14.8 Three different wavelets that indicate how the parameters ω_a and K (analysis frequency and "wavenumber", respectively) affect the wavelet. A wavelet has a limited extent in time (left part). We can specify a width for the envelope curve, e.g. by using the $f1/e$ criterion. If we make an inverse Fourier transform of this wavelet, we get the frequency responses shown to the right. Notice both the position in the frequency spectrum and the width of the Gaussian-shaped curves. There is a relationship between the widths in the time domain and the frequency domain. If we increase one the other will decrease, and vice versa

We know from before that the frequency spectrum of the Fourier transform of the product of a sine signal and a Gaussian curve is itself a curve with a Gaussian envelope. Again, this is get confirmed by Fig. 14.8.

The temporal width of the wavelet can be determined by starting from the envelope curve [from Eq. (14.8)]. If we define the width as the time difference between the peak value and a point where the amplitude of the envelope curve has dropped to $1/e$ of the maximum value, the half-width is:

$$\Delta t_{1/e} = 2K/\omega_a \ .$$

The corresponding width of the Fourier transform of the wavelet is quite close to [(as follows from Eq. (14.12)]

$$\Delta f_{1/e} = f_a/K = \omega_a/(2\pi K) \ . \tag{14.15}$$

It is interesting to note that

$$\Delta t_{1/e}\Delta f_{1/e} = (2K/\omega_a)\ (\omega_a/(2\pi K)) = 1/\pi \ .$$

We can calculate the "standard deviation" for time and frequency by using statistical formulas:

$$\sigma_t^2 = \frac{\int t^2 \Psi^2(t)\mathrm{d}t}{\int \Psi^2(t)\mathrm{d}t}$$

and

$$\sigma_f^2 = \frac{\int f^2 \hat{\Psi}^2(f)\mathrm{d}f}{\int \hat{\Psi}^2(f)\mathrm{d}f} \ ,$$

and it can be shown that

$$\sigma_t^2 \sigma_f^2 = \frac{1}{2\pi} \ . \tag{14.16}$$

This relation is analogous to the Heisenberg uncertainty relation. Examples conforming to this relation are shown in Fig. 14.8.

The relationship is very important for wavelet analysis. If we allow a wavelet to extend for a long time, the width in the frequency domain will be small and vice versa. In other words: *We cannot get accurate temporal details of a signal at the same time as we get an accurate frequency description.*

An interesting consequence of Eq. (14.15) is that

$$\Delta f_{1/e}/f_a = 1/K \ .$$

In other words, in a wavelet analysis, we usually keep K constant throughout the analysis. Then, the relative uncertainty in the frequency values is constant throughout the diagram.

It is then natural to choose a logarithmic frequency axis, in the sense that the analysing frequencies we choose are related to each other as

$$(f_a)_{k+1} = (f_a)_k \; f_{\text{factor}} \; .$$

We have chosen logarithmic axes for the selected analysis frequencies in all examples in this chapter, but it is of course possible to choose the analysis frequencies on a linear scale, at least if the difference between the smallest and the largest analysis frequency is small (e.g. factor two or less).

Comparing wavelets with piecewise FT

If we use piecewise FT, there will within each 'piece' (window) be room for only a few (or less) time periods for a low-frequency signal but many time periods for a high-frequency signal. This means we would get a poor frequency resolution for the lowest frequencies (measured as relative frequency), but a far better frequency resolution for the higher frequencies. That means we would end up with an analysis that would not be optimal.

The procedure used in wavelet analysis provides an optimum time resolution for *all* frequencies. But we *can* nevertheless choose to *somewhat* emphasize time resolution at the expense of frequency resolution and vice versa, depending on what we want to study. This makes the method a very powerful aid in many contexts.

14.5.3 Border Distortion

When we calculate a wavelet transform, we basically multiply a signal, point by point, with a wavelet and sum all the products. We then move the wavelet and go through the same steps. This is repeated over and over again, beginning with the situation in which the centre of the wavelet lies completely at one end of the signal and finishing when the centre of the wavelet is located at the other end of the signal.

We then change the frequency of the analysing wavelet and go through the same routine.

Here, however, we meet a problem. As long as the wavelet is not complete within the data range, we would expect a different result than if the entire wavelet was used in the calculations. This is illustrated in Fig. 14.9. For the position the wavelet has in relation to the data in this figure, only about half of the wavelet will be used in practice. This means that the sum of the products is expected to be much lower (about a half) than what it would be if we had complete overlap.

It is therefore common to mark the wavelet spectrogram with what is called a "cone of influence" in order to indicate the region where the analysis is susceptible to border distortion (the name used for problems caused by incomplete overlap at the edges).

In Fig. 14.10, an example of a wavelet spectrogram which shows an analysis of temperature oscillations in the South Pacific. Figures (numbers) and colours are

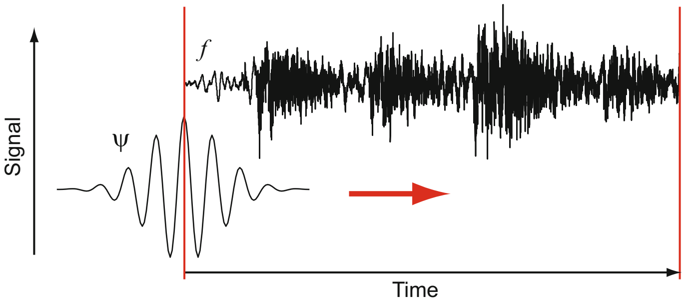

Fig. 14.9 It is not possible to get a correct wavelet result for times and frequencies where the entire wavelet does not come within the data range during the calculations

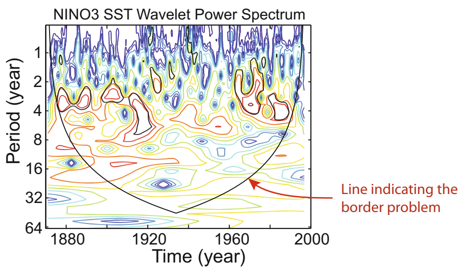

Fig. 14.10 Example of a wavelet spectrogram for temperature oscillations in the South Pacific. The figure was produced with data and software provided by C. Torrence and G. Compo made available at http://atoc.colorado.edu/research/wavelets/ and retrieved April 2016 [1]

used to display "energy" in different forms of oscillation (periodicity) as they have evolved over the last one hundred years.

In this diagram, a V-shaped curve is also drawn, the abscissa of whose cusp is at the middle of the scale and whose arms rise symmetrically on both sides, at first slowly and then steeply near the edges. This V curve is the above-mentioned cone of influence (COI), and it marks the area where most of the wavelets are complete within the data string: everything above the COI represents reliable data, but the results below the curve are suspect.

In the program examples given in the rest of this chapter, we have chosen to place a mark to indicate where the outer part of the wavelet with a value less than $1/e$ of the maximum is outside the diagram. We cover such a small frequency range that in our own examples so that we do not get the entire V curve, but only a small near-vertical part of the total V curve (except for Fig. 14.11). All parts of the wavelet spectrogram that lie between these marks have insignificant errors attributable to border distortion. We provide details below on how to set the selections.

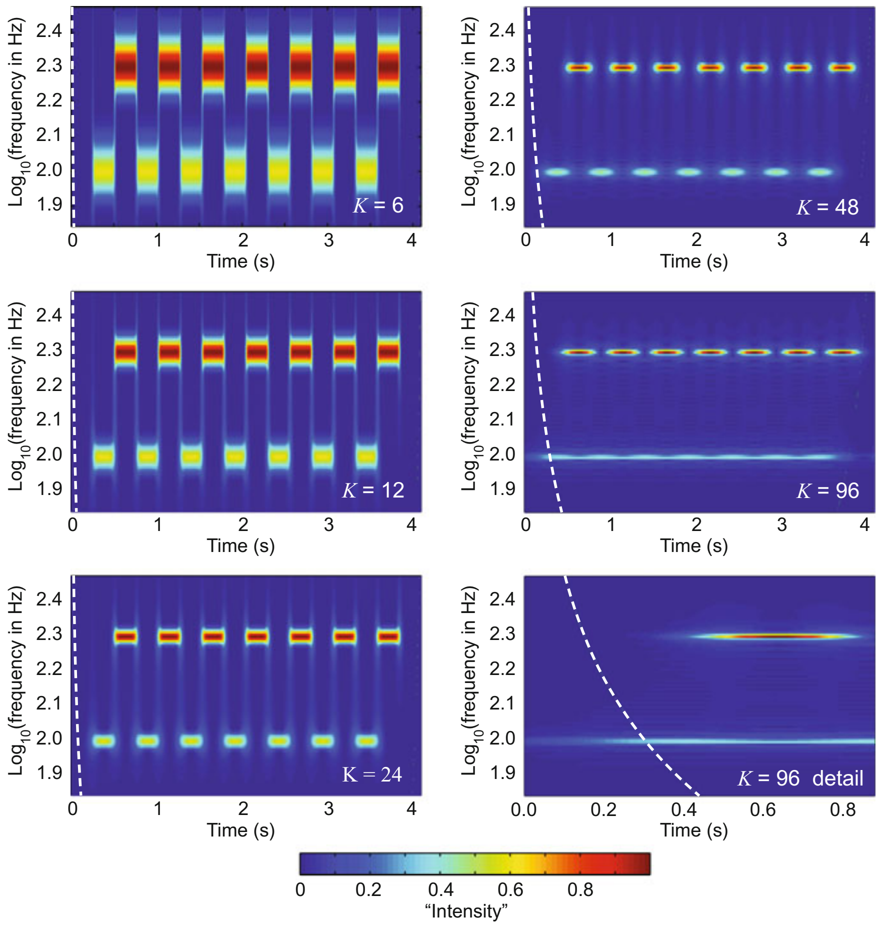

Fig. 14.11 Wavelet diagrams for the time signal in Fig. 14.5 for six different "wavenumbers" K. See the text for details

14.6 Optimization

Wavelet analysis is more demanding than normal Fourier transformation. One must choose what kind of wavelet we are going to use. Even though *we* stick to the Morlet wavelets, we do have to decide what "wavenumber" to use.

We have previously seen that by increasing the "wavenumber" K, the wavelet will have a significant value over a longer period than at low "wavenumber" (at the same analysis frequency). Further, we have seen that when the "width" of the wavelet in the time domain is large (that is, large K value), the "width" on the Fourier transform of the wavelet will be small. The product of the width of the wavelet in the time domain and the width of the wavelet in the frequency domain is constant.

The consequence is that there are no panaceas. If we want to get an accurate indication of the time course, small K values will be preferred. If we want to get as accurate frequency indications as possible, the K value should be large. In principle, we want as good time resolution and frequency resolution as possible, but always have to settle for a compromise.

The optimum result is often achieved if we keep an eye on the signal itself. The signal has often inbuilt uncertainty in time and/or frequency. We can never get a better resolution in time by wavelet analysis than the resolution in the signal itself, and likewise for frequency analysis.

Spelled out even more clearly: when the signal itself in the time domain has passages where oscillations are relatively constant in frequency and amplitude for M oscillation periods, we often get best results in a wavelet transform if the K-value is roughly equal to M (or slightly higher).

Figure 14.11 shows wavelet spectrograms of the signal (described above) that alternated between 100 and 200 Hz. Five different wavenumbers K are used. We see that for low K values the time resolution is very precise, but the frequency determination is poor. For high K values, the opposite holds: the frequency resolution is good, but the time resolution is poor.

In this case, there is really not much more to get in frequency resolution when we go from $K = 48$ to $K = 96$. This is because the signal itself has a "uncertainty in frequency" since the *duration* of each period of "constant frequency" is limited. In this case, there are 25 periods of oscillation within each 100 Hz interval and 50 periods of oscillation within each 200 Hz interval.

In Fig. 14.11, we have enhanced the marking of the left border distortion area. We see that the distortion increases with increasing K value. It may be interesting to note that the border distortion marking changes with the analysis frequency. Furthermore, it is useful to observe that the distance from the side edge to the border distortion mark also indicates smearing in the precision of time in the wavelet diagram. All time information in the analysis is smeared out to an extent that exactly corresponds to the distance from the edge to the border distortion mark.

What are the "best choices" of all the analyses presented in Fig. 14.11? Well, it depends on what we want to get out of the analysis. The diagram for $K = 6$ demonstrates that the change from 100 to 200 Hz (and vice versa) takes place very sharply in time. The $K = 96$ chart shows that the frequency is as uniform as it can be within each of the time intervals. If we need an overall optimization, perhaps $K = 48$ or so would be a good choice.

A standard Fourier transformation of the signal would yield two peaks, one for 100 Hz and one for 200 Hz. Had we taken the absolute value of the frequency spectrum we would not have seen any trace that could show that the signal varied between 100 and 200 Hz in time.

14.6.1 Optimization of Frequency Resolution (Programming Techniques)

Another form of optimization lies in the choice of frequency range for the analysis. In a digital fast Fourier analysis, we automatically get “all” frequencies between zero and the sampling frequency (but only half is useful due to folding). For a continuous wavelet analysis, we usually choose to narrow down the frequency range to the area where the frequency content is of interest.

In Fig. 14.11, we only chose to include frequencies between 70 and 300 Hz in the analysis. The reason is that we knew that the signal contained only frequencies close to 100 and 200 Hz. It may often be an advantage to start with a regular Fourier transform to ensure that we choose a frequency range that is suitable.

However, it is important to consider how many intermediate frequencies we will include in the analysis. In this context, we have to go back to the “width” of the wavelet in the frequency domain. This width, as we have seen before, was:

$$\Delta f = f_a / K \ .$$

This “width” was determined by the Gaussian frequency curve having fallen to $1/e$ of the maximum value. We do not want to take such great steps in frequency from one analysis frequency to the next, but maybe just a fraction of this.

Practical testing shows that an optimal choice of the difference between one analysis frequency and the next is then about

$$f_{a,\text{next}} - f_{a,\text{now}} = f_{a,\text{now}} / 8K \ . \tag{14.17}$$

If we are going to cover a frequency range $[f_{start}, f_{end}]$, then we can easily show that we should use M analysis frequencies in a logarithmic order where

$$f_{\text{end}} = \left(1 + \frac{1}{8K}\right)^{M-1} f_{\text{start}} \ .$$

The number of analysis frequencies is thus:

$$M = 1 + \log(f_{\text{end}}/f_{\text{start}}) / \log\left(1 + \frac{1}{8K}\right) \ . \tag{14.18}$$

14.6.2 Optimization of Time Resolution (Programming Techniques)

A continuous wavelet diagram may sometimes consist of very many points. For example, if we start with sound digitized at a sampling rate of 44.1 kHz, and we study sound with frequencies in the range of 10^2–10^4 Hz, we can in practice pick out only every fourth point in time from the diagram without losing significant information. Once we know that the wavelet has a width of about K times the period of the analysis frequency, we realize that we can remove even more points in time from the diagram without being detected in a wavelet diagram.

It may sometimes be of interest to optimize a wavelet diagram with the time indication. Not least, this is important to get plot files that are so small that they are easily incorporated into reports and the like.

In practice, one finds that it is enough to give each Pth item in the time dimension in a wavelet diagram (without loss of information) when P is given by:

$$P = \text{Integer-Value-Of}\left(\frac{K}{24}\frac{f_s}{f_{a,\max}}\right) . \quad (14.19)$$

In the computer program, the "floor" function is used to get the integer value.

14.7 Examples of Wavelet Transformation

14.7.1 Cuckoo's "coo-coo"

Figure 14.12 shows an example of optimized wavelet analysis. The signal is a CD quality audio file that gives the sound of a cuckoo singing its "coo-coo". The signal is given in three variants, namely as a pure time signal, as a frequency spectrum after a regular FT and finally as wavelet diagram.

In the total work plan, the first step is to select a suitable section from the audio file. This is done by selecting the starting point and total number of points to be retrieved from the available data file. Next, a Fourier analysis is performed. From the Fourier spectrum, we see that the sound usually contains only frequencies between 450 and 750 Hz. Accordingly, the wavelet analysis is limited to this frequency range.

Finally, we must try different K values and choose the "best" compromise between good time description and frequency description at the same time. We need to decide whether we want to prioritize time resolution (by having a small K), but at the expense of a fairly wide frequency response, or accept a slightly poorer resolution (by choosing a larger K) to get a slightly better frequency resolution. What is

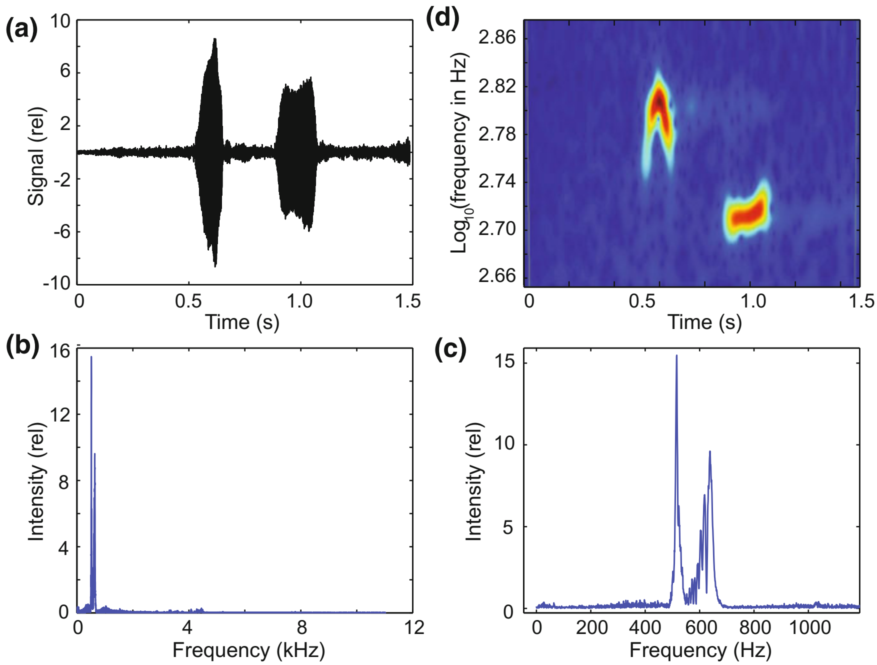

Fig. 14.12 A cuckoo bird's "coo-coo" analysed in the time domain, in the frequency domain (within a window) and in the combination frequency and time in the form of a wavelet diagram

optimal depends on the signal we analyse and will depend on what is important to the individual analysing the data. In our example, $K = 40$ was used.

Note the beautiful details that appear in the wavelet analysis. Have you been aware, for example, that the sound in the first "coo" actually changes significantly within the short period during which the sound lasts? Details in the wavelet analysis of birdsongs allow ornithologists to recognize birds individually. The details are finer than what human auditory apparatus is capable of perceiving.

It should be obvious that, for a sound of such type, wavelet transformation provides far more interesting data than a standard Fourier transform.

14.7.2 Chaffinch's Song

We include two further examples of wavelet analysis. The first (Fig. 14.13) is similar to the one we had for the cuckoo. We have chosen the chirping of a chaffinch, which dominates the birdsong in April. Chaffinch's song is characterized in several different ways. I even like the characteristic "*tit tit tit tit tit …I-love-you*". The joy of wavelet analysis is that the sound image is far more complicated than what we perceive. There is a very fast variation in frequency within each "tit", which we do not perceive. The K value used in the analysis was 48.0.

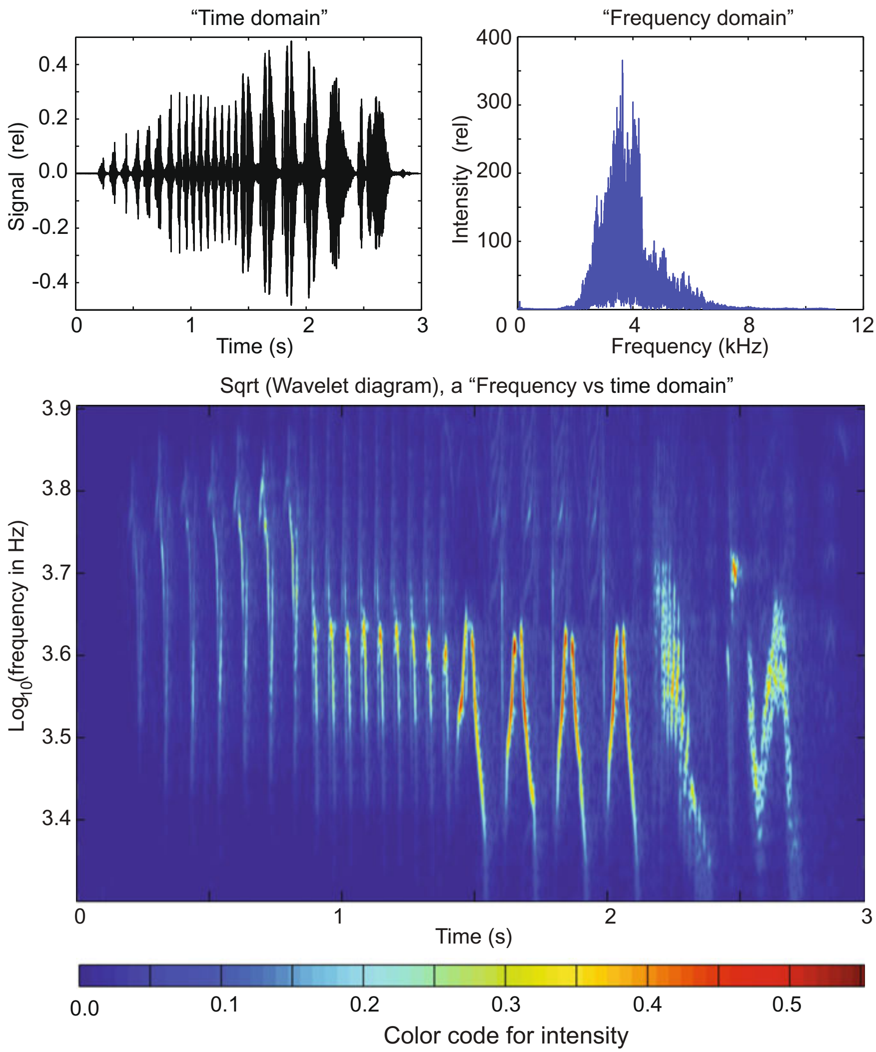

Fig. 14.13 Chaffinch's "tit tit tit tit tit…I-love-you" analysed in the time domain, in frequency domain and after wavelet transformation

14.7.3 Trumpet Sound, Harmonic in Logarithmic Scale

The last example is a wavelet analysis of a trumpet sound (Fig. 14.14). We have chosen a slice in time when the trumpet holds the same tone, and the intensity of the sound we experience is quite constant. The time-domain picture of the sound shows more variation in intensity than we perceive. However, the frequency spectrum (frequency domain) is the way we expected it. It consists of a series of sharp lines that show the fundamental tone and its harmonics.

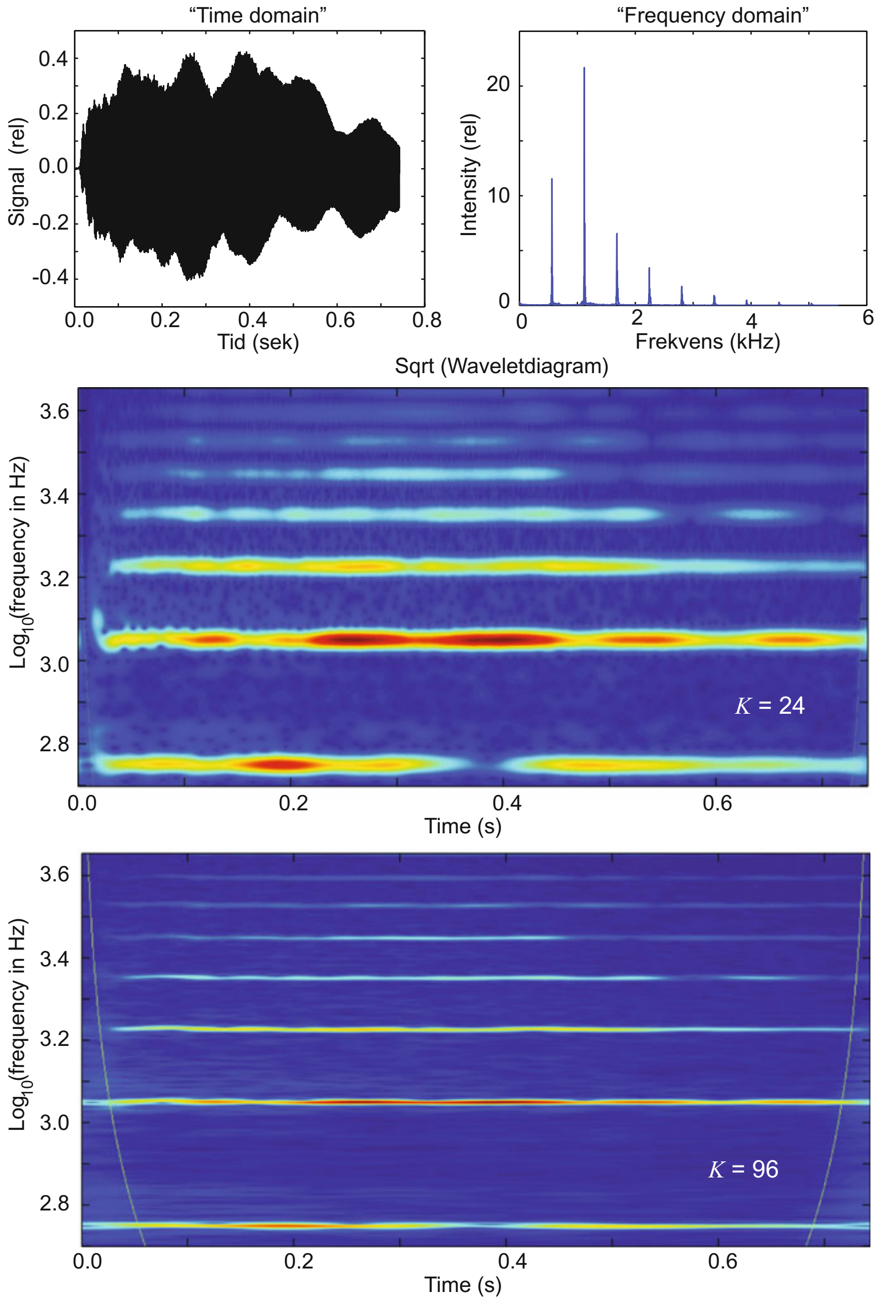

Fig. 14.14 A pure trumpet sound analysed in the time domain, in the frequency domain and in a wavelet transform. Two different K values are used in the wavelet analysis

The frequency axis is linear, and thus, the distance between two adjacent harmonics is constant, more particularly the frequency of the fundamental tone.

A wavelet analysis of such a signal fails to match the sharpness of the frequency spectrum; if we are primarily interested in the frequency of the fundamental tone and the harmonics *for a sustained tone*, the Fourier analysis method is to be chosen.

However, if we are interested in variations in the sound over time, Fourier analysis is not suitable. Then comes the wavelet analysis. We have included two different variants of analysis based on the wavenumbers $K = 24$ and $K = 96$. In the first case, the frequency resolution is rather poor, but the time resolution fits the signal. In the latter case, the frequency resolution is good, but the time resolution is poor.

Are we getting something more out of wavelet analysis than out of Fourier analysis? Yes, as a matter of fact. We see that the strength of the fundamental tone and harmonics varies slightly in time. We also see that there is a certain exchange between the intensities of the fundamental tone and the first harmonic: when one is powerful, the other is weak, and vice versa. This gives life to the sound and shows an example that it is difficult to replace real sound with synthetic sound.

Additionally, note that the distance between the harmonics is not constant in a normal wavelet diagram since we usually have a logarithmic frequency axis.

The frequencies f_n of the harmonics are, we recall, $f_n = nf_1$ where f_1 is the fundamental frequency. We have

$$\log(f_n) = \log(nf_1) = \log(f_1) + \log(n)$$

for $n = 1, 2, 3, \ldots$. It follows that:

$$\begin{aligned}
\log(f_1) &= \log(f_1) \\
\log(f_2) &= \log(f_1) + \log(2) = \log(f_1) + 0.301 \\
\log(f_3) &= \log(f_1) + \log(3) = \log(f_1) + 0.477 = \log(f_2) + 0.176 \\
\log(f_4) &= \log(f_1) + \log(4) = \log(f_1) + 0.602 = \log(f_3) + 0.125 \\
\log(f_5) &= \log(f_1) + \log(5) = \log(f_1) + 0.699 = \log(f_4) + 0.097
\end{aligned}$$

The distance between successive harmonics on a harmonic scale is seen to be:

$$0.301, \; 0.176, \; 0.125, \; 0.097, \; \ldots$$

regardless of the frequency of the fundamental tone (see Fig. 14.14), in sharp contrast with the usual Fourier spectrum.

It is sometimes very handy to use these distances both for recognizing harmonics and for making sure that you have chosen a frequency range for the analysis that includes the fundamental tone (where important).

14.8 Matlab Code for Wavelet Transformation

We have included a Matlab program that can be used to analyse bits of audio files. The code is available at a "Supplementary material" Web page at http://www.physics.uio.no/pow, so you will not need to re-type it.

The code is divided into four different functions: a "main program", a function that reads wav format audio files and plots the time frame of the signal, a function that performs a Fourier transform and plots the result, and a function that performs a wavelet analysis of the signal (given a set of parameters that we choose ourselves). The "main program" uses the three other features in turn.

Some remarks:
The program is self-written and bears imprints which reveal that I learned programming in an earlier era. Feel free to write your own program version more in line with modern trends and requirements. However, if you are short of time, you may want to use our Matlab or Python program directly to get what you need to do, and in the longer term, improve the program on your own.

Main Program

The code is available at the "Supplementary material" Web page for this book at http://www.physics.uio.no/pow.

```
function WLofWAV

% ''Main program'' that reads a wav-file, plots the signal
% in time and frequency domain as well as the wavelet
% trasformed signal. NOTE: All parameters in this program
% has to be chosen carefully according to your signal!
% The program is written by AIV. Version 16. October 2017

% Reads a wav-file
c = 'gjok.wav';  % Name of file
N = 1024*32;     % Number points to be analyzed
nstart = 31000;  % First point in file to be read
[fs,h] = readWavFile(c,nstart,N);

% Fourier transformation
[FTsignal] = fftAndPlot(h,N,fs);

% Wavelet-analysis
fmin = 400.0;  % Use FFT and preferences to choose this!
fmax = 800.0;  % (as above)
K = 32;        % Must be optimized for every signal
[msg] = wltransf(FTsignal,fmin,fmax,K,N,fs); % Wavelet
                                             % transform
```

Reading Wav-File and Plotting the Results

```
function [fs,h] = readWavFile(c,nstart,N)

% This function reads a wav-fil with the name c.
% Reading starts at point number ‘‘nstart’’ and N points are
%  read. The sound is played, and the signal is plotted.
% The function returns the sampling frequency and
% one channel of the stereo signal in the wav file.
% The program is written by AIV. Version 16. October 2017.

nend = nstart+N-1;
[y, fs] = audioread(c, [nstart nend]); % Read array y(N,2)
% from file.
% ’fs’ is usually 44100 (sampling frequency at CD quality)
h = zeros(N,1);  % Picks only one channel from stereo signal
h = y(:,1);
sound(h,fs);     % Play the sound track read from file
T = N/fs;        % Total time for the sound track read

% Plot the signal in time domain
t = linspace(0,T*(N-1)/N,N);
plot(t,h,’-k’);
title(’Wav-file signal’);
xlabel(’Time (sec)’);
ylabel(’Signal (rel units)’);
return;
```

Calculating FFT and Plotting the Results

```
function [FTsignal] = fftAndPlot(h,N,fs)

% This function perform a FFT of a signal h described in
% N points.
% Sampling frequency is fs. The Fourier transformed signal
% (absolute values) is plotted, but the full complex
% Fourier transform is returned to the calling function.
% The function is written by AIV. Version 16. October 2017.

% Calculate FFT of the time descripion of signal: h
FTsignal = fft(h);  % This is what is returned at exit.

% Plot the frequency spectrum (absolute values only)
```

```
f = linspace(0,fs*(N-1)/N, N);
nmax = floor(N/2);  % Plot only lower half (due to aliasing)
figure;
plot(f(1:nmax),abs(FTsignal(1:nmax)));
xlabel('Freqency (Hz)');
ylabel('Relative intensities');
title('Frequency spectrum of the signal');
return;
```

Calculating the Wavelet Transform and Plotting the Results

```
function [msg] = wltransf(FTsignal,fmin,fmax,K,N,fs)

% This function carries out a wavelet transform using Morlet
% wavelets.
% Input is a full FFT of the signal that should be analyzed,
% as well as min and max frequencies for the wavelet analysis.
% K and N are ``wavenumber'' in the Morlet and number of points
% and sampling frequency for the input signal, respectively.
% fs is the sampling frequency.
% The function is optimized so that it chooses both the
% resolution of frequencies and time in the final diagram.
% Two different choises of intensity scaling are possible:
% intScale 1 and 2 correspond to a high or low ``dynamical
% range'' in the wavelet plot (Use 1 to make weak signal
% details visible). At the end the wavelet transformed signal
% is plotted.
% The function is written by AIV. Version 16. October 2017.

% Calculate # frequencies for analysis, write to screen,
% make list of frequencies ready for plot

intScale = 1; % 1 makes weak signal details visible

% Make sure that the FT signal is a column array
SZ = size(FTsignal);
if SZ(1) > SZ(2)
    FTsignal = transpose(FTsignal);
end;

% Calculate/define parameters what will be used later
M = floor(log(fmax/fmin) / log(1+(1/(8*K)))) + 1;
NumberFrequenciesInAnalysis = M
fstep = (fmax/fmin)^(1/(M-1));
f_analysis = fmin;
```

```
T = N/fs;              % Total time for the sound track chosen
t = linspace(0,T*(N-1)/N,N);
f = linspace(0,fs*(N-1)/N, N);

% Allocate array for the wavelet diagram and array for
% storing frequencies
WLdiagram = zeros(M,N);
fused = zeros(1,M);

% Loop over all frequencies that will be used in the analysis
for jj = 1:M
   % Calcualate the FT for the wavelet directly
   factor = (K/f_analysis)*(K/f_analysis);
   FTwl = exp(-factor*(f-f_analysis).*(f-f_analysis));
   FTwl = FTwl - exp(-K*K)*exp(-factor*(f.*f));   % Minor
                                                  % correction term
   FTwl = 2.0*FTwl;    % Factor (different choices possible!)
   % Calculate a full line in the wavelet diagram in one
   % operation! (Inverse of the convolution, see textbook.)
   if intScale == 1
       WLdiagram(jj,:) = sqrt(abs(ifft(FTwl.*FTsignal)));
   else
        WLdiagram(jj,:) = abs(ifft(FTwl.*FTsignal));
   end;
   fused(jj) = f_analysis;  % Store frequencies actually used
   f_analysis = f_analysis*fstep;  % Calculate next frequency
end;
% The main loop finished! The wavelet diagram is complete!

% Reduce file size of the wavelet diagram by removing a lot
% of redundant information in time. The purpose is just
% to make the plotting more managable.
P = floor((K*fs)/(24 * fmax));  % The number 24 may be changed
                                % if wanted
UseOnlyEveryXInTime = P    % Write to screen (monitoring)
NP = floor(N/P);
NumberPointsInTime = NP    % Write to screen (monitoring)
for jj = 1:M
    for ii = 1:NP
        WLdiagram2(jj,ii) = WLdiagram(jj,ii*P);
        tP(ii) = t(ii*P);
    end;
end;

% Make a marking in the plot to visualaize border of
```

```
% distortion
maxvalue = max(WLdiagram2);
mxv = max(maxvalue);
for jj = 1:M
     m = floor(K*fs/(P*pi*fused(jj)));
     WLdiagram2(jj,m) = mxv/2;
     WLdiagram2(jj,NP-m) = mxv/2;
end;

% Plot wavelet diagram
figure;
imagesc(tP,log10(fused),WLdiagram2);
set(gca,'YDir','normal');
xlabel('Time (sec)');
ylabel('Log10(frequency in Hz)');
if intScale == 1
    title('Sqrt(Wavelet Power Spectrum)');
else
    title('Wavelet Power Spectrum');
end;
colorbar('location','southoutside');

msg = 'Done!';
return;
```

It should be noted that once in a while we may want to use an ‘intensity versus time plot’ instead of a full wavelet diagram, to demonstrate particular details (as we did in Fig. 14.6). It is easy to pick a particular horizontal (or vertical) line in the WLdiagram2(i, j) array and make a normal 2D plot of the result. However, care has to be taken to get the right frequency since we use a logarithmic frequency axis in our wavelet diagram.

14.9 Wavelet Resources on the Internet

1. A.-H. Najmi and J. Sadowsky: “The continuous wavelet transform and variable resolution time-frequency analyses.” *Johns Hopkins APL Technical Digest*, vol 18 (1997) 134–140. Available on http://www.jhuapl.edu/techdigest/TD/td1801/najmi.pdf accessed May 2018.
2. http://www.cs.unm.edu/ williams/cs530/arfgtw.pdf “A really friendly guide to wavelets”, C. Valens and others. Accessed May 2018.
3. http://tftb.nongnu.org/, “Time-frequency toolbox”. Accessed May 2018.
4. http://dsp.rice.edu/software/, “Rice Wavelet Toolbox.” Accessed May 2018.
5. http://www.cosy.sbg.ac.at/ uhl/wav.html, Several wavelet links. Accessed May 2018.

6. A 72-page booklet by Liu Chuan-Lin: “A tutorial of the wavelet transform” (dated 23 February 2010) is available on http://disp.ee.ntu.edu.tw/tutorial/WaveletTutorial.pdf Not available May 21th 2018 (Will be available through the web-pages for this book if it is a continuous problem with access.). The booklet also deals with the use of wavelets in image processing.

14.10 Learning Objectives

After working through this chapter, you should be able to:

- Describe similarities and differences between Fourier transformation and wavelet transformation.
- Describe for which signals Fourier transformation is preferred and for which signals wavelet transformation is preferred. Explain why.
- Explain what we can deduce from a given wavelet diagram.
- Explain how we can adjust a wavelet transformation to accentuate temporal details, or details in frequency.
- Explain qualitative analogues between wavelet transformation and Heisenberg’s uncertainty relationship.
- Use a wavelet analysis program and optimize the analysis.

14.11 Exercises

Suggested concepts for student active learning activities: Short-time Fourier transform, time domain, frequency domain, fast Fourier transform, Morlet wavelet, wavenumber K for wavelets, “discrete continuous”, optimization, classical analogy to Heisenberg’s uncertainty relation, frequency resolution, time resolution, absolute value of the transform, cone of influence.

Comprehension/discussion questions

1. What is the most important difference between Fourier transformation and wavelet transformation?
2. In what situations does Fourier transformation provide a rather useless result?
3. What are the disadvantages of wavelet transformation compared to Fourier transformation?
4. When was Fourier transformation implemented on a large scale (FFT) and when did wavelet transformation come into vogue?

5. Wavelet transformation is affected by “border distortion”. What is meant by this? How big is the border zone?
6. Can you outline how wavelet transformation might be used to generate notes directly from a sound recording? What problems do you think may occur?

Problems

7. (a) Calculate a Morlet wavelet (in time domain) for analysis frequencies 250 and 750 Hz when the sampling rate is 5000 Hz and the K parameter is 16. Plot the result with correct time on the x-axis (a figure similar to Figure 14.3).
(b) Calculate the Fourier transform of each of the two wavelets. Use both an FFT directly on the Morlet wavelet described in the time domain and by calculating the Fourier transform directly using Eq. (14.12). Plot the results with correct indications of frequency on the x-axis.
(c) Make sure that the peak occurs at the place you would expect. Do you see mirroring?
(d) Repeat points a–c also when $K = 50$.
8. In this task, the underlying theme is the analogy with Heisenberg’s uncertainty relation.
(a) Generate a numeric data string representing the signal

$$f(t) = c_1 \sin(2\pi f_1 t) + c_2 \cos(2\pi f_2 t) \ .$$

Use 10 kHz sampling rate and $N = 8192$ points, $f_1 = 1000$ Hz, $f_2 = 1600$ Hz, $c_1 = 1.0$, $c_2 = 1.7$. The signal must last throughout the period under consideration. Plot an appropriate section of the signal in the “time domain” (amplitude as a function of time) so that the details will become noticeable. Be sure to provide correct numbers as well as text along the axes, preferably also a heading.
(b) Calculate the Fourier transform of the signal. Plot an appropriate section of the signal in the “ frequency domain” (choose absolute values of Fourier coefficients as a function of frequency), with numbers and text along the axes as above.
(c) Calculate the wavelet transform of the signal (may well be based on the programs given in this chapter, or you can write the program more or less from scratch yourself). Use Morlet wavelets, and let the analysis frequency go, for example, from 800 to 2000 Hz (logarithmically scaled as usual within wavelet transformation). Then, plot the result for the wavenumber K equal to 24 and 200. Comment on the result.

(d) Let the signal be a harmonic signal as before, but now multiplied with a Gaussian function so that we get two "wave packets":

$$f(t) = c_1 \sin(2\pi f_1 t) \exp\left(-[(t - t_1)/\sigma_1]^2\right) + c_2 \cos(2\pi f_2 t) \exp\left(-[(t - t_2)/\sigma_2]^2\right)$$

where $t_1 = 0.15$ s, $t_2 = 0.5$ s, $\sigma_1 = 0.01$ s and $\sigma_2 = 0.10$ s. Calculate the Fourier transform of the signal and also the wavelet transform of the signal. Plot the signal in the time domain, the frequency domain (match the section) and the wavelet transform of the signal for $K = 24$ and 100 (please test more values!), and other parameters as given in point (c) above. Comment on the results!

9. Analyse the song of a blackbird (svarttrost in Norwegian) using wavelet transformation. An audio file is available from the "Supplementary material" Web page. Use the program in this chapter (or own version) and analyse a time string of 1.4 s. Parameters of analysis: Filename: 'Svarttrost2.wav', Nstart = 17000, data string length "64 k" = 2^{16}, frequency range 1500–8000 Hz, wavenumber K equal 12 and 96 (and preferably some values in between as well). The signal consists of five different audio groups. We are primarily interested in the fourth of these!
   Plot the signal in the time domain, in the frequency domain, and the signal analysed by wavelet transformation for this fourth audio group. Be sure to include some sections of the original plots to get details. This applies in particular to the time of the original sound! Hopefully you will then recognize a signal we have encountered at least twice in previous chapters. You should recognize how we can make such a signal mathematically.
   Careful analysis of the fourth bit of the sound signal in the (1) time domain and (2) wavelet diagram makes it possible to see how a close analogy to Heisenberg's uncertainty relationship comes into play. To get the full benefit, you should extract time differences and frequency differences in the charts and compare these with the wavelet evolution in the time and frequency domain for the two selected K values.
   If you are a student and have offer for help from teachers, we strongly recommend that you discuss the relevant details with the teacher until you get a firm grasp of the relationships we wish to bring to the fore. There is a lot of valuable knowledge to extract from this problem, knowledge that can be valuable also in many other parts of physics!
10. Make a wavelet analysis of chaffinch sound (the audio file "bokfink" on the "Supplementary material" Web page). Try a K factor twice as large as in the example in Fig. 14.13. Also, try an analysis for half the K value used in the above figure. Describe the differences you see.
11. In this problem, the theme is to explore various ways to display a wavelet transformed signal so that you get as much information from the diagram as possible. The "dynamic rage" in a colour coded diagram varies with the different choices. Repeat wavelet analysis of chaffinch sound again for $K = 48$. Choose successively "power spectrum" (absolute value after inverse Fourier transformation

squared), wavelet analysis "on amplitude level" (absolute value after inverse Fourier transformation directly) and wavelet analysis with the square root of the wavelet analysis (square root of absolute value after inverse Fourier transformation). Make your assessments of which of these methods you like best for this particular signal. Perhaps you would prefer one of the other viewing modes (squares or square roots) for another signal?

12. Use the knowledge from Chaps. 2 and 3 to calculate the timing of a spring pendulum after it is set in motion by a harmonic force with a frequency equal to the resonance frequency. Follow the oscillations also some time after the harmonic force is removed. Then, perform a wavelet analysis of the oscillation process. Attempt to optimize the analysis with respect to the K value. Do you find an apparent correlation between the Q value for the pendulum oscillation and the K value that gives the optimal wavelet diagram?
13. Use the various wav files available at the "Supplementary material" Web page in order to get used to wavelet transform, how to use it in practice, choice of parameters and analysis of the results. Everyone need practice in order to utilize new tool as well as possible.
14. Select yourself an audio file that you can transform into a .wav file and select a slice that you may want to analyse. Optimize the analysis and tell what information you get from the chart.
16. Find data online that show a time sequence you think might be interesting to study. It may be weather data, solar spots, power consumption or what you need to find. Analyse the data set both in traditional Fourier transformation and wavelet analysis. Which method do you think is best suited to the data you selected? (Should have data with some form of loose periodicity with at least 20–30 periods within the data you have available.)
17. Compare the voices of Maria Callas and Edith Piaf (sound files available at the "Supplementary material" Web page at http://www.physics.uio.no/pow.). Is the vibrato an oscillation in frequency and/or in intensity? Which one of the two artists has the highest number of harmonics? Could you guess this just by listening to their voices?

Reference

1. C. Torrence, G. Compo, http://atoc.colorado.edu/research/wavelets/. Accessed 20 April 2016