MASTER

Wavelets in control engineering

Schneiders, M.G.E.

Award date:
2001

Link to publication

Disclaimer
This document contains a student thesis (bachelor's or master's), as authored by a student at Eindhoven University of Technology. Student theses are made available in the TU/e repository upon obtaining the required degree. The grade received is not published on the document as presented in the repository. The required complexity or quality of research of student theses may vary by program, and the required minimum study period may vary in duration.

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
Wavelets in control engineering
Master’s Thesis

M.G.E. Schneiders
DCT 2001.38
August 2001

TU/e

Coaching:
dr. ir. M.J.G. van de Molengraft
prof. dr. ir. M. Steinbuch

Eindhoven University of Technology
Faculty of Mechanical Engineering
Dynamics and Control Technology
Wavelets in control engineering
Abstract

In this report a for control engineering new analysis and filter technique is presented: the wavelet transform. Basically the wavelet transform can be described as a time-frequency technique. It presents the signal on both a time as frequency axis, comparable with a windowed Fourier transformation that is shifted along the time axis. The basic difference with Fourier techniques is that the window width is changed as function of analyzing frequency. Furthermore there is much freedom in choosing the analysis function, which let the wavelet transform do more than only discover frequency information. Besides as analysis technique the wavelet transform can be used for filtering purposes. With the discrete version of the wavelet transform (DWT) a decomposition in frequency-dependent coefficients is possible, from which the original signal can be reconstructed. The coefficients can be processed in several ways, giving the DWT exotic properties compared with linear filters. To understand this, an introduction in filter banks is presented which forms the basis for the discrete wavelet transform. For both analysis and filter technique general applications are presented.

Next is examined if wavelets can be interesting in control engineering. Since the origins of the DWT lie in the field of signal processing, the available algorithms are efficient but not optimal with respect to the delay times. Therefore a real-time wavelet filter algorithm is derived, which can also be used for analyzing purposes. Expressions for the delay time show that such a filter cannot be used as an online controller. However, for off-line filtering or in supervisory loops wavelets can be interesting. Two application are worked out:

1. Wavelet filters for encoder quantization denoising

   Encoders, widely used in motion systems, always generate noise which is especially annoying when derivatives of the measurement have to be calculated. Often lowpass filtering is applied, but if the amplitudes and frequencies of a signal are spread over a broad range, this approach fails. After an study on quantization noise, the properties of the DWT seems very suitable to minimize quantization effects. Some design rules for a dedicated wavelet filter are proposed and the technique is tested on several artificial and real-life signals.

2. Online feature detection on a CD-player setup

   Sometimes it is desirable to adapt the controller if certain unwanted events disturb the closed-loop. If these events are the result of external disturbances, often the only way to detect them is using a measurement which is part of the closed-loop. Fast detection can then be beneficial to minimize performance loss. Wavelets show good results in isolating features (time-patterns in signals), especially short-living events. Building dedicated waveforms can improve the results and the real-time algorithm makes fast detection possible. The experiences are validated on a servo-loop of a CD-player setup, on which external disturbances are presented in the form of shocks and disc-faults. For some disturbances very early detection is possible: the wavelet filter already isolates features where the measurement still moves within the noise-level.
# Contents

1 Introduction 1

2 Time-frequency representations 3
   2.1 Frequency analysis 3
      2.1.1 Fourier transformations 3
      2.1.2 Fourier transformations in practice 5
   2.2 Time-frequency analysis 6
      2.2.1 Short-time Fourier analysis 6
      2.2.2 Wavelet analysis 9
      2.2.3 Other time-frequency decompositions 12
   2.3 General analysis applications 13

3 Filter banks and the discrete wavelet transform 15
   3.1 Filter banks 15
      3.1.1 Downsampling and upsampling 15
      3.1.2 Perfect reconstruction 17
      3.1.3 Filter banks and the STFT 18
   3.2 Tree structured filter banks 19
      3.2.1 Multiresolution tree 20
      3.2.2 Multiresolution filter banks and the wavelet transform 21
   3.3 General applications of the DWT 24

4 Real-time DWT implementation 27
   4.1 Choosing a structure 27
   4.2 The analysis bank 28
   4.3 The synthesis bank 30
   4.4 Assembling the total system 33

5 Applications in control engineering 37
   5.1 Wavelet filter properties 37
      5.1.1 DWT example: denoising 38
   5.2 Wavelets in control loops 41
   5.3 Application I: encoder-quantization denoising 42
      5.3.1 Encoders and quantization 42
      5.3.2 Quantization errors 43
      5.3.3 Filtering quantization effects with the DWT 45
   5.4 Application II: real-time feature detection on a CD-player setup 48
      5.4.1 Feature detection 48
      5.4.2 Real-time feature detection 49
      5.4.3 CD-player setup 49
      5.4.4 Characterizing disturbances 50
      5.4.5 Building dedicated waveforms 52

6 Conclusions & Recommendations 57

References 58
## A Theoretical backgrounds

<table>
<thead>
<tr>
<th>Subsection</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.1 Uncertainty principle</td>
<td>61</td>
</tr>
<tr>
<td>A.2 Orthonormal bases</td>
<td>62</td>
</tr>
<tr>
<td>A.3 System representations</td>
<td>63</td>
</tr>
<tr>
<td>A.4 Design rules for QMF banks</td>
<td>64</td>
</tr>
<tr>
<td>A.5 Quantisers</td>
<td>65</td>
</tr>
</tbody>
</table>

## B Some wavelet families

<table>
<thead>
<tr>
<th>Subsection</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>B.1 Continuous WT waveforms</td>
<td>67</td>
</tr>
<tr>
<td>B.1.1 Morlet</td>
<td>67</td>
</tr>
<tr>
<td>B.1.2 Meyer</td>
<td>68</td>
</tr>
<tr>
<td>B.1.3 Gaussian</td>
<td>68</td>
</tr>
<tr>
<td>B.2 DWT waveforms</td>
<td>69</td>
</tr>
<tr>
<td>B.2.1 Daubechies</td>
<td>69</td>
</tr>
<tr>
<td>B.2.2 Coiflets</td>
<td>70</td>
</tr>
<tr>
<td>B.2.3 Symlets</td>
<td>71</td>
</tr>
<tr>
<td>B.2.4 Biorthogonal</td>
<td>71</td>
</tr>
</tbody>
</table>

## C MATLAB functions

<table>
<thead>
<tr>
<th>Subsection</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>C.1 Real-time wavelet filters: rtwavfilt.m</td>
<td>75</td>
</tr>
<tr>
<td>C.2 Real-time wavelet analysis for DWT-wavelets: rtwavana.m</td>
<td>78</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

From an historical point of view, wavelet analysis is a new method, though its mathematical backgrounds date back to the work of Joseph Fourier in the nineteenth century. Fourier laid the foundations of frequency analysis, which proved to be enormously important and influential. The first recorded mention of what we now call a wavelet seems to be in 1909, in a thesis by Alfred Haar. The concept of wavelets in its present theoretical form was first proposed by Jean Morlet. The methods of wavelet analysis have been developed mainly by Y. Meyer and his colleagues, who have ensured the methods dissemination. The main algorithm dates back to the work of Stephane Mallat in 1988. Since then, research on wavelets has become international. With the work of Ingrid Daubechies the mathematical backgrounds of wavelets are as good as covered. It is now time to explore the use of wavelets and find new applications. With this report a chance is taken to find new applications in control engineering.

To many readers of this report wavelet is probably a new term. Some claimed that wavelet is the analyzing technique of the future. Other say it’s just a hype from the nineties and will disappear to the background. This report adds to this discussion in seeing what wavelet can mean for control engineering. The main objective of the report is to present wavelet analysis for the purpose of control engineering. Although wavelets are not new in control, the applications in this report possibly are. In the past, wavelet functions are used for modeling and identification of nonlinear behaviour (i.e. see [BB00] or [GR01]). Recently, neural networks have been equipped with wavelets and show excellent results (see [MMPW97],[BS97]). However, the basic techniques in these application are not new. The only difference is that wavelet functions are used instead of polynomials, splines or other dedicated functions.

This report focuses on a different use of wavelets in control. Wavelets are studied as a signal processing technique, with good analysis and filter properties. The first chapters give a short introduction in wavelet theory and time-frequency techniques in general. To understand wavelets, some basic knowledge of another technique is needed, i.e. filter banks. Because all these techniques are born in the field of signal processing, all information in literature is presented in a certain style. In this report it is tried to use the notations and analogies from the field of control engineering as much as possible.

Besides an introduction in wavelets for control people, this report presents some interesting applications. The most interesting and renewable in this report is that all applications can be performed in real time thanks to a real-time wavelet filter algorithm presented in Chapter 4. For some applications other techniques are also an option; wavelets are not the only answer to some problems. With the discussion of the new applications it is tried to emphasize the benefits of wavelets as much as possible.
Chapter 2

Time-frequency representations

Most signals are represented as sequences of numbers along a time axis. However, certain aspects of a signal are better revealed when the signal is represented in the frequency domain. To get global information on the frequency distribution a Fourier transform can be used. Because Fourier transformations are the basic tool for harmonic analysis, the first section provides an overview of the different transforms and their properties. Fourier techniques provide a good description for stationary and pseudo-stationary signals, but they face some limitations when analyzing highly non-stationary signals. These limitations are overcome using a time-frequency analysis technique. Using such a technique the signal can be represented in a domain which is hybrid between time and frequency. The second section presents the short-time Fourier transform as such local analysis technique. Next the wavelet transformation is introduced as a multiresolution time-frequency technique. The benefits of this transform are made clear, showing that wavelet analysis is more than only a time-frequency technique. In the last section some other time-frequency techniques are presented.

2.1 Frequency analysis

2.1.1 Fourier transformations

The aim of signal analysis is to extract relevant information from a signal by transforming it. Most harmonic analysis tools in control engineering (as not all) decompose the original signal into orthogonal trigonometric basis functions. Such tools are called Fourier methods because of the ideas of Joseph Fourier (1768-1830) and his determination to get them accepted. However the origin of these methods predate Fourier. The best known but not the oldest transformation is the Fourier transform (FT) as presented in (2.1)\(^1\). This orthogonal decomposition and the inverse transformation (2.2) are defined for continuous signals of infinite length.

\[
X_{FT}(f) = \int_{-\infty}^{\infty} x(t) e^{-j2\pi ft} dt \quad (2.1)
\]

The analysis function \(X_{FT}(f)\) gives information about the global frequency distribution in a signal (spectral density).

\[
x(t) = \int_{-\infty}^{\infty} X_{FT}(f) e^{j2\pi ft} df \quad (2.2)
\]

\(^1\)In control engineering all signals \(x(t)\) are expected to be real. But for completeness, all transformations presented in this section will also be suitable for complex signals.
To make this decomposition possible, the integral $\int_{-\infty}^{\infty} |x(t)| \, dt$ must exist (which also means that the signal energy is finite) and $x(t)$ must be piecewise smooth (Dirichlet condition [DK97] p.58). Many signals, especially periodic, ones do not fulfill the first condition so the Fourier transform cannot be applied. Hence for pure periodic signals ($x(t) = x(t + T)$) the Fourier series (FS) ([RM87] p.99) can be used:

$$X_{FS}(f_n) = \frac{1}{T} \int_{0}^{T} x(t) e^{-j2\pi f_n t} \, dt; \quad f_n = 0, \pm \frac{1}{T}, \pm \frac{2}{T}, \ldots$$  \hspace{1cm} (2.3)

This transformation shows that pure periodic signals have a discrete frequency spectrum ($f = \frac{n}{T}$). This is trivial because only multiples of a ground frequency $\frac{1}{T}$ fit exactly in a time period $T$. Reconstruction is also possible:

$$x(t) = \sum_{f_n=-\infty}^{\infty} X_{FS}(f_n) e^{j2\pi f_n t}; \quad f_n = 0, \pm \frac{1}{T}, \pm \frac{2}{T}, \ldots$$  \hspace{1cm} (2.4)

If a signal is discrete in time (sampled with sample time $\Delta T$) and has finite energy the discrete-time Fourier transform (DTFT) and inverse can be calculated:

$$X_{DTFT}(f) = \sum_{k=-\infty}^{\infty} x(k) e^{-j2\pi f k \Delta T}$$  \hspace{1cm} (2.5)

Due to the sampling of the signal, the frequency spectrum becomes periodic. Therefore the frequency-band can be decreased to the interval between minus and plus the half sample frequency: $[-\frac{1}{2\Delta T}, \frac{1}{2\Delta T}]$.

$$x(k) = \int_{-\frac{1}{2\Delta T}}^{\frac{1}{2\Delta T}} X_{DTFT}(f) e^{j2\pi f k \Delta T} \, df$$  \hspace{1cm} (2.6)

From this transform we see that an infinitely long sampled signal possesses a continuous frequency spectrum which is periodic. As in the continuous-time case for periodic signals, that have infinite energy, there exists an appropriate transformation known as the discrete Fourier transform (DFT) for periodic discrete-time signals:

$$X_{DFT}(f_n) = \frac{1}{N} \sum_{k=0}^{N-1} x(k) e^{-j2\pi f_n k \Delta T}; \quad f_n = 0, \frac{1}{T}, \frac{2}{T}, \ldots, \frac{N-1}{T}$$  \hspace{1cm} (2.7)

This transform holds for discrete-time signal $x(k)$ sampled at $\Delta T$ and (smallest) period time $T$. Knowing this we can define $N = \frac{T}{\Delta T}$ as the number of samples. Again we see that the spectrum is discretized due to the periodic behaviour of the signal. The spectrum is also periodic, so the frequencies that can be analyzed are finite. The DFT is the only Fourier transform which can be finitely parameterized. The inverse formulation is given by:

$$x(k) = \frac{1}{\Delta T} \sum_{f_n=0}^{N-1} X_{DFT}(f_n) e^{j2\pi f_n k \Delta T}$$  \hspace{1cm} (2.8)

Note that all Fourier transformations present the signal’s frequency content as a complex value. In general only the frequency spectrum is used, which can be calculated by taking the magnitude of the complex value. Phase information is only interesting when examining periodic signals and can be obtained taking the angle of the transform.
2.1 Frequency analysis

2.1.2 Fourier transformations in practice

Now it seems easy to transform any signal using one of the four transforms. In control engineering most measurements are sampled discrete-time signals. Although they are sampled and known in a limited time-interval, they are not periodic in general. So neither the DTFT nor the DFT seems an appropriate transform.

<table>
<thead>
<tr>
<th>Aperiodic in time / Finite energy (continuous in frequency)</th>
<th>Continuous in time</th>
<th>Discrete in time (periodic in frequency)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Periodic in time (discrete in frequency)</td>
<td>FT</td>
<td>DTFT</td>
</tr>
<tr>
<td></td>
<td>FS</td>
<td>DFT</td>
</tr>
</tbody>
</table>

Calculation of the DFT is always possible in these cases but some attention is needed to avoid bad results. Using only the interval $0 \leq t \leq T$ of an arbitrary signal will create signal leakage in general ([dK97] p.66). Periodically extending such signal creates discontinuities which will distort the DFT since this transforms assumes the signal to be periodic. Some of the energy of the fundamental frequency will leak out to neighboring frequencies underlining the name of the phenomenon. Signal leakage does not appear if the signal is:

- periodic and $T$ is a multiple of the basic period of the signal.
- transient (i.e. an impulse response) and is (practically) damped out at $t = T$.

To avoid the influence of signal leakage on the frequency spectrum the signal can be weighted with a dedicated window function: $x_w(t) = x(t)w(t)$. Windowing a signal means making it more periodic to the window length $T$. The disadvantage of using a window is the contribution of the window in the frequency spectrum. Just as signal $x(t)$ the window $w(t)$ has a frequency spectrum: $W(f)$. The frequency spectrum of the windowed signal becomes $X_w(f) = W(f) \otimes X(f)$, since multiplication in the time domain means convolution in the frequency domain. The energy of the different frequency components will be spread out more or less and as a consequence dummy frequencies will appear. In general the disturbance in the spectrum using a dedicated window is many times smaller than without it (thus allowing signal leakage). The use of several windows over the signal’s time-span will be beneficial when analyzing highly non-stationary signals. Averaging the spectral densities of these parts gives a more detailed and realistic view of the frequency distribution of a non-stationary signal. Tuning the window width and applying some overlap between windows will benefit the results.

The DFT has some interesting properties. Analyzing a signal of $N$ samples produces only $N$ complex DFT coefficients. For a real signal only the first $0.5N$ contain essential information so there is never redundancy (the number of coefficients equals the number of samples). From the DWT coefficients it is also possible to reconstruct the original signal, periodic or not. Furthermore the transformation is orthonormal, as all Fourier transformations are, which gives the transformation some nice properties. See appendix A.2 for more information on orthonormality.

From (2.7) it is clear that the frequency spectrum is discrete. The frequency resolution is defined as $\Delta f = \frac{1}{T}$, so only depending on the analyzing time $T$. Note that $\Delta f$ is independent of the sample time $\Delta T$. Decreasing the sample time would mean increasing the number of sample points $N$ to retain the same analyzing time $T$ since $N = \frac{T}{\Delta T}$. The relationship between $\Delta f$ and $T$ is known as the bandwidth-time product and for the DFT given in (2.9). This drawback is independent of the weighting function and is due to the DFT technique. Originally, the DFT is made to examine periodic signals which posses a limited number of frequencies.

$$\Delta f \cdot T = 1$$ (2.9)
Another option is to use the DTFT for analyzing the signal’s global frequency content. In spite of the fact that this transform is only defined for infinitely long signals (2.5) it can be used: a signal of finite length can always be seen at as the result of windowing a signal of infinite duration. Again the influence of the window will be visible in the spectrum, but the spectrum can be determined with an arbitrary density. Contrary to the DFT the frequency resolution performing the DTFT is not bounded by the transform itself. However this resolution is not infinitely small but is determined by the window, which will be made clear in the next section.

In spite of the better resolution of the DTFT technique, it is not much applied in real-life data processing. The reason for this is the existence of the fast Fourier transform (FFT). The FFT is an algorithm for calculating DFT in a very efficient way ([dK97], [GGH96]). For on-line spectrum analyzing purposes and off-line computation of huge data-sets this is the only workable method. Even dedicated digital signal processors (DSP’s) are equipped with FFT algorithms in hardware. The only drawback of the FFT is that the number of samples must equal \( N = 2^m \) (with \( m \in \mathbb{N} \)), but this is no problem in general. The loss of resolution using DFT instead of DTFT is not significant in practice and does not compensate against the calculation efficiency of the FFT.

## 2.2 Time-frequency analysis

### 2.2.1 Short-time Fourier analysis

Fourier transformations offer a one-dimensional projection of a frequency spectrum and will only define the notion of global frequency \( f \) in a signal. The analysis works well on stationary and pseudo-stationary
2.2 Time-frequency analysis

signals. In case of abrupt changes in time or bursts of equal frequency with delays in between, we cannot talk about the phase or the amplitude of the signal’s spectral components. Making a Fourier transformation of such signal, these components will be spread out a little over the whole frequency axis. (Fig. 2.2). However, one is often more interested in the momentary or local distribution of the energy as a function of

![Fig. 2.2: Non-stationary (artificial) signal and its Fourier transform](image)

frequency. To get time-dependent frequency information of a signal, a two-dimensional representation of the signal is needed, composed of spectral characteristics depending on time. In many reports and books this representation is compared with a musical score: a way to show which tones (frequencies) at what time have to be played. An adapted Fourier transformation for this purpose is known as the short-time Fourier transform (STFT) which also uses windowing:

\[
X_{STFT}(\tau, f) = \int_{-\infty}^{\infty} x(t)g^*(t - \tau)e^{-j2\pi f t}dt
\] (2.10)

The window function \(g(t)\) separates the signal into more periodic pieces. The parameter \(f\) in (2.10) is similar to the Fourier frequency and \(\tau\) represents the central time on which the local Fourier transform is made. Note that (2.10) is defined for continuous time signal with infinite duration, like (2.1). In practice (2.10) is not workable but can be rewritten to another Fourier transform presented in the previous section. In that way both time and frequency can be discretized.

However, the success of the analysis on a certain signal depends strongly on the choice of the window \(g(t)\). We can feel that a short window will be able to act very locally in time (good time resolution) but won’t be able to discriminate very good between different frequencies. A longer window is less concentrated in time, but can better discriminate between frequencies (see Fig. 2.3). The whole problem is to balance the frequency resolution against time resolution.

**Conclusion:** in signal analysis, we cannot have both good frequency (\(\Delta f\)) and good time (\(\Delta t\)) resolution. Intuitively we can feel that it is not possible to determine the frequencies that exist on a certain moment (\(\Delta t \to 0\)) of the signal, and vice versa. This is referred to as the uncertainty principle or Heisenberg inequality (see appendix A.1), which states that the bandwidth-time product is lower bounded.

\[\text{In case of complex signals the second signal must be complex conjugated, denoted with the asterisk }^*\].
Reconstruction of $x(t)$ is possible ([Vai93], [dBB00]):

$$x(t) = \frac{1}{G} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} X_{STFT}(\tau, f) g(t - \tau) e^{j2\pi f t} d\tau df$$

(2.11)

This represents the signal as a linear combination of shifted (by $\tau$) and modulated (by $e^{j2\pi ft}$) window functions, with $X_{STFT}$ as a weighting function. Note that $G = \int |g(t)|^2 dt$; a constant which implies that $g(t)$ must have finite energy to make reconstruction possible. It can be proven that the linear combination is not unique: there are many ways to reconstruct the signal.

However, in order to reconstruct the signal, we need not to know the entire spectrum; it is sufficient to know its values at discrete points: $t = n\tau_0$, $f = m\tilde{f}_0$ (with $n, m \in \mathbb{N}$). If $\tau_0\tilde{f}_0 = 1$ is fulfilled the points are said to be on a Gabor lattice ([dBB00]). This is the minimum spread of points in the time-frequency plane that guarantee perfect reconstruction of $x(t)$. This bears resemblance to (2.9); indeed in discrete time the Gabor lattice means there is no redundancy. The sampled version of (2.10) looks like:

$$X_{STFTn,m} = \int_{-\infty}^{\infty} x(t) h_{n,m}(t) dt, \quad \text{with} \quad h_{n,m}(t) = g_1^*(t - n\tau_0) e^{-j2\pi m\tilde{f}_0 t}$$

(2.12)

This looks like projecting the signal onto a set of basis functions $h_{n,m}(t)$ (see appendix A.2). In general this set of basis functions is not orthonormal and reconstruction is only possible in special cases; the use of different windows for analysis and synthesis is needed. Only for special combinations of $g_1(t)$ (2.13) and $g_2(t)$ (2.12) (if they form a bi-orthonormal set) reconstruction with a Gabor lattice is possible. Reconstruction means summing up the weighted basis functions (2.13). This technique was discovered in 1946 by Gabor and is called Gabor's signal expansion.

$$x(t) = \sum_{n} \sum_{m} X_{STFTn,m} h'_{n,m}(t), \quad \text{with} \quad h'_{n,m}(t) = g_2(t - n\tau_0) e^{j2\pi m\tilde{f}_0 t}$$

(2.13)

Although reconstruction exists, in practice the procedure is unstable ([Vai93]) and the STFT is basically used as an analysis tool. If the bases functions ($h'_{n,m}(t)$ in (2.12) and $h'_{n,m}(t)$ in (2.13)) may be chosen more freely (without the complex exponential), we obtain more flexibility, i.e. orthonormal bases are possible then. Such transform in not really a STFT and is known as generalized STFT. Discrete versions of such transform are much easier to design and can be regarded as filter banks, which will be described in Chapter 3.
2.2 Time-frequency analysis

2.2.2 Wavelet analysis

In the case of the STFT a trade-off between time and frequency resolution has to be made. Once a window has been chosen for the STFT, the time-frequency resolution is fixed over the entire time-frequency plane since the same window is used at all frequencies. A fixed frequency/time resolution feels not very natural looking at real world signals. To examine (low-frequent) transients or gross features in a signal a long window is preferable. Similarly, detecting (high-frequent) details or small features will only work well if a small window is used. This property can be achieved by changing the window width with the analyzing frequency. The price we pay using this multiresolution approach is a less high frequency resolution, and less time resolution in the lower frequency range. However, looking at the nature of most signals this approach will give better results than a fixed-resolution one. The wavelet transform (WT) is developed as an alternative approach to the short-time Fourier transform to make a multiresolution analysis possible.

\[ X_{WT}(\tau, s) = \frac{1}{\sqrt{|s|}} \int_{-\infty}^{\infty} x(t) \psi^*(t - \tau s) dt \]  \hspace{1cm} (2.14)

The wavelet analysis is done in a similar way as the STFT analysis, in the sense that the coefficients are determined by measuring similarity between the signal and an analyzing function. The transform is also computed separately for different segments of the time-domain signal, resulting in a two-dimensional representation. Looking at (2.14) we see an analyzing function \( \psi(\frac{t}{s}) \). The analyzing shape and window shape are defined in one function. Since this function must be oscillatory in some way to be able to discriminate between different frequencies, it is called wavelet, which means small wave. The wavelet basis function (or mother wavelet) can be chosen in several ways, without the need of using sine-forms as in the Fourier analysis. This freedom makes wavelet analysis not just a harmonic analysis method, as we will see later. The mother wavelet \( \psi(t) \) is contracted and dilated by varying \( s \), which means changing the scale of our analyzing function \( \psi(\frac{t}{s}) \). By varying \( s \) not only the central analysis frequency but also the effective window width is changed (Fig. 2.4). Therefore, in this multiresolution analysis scale \( s \) is used instead of frequency: large scale means taking a global view of the signal, so analyzing lower frequencies. Small scale takes a short detailed look, revealing high frequency information. Every scale corresponds to a central analyzing frequency and is inversely proportional to that frequency. In order to normalize the signals energy for every scale, the wavelet coefficients in (2.14) are divided by \( \sqrt{|s|} \). The analyzing function \( \psi(t) \) is both localized in frequency and time. Therefore, as with the STFT, it can be proven that the bandwidth-time product \( \Delta f \Delta t \) is constant (A.4) and lower bounded by the Heisenberg inequality (A.1).

The STFT uses a fixed window width (Fig. 2.5), so both \( \Delta t \) and \( \Delta f \) are constant. Comparing with the wavelet, we see that the wavelet uses a varying window width \( (\psi(t) \rightarrow \psi(\frac{t}{s})) \) leading to a fixed number of cycles in the analyzing function (Fig. 2.4). Increasing \( s \) (enlarging the window width) will decrease time resolution \( \Delta t \) inherently resulting in an increase of frequency resolution \( \Delta f \). Because \( \Delta f \Delta t \) is constant.

\[ 3 \] We have assumed that the signal \( x(t) \) is real, so only positive dilations \( s > 0 \) have to be taken in account.
this imposes that $\Delta f$ is proportional to $f$, or:

$$\frac{\Delta f}{f} = c$$

(2.15)

where $c$ is a constant. So the relative frequency resolution is constant in a wavelet analysis. This property is no renewal: many systems have relative frequency resolution. The perception of our ears has this property and tonal musical is arranged around a logarithmic scale. Transfer functions are presented on a logarithmic grid and normal (LTI) filters have passbands that are defined in dB/octave. Since octaves place the frequency axis in a dyadic grid these filters also have relative frequency resolution. Using (2.15) is a very natural way of presenting frequency-dependent phenomena. This is also known as the *constant Q* property, Q being the quality factor of the filter, defined as center-frequency divided by the bandwidth.

A possible wavelet basis function could be that of the STFT (2.16). This often used wavelet for analysis purposes is called the *Morlet* wavelet (already used in Fig. 2.4). It is obtained by using a Gaussian (bell-shaped) window:

$$\psi(t) = g(t)e^{-j2\pi f_c t}, \quad g(t) = \sqrt{\pi f_c} e^{\frac{t^2}{4f_c}}$$

(2.16)
The wavelet center frequency $f_c$ and the bandwidth parameter $f_b$ are tuning parameters. Together they determine the number of cycles in the analyzing function. Looking at (2.14), we see that $t$ is replaced by $\frac{t}{a}$. For the Morlet wavelet, scale and frequency are then coupled as:

$$f = \frac{f_c}{a}$$

(2.17)

Comparing Fig. 2.7.a with Fig. 2.3 it is clear that wavelet analysis gives a better view of the frequency content of the signal than one of the separate STFT analysis. The wavelet transform has a good time and poor frequency resolution at high frequencies (small scales), and good frequency and poor time resolution at low frequencies (high scales). In most non-stationary signals high-frequent events take place in a short timespan while low-frequencies build up the signal and exist for a longer time.

Wavelet analysis can be used to detect other properties than only frequency content of the signal (i.e. [Lew95]). The Mexican hat wavelet is a real waveform and isolates local minima and maxima at the different scales (Fig. 2.7.b). All waveforms have their specific properties and the characteristics that can be revealed from the signal are always coupled to a certain scale, so they are more or less frequency related. More waveforms for analysis purposes are presented in appendix B.1.

Just as the STFT the wavelet also has an inverse transformation; it can be proven that reconstruction of the original signal $x(t)$ is possible using:

$$x(t) = \frac{1}{c_\psi^2} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} X_{WT}(\tau, s) \frac{1}{a^2} \psi(\frac{t-\tau}{a}) d\tau ds$$

(2.18)

where $c_\psi$ is the admissibility constant depending on the (mother) wavelet. It must satisfy the following admissibility condition:

$$c_\psi = \sqrt{2\pi} \int_{-\infty}^{\infty} \left| \frac{\Psi(\omega)}{\omega} \right|^2 d\omega < \infty$$

(2.19)

in which $\Psi(\omega)$ is the Fourier transform of the analyzing wavelet. This condition implies that $\Psi(0) = 0$, which means that the wavelets must have zero mean: $\int_{-\infty}^{\infty} \psi(t) dt = 0$. Since all wavelet-function candi-
times can be accompanied by a constant to let the integral be zero this is not a very restrictive requirement. Furthermore, to satisfy (2.19) the wavelet must be at least oscillatory.

It is apparent that neither the wavelet analysis nor the synthesis can be practically computed using the analytical equations with integrals from (2.14) and (2.18). Analogous the STFT both time and scale can be discretized and we get a wavelet series. Discretization can be done in any way without any restrictions as far as only analysis is concerned. The restrictions become important if reconstruction is desired.

Looking at Fig. 2.6.b a uniform sampling rate feels unnatural because it is not consistent with the time-frequency resolution properties of the wavelet transform. Decreasing scale means reducing the window width so it is reasonable to make the time sampling smaller when scale is decreased. As a consequence frequency resolution decreases as (2.15) and a dyadic sample-grid for the scaling seems suitable for such constant Q analysis. The most natural choice to construct a wavelet series looks like:

\[ X_{WT_{s,m}} = \int_{-\infty}^{\infty} x(t) \psi_{n,m}(t) dt, \quad \text{with} \quad \psi_{n,m}(t) = s_0^{-m} \psi(s_0^{-m} t - n \tau_0) \]  

(2.20)

Comparing this with (2.14), we see that \( s = s_0^m \) and \( \tau = ns_0^n \tau_0 \), with \( m, n \in \mathbb{N} \). If \( s_0 \) is close to 1 (and \( \tau_0 \) is small enough), then the grid is over-complete and reconstruction is possible with little restrictions on \( \psi(t) \). Such projection of the signal on the wavelet coefficients \( X_{WT_{s,m}} \) is called a wavelet frame. If there is no redundancy then the basis function is very constrained, like \( h_{n,m}(t) \) and \( h'_{n,m}(t) \) in (2.12) and (2.13) on the minimum Gabor lattice for STFT-reconstruction. Then the wavelet frame becomes a wavelet base and similar to (2.13), reconstruction for wavelet analysis is defined as:

\[ x(t) = \sum_m \sum_n X_{WT_{s,m}} \psi_{n,m}, \quad \text{with} \quad \psi_{n,m} = \psi(s_0^{-m} t - n \tau_0) \]  

(2.21)

In this way the wavelet coefficients form a unique representation of the signal. This is only possible for special choices of \( \psi(t) \). Note that that set of basis functions on which the signal is projected \( \psi_{n,m} \) are the same for analysis and reconstruction: the base can be orthonormal. This is in sharp contrast with the STFT, where it can be proven that it is impossible to have orthonormal bases at the Gabor lattice with functions well localized in time and frequency ([RV91]).

Besides the wavelet series (2.20) there exists a true discrete wavelet transform (DWT) which is different from the continuous WT on first sight. The DWT is considerably easier to implement and analysis and easy reconstruction is possible with significant reduction in the computation time. The existence of orthonormality is easier to understand in discrete-time and can be translated to the continuous-time case (Chapter 3).

### 2.2.3 Other time-frequency decompositions

All the transformations mentioned above belong to a broad class of space-frequency functions known as the Cohen class ([dBBO00]). Another interesting time-frequency technique in this class is the Wigner-Ville distribution (WVD). The WVD is a global transform and is regarded as being the most fundamental of all time-frequency representations. In 1932 Wigner introduced a distribution in the context of quantum mechanics where it is desirable to express the position and momentum of a particle simultaneously. In 1948 Ville introduced the distribution in signal analysis. Later the Wigner distribution was used in optics but it was not until 1980 that the WVD really started to be applied in signal analysis (i.e. in analysis of impulse responses of loudspeakers).

Just as the STFT and WT this technique presents the local distribution of the energy as a function of the frequency. The Wigner distribution function of the signal \( x(t) \) is defined by:

\[ WVD(t,f) = \int x(t+0.5\tau)x(t-0.5\tau)e^{-j2\pi f \tau}d\tau \]  

(2.22)
2.3 General analysis applications

Note that the WVD is a combination of a Fourier transformation of the product $x(t+0.5\tau)x(t-0.5\tau)$ and an autocorrelation calculation. There also exists inverse formulations just like for the FT and STTF this inverse formulations are not workable and are highly redundant.

The main characteristics of this transform is that it does not place any restriction on the simultaneous resolution in time and frequency ([GGH96]). In other words, the WVD on itself is not limited by the uncertainty relationship (A.1). The WVD (2.22) can be seen as the Fourier transform of a signal which is windowed by its own reverse. In this way the spectrum is minimally disturbed, since this window function is always optimal adjusted to the signal (the window function has almost the same spectrum as the signal). Unfortunately, this transform has some disadvantages. The distribution can show negative energy levels and cross terms, which are irrelevant from a physical point of view. A WVD of a stationary signal containing two pure frequencies $f_1$ and $f_2$ will show a cross term of frequency $\frac{1}{2}(f_1 + f_2)$ which oscillates between positive and negative energy values. Another problem lies in the signal multiplication. The sampling frequency must be at least 4 times higher than the maximum frequency in the signal in order to avoid spectrum aliasing. It may be clear that for several reasons the analysis results of the WVD can sometimes be difficult to interpret.

It is often useful to smooth the distribution along both time and frequency axis in order to get rid of (or at least minimize) the drawbacks mentioned above. This smoothing is done by dedicated window functions, the so called smoothing kernel. After all, the WVD in practice must use a window function, so the uncertainty principle is satisfied. All the kernel extensions on the WVD belong also to the broad Cohen class. Unfortunately the amount of calculation involved with WVD-analysis is significantly more than in the case of STFT or WT, where efficient algorithms exist as will be made clear in chapter 3. Although analytically and mathematically very interesting, the WVD has not found its place in (digital) signal processing.

2.3 General analysis applications

In the previous section the ability to reveal pure frequency information from a signal was already showed. Wavelets can be used for identifying pure frequencies. In this section some other general applications are presented that only use the analysis step.

Because of the multiresolution properties wavelet analysis is proving to be a very powerful tool for detecting self-similarity over a wide range of scales. The NASA collects huge time-series of star systems consisting of a wide range of time scale, from 2 ms to almost 10 hours. Using the WT on these data researchers noticed that some phenomena varied in self-similar manner, that is, the statistical character of the phenomena examined at different resolutions remained the same ([Gra95]). Furthermore, wavelet decomposition is very well adapted to the study of the fractal properties of signals and images.

Wavelets can also be use to detect certain features appearing in the signal. For these applications the shape of the wavelet is important. The wavelet function must reflect the type of features present in the time series. The better the waveform resembles with the feature, the more the feature is isolated by the wavelet coefficients. It is possible to detect discontinuities (in higher derivatives), long-term evolution and break-down points (i.e. see [MMOP00]). Especially for short-time events wavelets perform good and give excellent time localization.
Time-frequency representations
From the previous chapter we know that time-frequency analysis is always possible but reconstruction is difficult and very restricting for the basis function. In this chapter a structure is presented to perform analysis and reconstruction in discrete time without redundancy. We will see that orthonormal bases in discrete-time are much easier to design and interpret than their continuous equivalent. The structure that makes this possible is the filter bank. A filter bank can be seen as a discrete and possible orthonormal version of the generalized STFT. In the second section it will be made clear that an expanded filter bank has the same time-frequency resolution properties as the wavelet transform. The connection between this discrete wavelet transform, actually a tree-structured filter bank, and the (continuous) wavelet transform from the previous chapter is also presented. Between analysis part and synthesis part a filter bank (and also the DWT) produces coefficients on several levels. Each application field has his own specific treatment of the coefficients giving the reconstructed signal the desired properties. An overview of general applications of filter banks and the DWT in specific is presented in the last section.

3.1 Filter banks

A filter bank is a collection of discrete filters, with a common input and/or a common output (Fig. 3.1). We can distinguish two subsets of filters: an analysis and a synthesis bank. The analysis bank consist of two or more (see Fig. 3.7) filters, which separate the input signal into frequency bands of equal width. The resulting coefficients from these analysis filters can be downsampled to avoid redundancy. This means regularly throwing away samples without any loss of information of the input signal. The coefficients can be processed in a certain way, depending on the application. After upsampling (putting zeros in places where samples are left out in the downsampling process), reconstruction of the original signal is possible by using the filters in the synthesis bank. We are particularly interested in perfect reconstruction filter banks. In these PR banks the output is identical to the input, which requires special conditions on the filters.

3.1.1 Downsampling and upsampling

Downsampling by a factor two after applying the analysis bank seems logic in one way: since the information is not doubled it is not needed to double the number of samples. But on the other hand downsampling (\( \downarrow 2 \)) is a non-invertible operation; there is clearly the possibility of losing information since half of the data is discarded. The equivalent of this effect in the frequency domain is called aliasing which states
that the result of this loss of information is a mixing up of frequency components. Only if the original signal is bandlimited, there is no loss of information caused by downsampling according to the Shannon or Nyquist (down)sampling theorem. A popular translation of this theorem sounds: a continuous signal is fully determined by the sample point if the sample-rate is at least twice the maximum frequency \( f_N \), which is called the Nyquist frequency.

Fortunately this theorem has a much broader use. In discrete time we know that the spectrum is periodic with the sample frequency \( f_s \). Suppose that \( x(k) \), the discrete-time input of the filter bank is an alias-free bandlimited signal (Fig. 3.2.a\(^1\)). First suppose that the filters do nothing: \( H(z) = L(z) = 1 \). Downsampling by two now means aliasing in general because the spectrum’s periodicity is changed to the new sample-rate \( \frac{1}{2} f_s \) (Fig. 3.2.b). Aliasing does not occur if the signal is bandlimited to half the Nyquist frequency \( \frac{1}{2} f_s \) (Fig. 3.3). Reconstruction of the original (higher) sampled signal is possible then.

As a consequence, if the system is bandlimited between \( \frac{1}{4} f_s < f < \frac{1}{2} f_s \), downsampling also creates no aliasing and the final frequencies are completely the result of the bandlimited signal above the Nyquist frequency (Fig. 3.4). Also then the original signal (with only high frequency content) is fully determined by these discrete values and can be reconstructed (see Fig. 3.5).

\(^1\)For the sake of simplicity only the positive part of the frequency axis is presented. Apart from that, real signals satisfy \( |X(f)| = |X(-f)| \).
3.1 Filter banks

The explanation is not only valid for downsampling by two. The decimation theorem states that downsampling a sampled signal by a factor \( M \) produces a signal whose spectrum can be calculated by partitioning the original spectrum into \( M \) equal bands and then summing those bands. Normally aliasing occurs at this moment. However, if only one of the \( M \) bands is nonzero, the original signal at the higher sampling rate is exactly recoverable.

![Fig. 3.4: Downsampling without aliasing: frequency content of the signal is only present above the new Nyquist frequency](image)

3.1.2 Perfect reconstruction

Now the role of the analysis filters is also clear; they have to ensure that on each level of the filter bank only one such frequency band is nonzero. The filters are all bandpass filters. For reconstruction the downsampled signals are upsampled again. Only placing zeros between the downsampled data points does not reconstruct the band-limited part to the original signal. The reconstruction filters are responsible for placing back the filter coefficients \( c \) to their original frequency region.

Considering the two-channel filter bank (Fig. 3.1) again: the input signal is split up into two frequency bands. This means the two filters must have a highpass and a lowpass character. If lowpass filter \( L(z) \) and highpass filter \( H(z) \) were perfect (brickwall) halfband filters, the downsampling would not imply information loss. But we know that any realizable (non-ideal) filter has a transition band (see Fig. 3.6). The frequency responses of the two filters will overlap. So besides aliasing (after downsampling), there will be some amplitude and phase distortion in each channel of the filter bank. It is possible to cancel all three problems by tuning analysis and reconstruction filters onto each other; it is possible to design perfect reconstruction filter banks. The PR condition leads to a set of design conditions on both the analysis and
Filter banks and the discrete wavelet transform

Fig. 3.6: Sketch of the frequency responses of the filters of a 2 channel filter bank (amplitude only). Note that only asymptotes are plotted (in general $|H| + |L| = 1 \quad \forall \omega$ is not expected)

synthesis filters (i.e. see [SN96] chapter 4 or [Vai93] chapter 5 to 8). Most popular designs use Finite Impulse Response (FIR) filters since implementation and calculation are much easier in comparison with Infinite Impulse Response (IIR) filters.

Calculating the filter coefficients $c$ at all levels can be seen as projecting the signal onto a new (non-redundant) base. If this projection is orthonormal (which is no exception) the filter bank is said to be paraunitary or lossless. The name quadrature mirror filter (QMF) for a special two-channel filter bank was first used in 1976 by Croisier, Estaban and Galand. Since then the name QMF has been used for general PR filter banks. The channels of a filter bank are also called subbands, and filter-bank technique is called subband coding. To create some insight in the design techniques to cancel aliasing and guarantee perfect reconstruction, the basic design rules are presented in Appendix A.4.

3.1.3 Filter banks and the STFT

The whole story of PR and design conditions also holds for multi-channel filter banks (Fig. 3.7). In the analysis bank, the bandpass filters $H_0, H_1, \ldots, H_{M-1}$ decompose the input signal into $M$ equally spaced frequency bands. Using $M$ channels also means that downsampling is performed with a factor $M$: there is never redundancy. That is why these filter bank are also called maximally decimated/downsampled. Filtering can be interpreted in terms of a window sliding past the data. A filter bank produces coefficients in several channels on a fixed time grid all covering a frequency band with fixed width. Surprisingly the time-frequency resolution of a general multi-channel filter bank is like Fig. 2.6.a. This is similar to the computation of the STFT, particularly the generalized version (2.12).

Fig. 3.7: General M-channel filter bank

The role analyzing function $h_{n,m}(t)$ is replaced by the discrete-time filters in each channel and the downsampling. Although the filters do not have sinusoidal impulse responses, they split up the signal in several frequency bands. Now shifting over the time grid (by means of $n \tau_0$ in (2.12)) is performed by the downsampling ($\downarrow M$). For every frequency region (change of $n$) a specific filter $H_{m}(z)$ is used. The kind of
time resolution of a filter bank (like with the DFT) can be expressed as:

\[ \Delta t = \frac{M}{f_s} \]  \hspace{1cm} (3.1)

The total discrete-time spectrum (up to the Nyquist frequency) is divided in \( M \) parts. So the frequency resolution is:

\[ \Delta f = \frac{f_s}{2M} \]  \hspace{1cm} (3.2)

Now the bandwidth-time product for such a filter bank is given by:

\[ \Delta f \Delta t = \frac{1}{2} \]  \hspace{1cm} (3.3)

This value is determined only by the filter bank technique, like with the DFT. It is independent of the chosen filters and higher than the minimum of the uncertainty principle. But filter banks are optimized for analysis and reconstruction with a non-redundant set of coefficients and not for the highest time-frequency resolution. This bandwidth-time product (3.3) is the minimum value for perfect reconstruction.

Comparing this with the Gabor lattice, filter banks seem to have a lower bandwidth-time product than strictly necessary for perfect reconstruction (the Gabor lattice also holds in discrete time [dBB00]). The difference is that the STFT uses complex exponentials in the analyzing function ((2.12)). So, as well as amplitude information, the STFT reveals phase information of a certain frequency band. This extra information halves the frequency resolution of the analysis. This also emphasizes the difference between the STFT and a filter bank. Because of the complex coefficients, the STFT really gives information on the energy of a certain frequency band. On the other hand, the different channels of real-valued filter banks are more a representation of the course of frequency dependent events in the original signal. Illustrative for this is the difference between the complex and real valued wavelet representations in Fig. 2.7. It is also possible to construct complex filter banks (i.e. DFT filter banks [Vai93]), which satisfy \( \Delta f \Delta t = 1 \).

\[ \text{Fig. 3.8: Possible magnitude responses of the filters in a general } M \text{-channel filter bank.} \]

### 3.2 Tree structured filter banks

Consider the analysis part of the QMF in Fig. 3.1. The two output series \( c_p(k) \) and \( c_l(k) \) can be seen as two ordinary discrete-time signals. Now take a look at Fig. 3.9: just as the input signal \( x(k) \) of the QMF the outputs are led into another QMF. If a filter bank has the PR property and the coefficients are not processed, it is doing nothing and can be disregarded. So if a single QMF has the PR property then the total tree structured filter bank also allows perfect reconstruction. As a matter of fact, this structure can be extended to more than two levels.
3.2.1 Multiresolution tree

Suppose we only want to expand the tree to the lowpass side as in Fig. 3.10. The coefficients $c_l(k)$ only represent the lowest half of frequencies in $x(k)$. The downsampling gives $c_l(k)$ now double frequency resolution compared with $x(k)$ since the same sample range is used for only half of the frequencies. Inher-ently the time resolution is halved since $c_l(k)$ represents $x(k)$ with only half the number of sample points. So if the same filters $H(z)$ and $L(z)$ work on the downsampled signal $c_l(k)$, the time grid is doubled. So the window width is increased as frequency decreases. After the second analysis bank downsampling is applied, so the next level coefficients are stored on a doubled time grid. At each extension, the current high band portion corresponds to the difference between the previous lowband portion and the current one: a passband. Schematically, the frequency response for this filter tree looks like Fig. 3.11.

Fig. 3.10: Wavelet filter bank tree (3 levels)

The time-frequency resolution properties of this structure is similar to Fig. 2.6.b. Hence the name of this
3.2 Tree structured filter banks

Tree structure: the **multiresolution tree**. For a special set of filters $H(z), L(z)$ this structure is called the **discrete wavelet transform** (DWT). The four filters of the QMF structure are said to be wavelet filters.

![Fig. 3.11: Frequency Responses for wavelet filter tree (3 levels)](image)

**3.2.2 Multiresolution filter banks and the wavelet transform**

At first sight it is difficult to see similarities between this DWT and the continuous WT. In the WT of (2.14) the scaled mother wavelet is used to measure similarity between wavelet and signal. Since the wavelet is also localized around some frequency it acts as a bandpass filter when it is multiplied with the signal $x(t)$ (2.14). In the discrete time case, the role of the wavelet is played by the highpass filter $H(z)$ and the cascade of downsampled lowpass filters $L(z)$ followed by the highpass filter (Fig. 3.10). The latter structure provides a set of bandpass filters each covering roughly one octave as a result of the 2-downsampling. The highpass filter covers the highest octave up to the Nyquist frequency $\frac{1}{2} f_s$. Due to the downsampling between the successive filters, the effective filter length increases going downwards in the filter bank. Each filter sequence in Fig. 3.10 can be rewritten as one filter with a downsampler behind, as in Fig. 3.12. The time-grid is enlarged for lower frequencies (higher scales) by the increasing downsample rates (comparable with $\tau = n s^\mu \tau_0$ in the wavelet series (2.20)). These filters can be seen as the wavelet function acting at different scales. However, looking at the impulse responses of these filter sequences they are not exact scaled versions of each other, as in the continuous-time case. Actually scaling is not as easily defined in discrete time since it involves interpolation as well as time expansion. If the structure in Fig. 3.10 is extended a larger number of times, the resulting impulse responses of the equivalent filters (as in Fig. 3.12) can start to show a certain steady pattern (Fig. 3.13). If this iteration process converges to a stable waveform, the set $H(z), L(z)$ are wavelet filters. Then the subsequent filters become scaled versions of each other. Such wavelet filters are said to be **regular**. It can be proven that the final pattern or regular limit function becomes continuous. Further comment on regularity can be found in [Dau92], [Dau88] or [Vai93], but is not relevant for this report.

The easiest way to construct the limit functions is calculating the impulse responses from the reconstruction path. Starting with the lower branch of the wavelet decomposition tree (Fig. 3.14.a) which is just a sequence of lowpass filters with upsamplers in between. Both filters $H'(z), L'(z)$ are FIR filters as defined...
in appendix A.4. If after several iterations the impulse response of this sequence converges to a continuous limit, the following difference equation will hold for that final function \( \phi(t) \):

\[
\phi(t) = \sum_{n=0}^{N} I(n) \phi(2t - n)
\]

(3.4)

This can be seen as convoluting the converged function more times will only scale up the function but not change its shape. The final function itself is lowpass and is called the scaling function of the wavelet. The final function for the bandpass sequences is obtained in the same way, except for one highpass filter at the start of the filter sequence (Fig. 3.14.b). This function \( h_c(t) \) is known as the wavelet \( \psi(t) \):

\[
\psi(t) = \sum_{n=0}^{N} h(n) \psi(2t - n)
\]

(3.5)

If the filters \( L(z) \) and \( H(z) \) form an orthonormal filter bank and the sequences from Fig. 3.14 converge then it can be proven ([Dau88]) that the iterated waveform \( \psi(t) \) can be used as orthonormal wavelets in continuous time case (2.20). In the DWT both time and frequency resolution are on a 2-dyadic grid since downsampling is performed by a factor 2 and frequency bands are halved at each extension. So if we use a \( \psi(t) \) that satisfies (3.5) and \( s_0 = 2 \) in (2.20) the wavelet base becomes:

\[
\psi_{m,k}(t) = 2^{m/2} h_c(2^{-m}t - k)
\]

(3.6)
Now the shifted and scaled version of $\psi_{m,k}(t)$ are orthonormal to each other which can be expressed as:

$$\int_{-\infty}^{\infty} \psi_{m_1,k_1}(t) \psi_{m_2,k_2}(t) dt = \delta(m_1 - m_2)\delta(k_1 - k_2)$$

(3.7)

Furthermore, shifted versions of the scaling function $\phi(t - k)$ are orthonormal to each other and to the set of shifted and scaled wavelets $\psi_{m,k}(t)$. Proof of these consequences is shortly presented in [SN96]. Such wavelet bases derived from wavelet filters are called (continuous-time) compactly supported wavelets, discovered by Ingrid Daubechies. Her work made discrete-time wavelet analysis and reconstruction possible. The converse is also true: orthonormal sets of scaling functions and wavelets can be used to generate PR wavelet filter banks ([Mal89] and [Dau88]). Examples of wavelet shapes and their corresponding scaling functions derived from wavelet filters are presented in appendix B.2. Besides frequency characterization wavelets have a certain time-pattern. Note that their often uncomfortable shapes are a consequence of design criterions that ensure their specific properties (i.e. orthonormality and PR). Unfortunately, the success of a certain decomposition depends strongly on the choice of the wavelet in combination with the signal properties. However, the classical STFT decomposition is signal independent.

The continuous WT detecting low frequencies (high scales) forces the wavelet length to high values ($\Delta t \rightarrow \infty$ if $\Delta f \rightarrow 0$ in (A.4)). For finite-time signals this means that the time-span of the scaled wavelet becomes bigger than the signal length, which is corrupting the result and not workable. It implies that is not possible to determine the mean value of a signal using the WT. This is also a reason why reconstruction with the WT (2.21) is not possible with finite coefficients, besides discretization problems. In the DWT the problem for high scalings is solved by the scaling function. For an arbitrary level of decomposition, the lower branch of the tree (Fig. 3.14.a) covers the lower frequencies of the analysis down to $f = 0$. This function is orthonormal to the wavelet base so the DWT is completely orthonormal. Note that there are a number of filter designs for 2-channel PR filter banks that, unlike the wavelet filters, are not regular. Their iterated sequences do not tend to regular limit functions, but diverge. However, they possess the PR property if used in tree structured filter banks and they can also be used for multiresolution analysis. The only difference is that the levels use not one prototype waveform for the analysis. In some application fields it is believed but not proven that wavelet filters with high regularity perform better in multiresolution analysis than ordinary PR filter designs ([RV91]).

Let $p$ be defined as the decomposition level of the multiresolution tree. Note that it plays the role of the scale-factor $s$ in the continuous WT. Just as with normal filter banks the multiresolution decomposition has
Filter banks and the discrete wavelet transform

certain resolution properties, only these are dependent of the decomposition level. For the time resolution

can be written:

$$\Delta t = \frac{2^n}{f_s}$$  \hspace{1cm} (3.8)

This is proportional to the effective downsampling for each level. The frequency resolution equals:

$$\Delta f = \frac{f_s}{2^{n+1}}$$  \hspace{1cm} (3.9)

If we calculate the bandwidth-time product for such filter bank we get:

$$\Delta f \Delta t = \frac{1}{2}$$  \hspace{1cm} (3.10)

This value is the same as with the normal filter bank technique. Again the bandwidth-time product is deter-

mined by the non-redundant technique of subband coding; the minimum value for perfect reconstruction.

### 3.3 General applications of the DWT

Now it is possible to project a discrete-time signal onto another base in the form of the coefficients $c$. In
general this is done to represent particular information in better a way than the original signal is able to. With filter banks and the DWT the base projects the signal in a frequency dependent way. Before reconstructing the original signal we can modify the coefficients which will modify the input signal in a frequency dependent way. This can be seen as filtering. Efficient denoising can be performed with the DWT. The properties of such wavelet filter are exotic in comparison with the normal LTI filter known in

control engineering (more on this in Chapter 5).

Filter banks are used in communication. Sometimes it is efficient to transport a signal over several serial

lines with less bandwidth. A filter bank can divide a signal in a large number of channels. The opposite

is also used: glass fibers are used for serial transport with high bandwidths. Analysis and reconstruction

banks are then used in a reverse order to split the serial code into several parallel on both sides of the

communication channel. These systems are called transmultiplexers.

Wavelets and filter banks are used for compression purposes (i.e. audio). In general the coefficients in the

new base represent the signal more efficient than the raw data-series does. More efficient means that the

energy (the magnitudes of the coefficients) is present in a small number of coefficients whereas the others

are almost zero. Such signal has a high entropy level. If the coefficients are close to zero, their influence in

the original signal can be neglected. In this way it is possible to reconstruct the signal with little distortion

with only a small number of coefficients. This is also possible for 2-dimensional signals for which special

bases exist (also wavelet bases). These are used for image and audio compression.

However, at this moment the most popular compression applications use no DWT transform but ordinary

filter banks. Examples are mpeg (layer 3) and jpeg which use respectively a 32-channel filter bank and a

DFT-based filter bank. The constant window width of these methods can cause problems (i.e. blocking

artifacts in image compression [SN96]). The multiresolution approach of the DWT has much better prop-

erties and is finding its way into a very common and widely used application: in the new jpeg2000 standard

wavelets are used.

Another interesting technique for compression uses the wavelet packet decomposition. Actually, the

wavelet/multiresolution tree is a subset of the tree structured filter banks. Is is also possible to extend the

filter bank with wavelet filters as in Fig. 3.7 to higher levels. Then a fully decomposed wavelet packet tree

is created. In the adapted wavelet packet analysis an algorithm is used to determine the optimal decomposition depth at all possible levels to get the best wavelet packet tree. *Best* means highest entropy level. This generalization of the DWT offers even better compression properties but works signal dependent. Wavelet packets have also special filter properties since it is possible to zoom in on specific frequency-bands.
Filter banks and the discrete wavelet transform
Chapter 4

Real-time DWT implementation

From the previous chapter it is clear that the DWT can be seen as a filter. To see if such wavelet filter can be used in on-line applications a real-time algorithm is needed. Most existing implementations of the DWT are off-line versions based on the structure in Fig. 3.10. They are efficient in calculation effort and memory usage since they use a structure similar to the FFT-algorithm, the so called Mallat algorithm (see [Mal89]). The disadvantage of this algorithm is that the whole signal (or a huge part of it) must be measured before the coefficients can be calculated. It can be implemented on-line (i.e. [NC97]) but such architecture is not optimal with respect to the delay times. Real-time in this case means that for every new input a new output is calculated with minimal delay. For the time being such real-time implementation is not found in literature. A disadvantage of such structure is the increase in computational effort since the coefficients are calculated with redundancy. However, this redundancy is beneficial for the time-resolution of the analysis as we will see. The real-time implementation is programmed for use with MATLAB. It will be clear that there is always a significant amount of delay between the input and the reconstructed output if border distortions want to be avoided.

4.1 Choosing a structure

Digital filters can be implemented in many ways. To make the implementation easy to understand a well-known structure for control engineers is chosen: the discrete state-space representation.

\[
\begin{align*}
\tilde{x}(k+1) &= A \tilde{x}(k) + Bu(k) \\
y(k) &= C \tilde{x}(k) + Du(k)
\end{align*}
\]

(4.1)

However, such setup fails somehow in our requirements: on each sample hit we want to calculate the newest wavelet coefficients, process them and then reconstruct the signal. In the just proposed setup the newly calculated state space \( \tilde{x}(k+1) \) is not used in the output at that time \( y(k) \). So this structure creates an unnecessarily delay of one sample, and lets us no freedom in processing the wavelet coefficients \( \tilde{c} \). A slightly modified structure will serve our goal:

- acquire a new input sample and calculate new wavelet coefficients
- process these coefficients
- reconstruct a new output
As a system we can denote such structure as:

\[
\begin{align*}
\dot{x}(k) &= A \dot{x}(k - 1) + Bu(k) \\
\downarrow \quad \text{Processing} \quad \tilde{c} \\
y(k) &= C \tilde{x}(k)
\end{align*}
\] (4.2)

Both representations show much resemblance and can be converted into each other (appendix A.3). There are several ways to implement such system: the state-vector can be chosen in various ways. However, the state space must include:

- old inputs \( u \): these are needed to compute new wavelet coefficients.
- new and old wavelet coefficients \( \tilde{c} \): after updating our states we want to process the new wavelet coefficients. With the new and old coefficients the input can be reconstructed.

We can store intervening coefficients (like \( c^i \) in Fig. 4.1) and use these to calculate deeper level coefficients (as \( c^{ih} \)). We can also calculate the final coefficients complete out of old and new input coefficients. In the synthesis step we can also choose to store intervening coefficients. Doing this will reduce the number of coefficients in the state space, but will have a higher computation effort on each sample hit. Since memory is not really a problem, but computational effort is in real-time, we choose to store only the final wavelet coefficients on each level. After all, only this setup has minimal delay in the reconstruction process. To guarantee this minimal delay, the structure in Fig. 3.10 is adapted. For minimal delay at each level all coefficients at all levels have to be stored (Fig. 4.1), which will be made clear in the next subsections. Note that Fig. 4.1 and Fig. 3.10 are basically the same. The filters are rewritten into filter matrices which work on coefficient-vectors. The coefficient-vectors can be seen as time-series, so this notation makes it possible to study the timings in the calculation process of the wavelet tree. Doing so, it is possible to find general expressions for the delay times and to build a real-time structure.

![Fig. 4.1: Real-time structure for a 2-level wavelet filter implementation](image)

### 4.2 The analysis bank

First the analysis part is rewritten. As we have seen in Chapter 3, the DWT makes use of a lowpass and a highpass filter (A.17). As a consequence of the design rules, these filters always have odd order \( 2 \) (see Appendix A.4). To calculate a level 1 coefficient (i.e. \( c_b \)) we just apply the analysis filter to the input signal. The used input points can be put in a vector and the filter can also be rewritten as a vector Fig. 4.2.:

\[
c_b(k) = H_1 \begin{bmatrix} u(k) \\ u(k - 1) \\ \vdots \\ u(k - N) \end{bmatrix}, \quad H_1 = \begin{bmatrix} h(0) & h(1) & \cdots & h(N) \end{bmatrix} \] (4.3)
The discrete-time index is used to express that $c_h(k)$ is calculated using $u(k)$ as the newest input sample. We see that we need $N$ old input points. To compute $c_{th}(k)$ out of $\hat{c}_l$ we can write:

$$c_{th}(k) = H_1 \begin{bmatrix} c_l(k) \\ c_l(k-2) \\ \vdots \\ c_l(k-2N) \end{bmatrix}$$ (4.4)

We see that there are $N + 1$ coefficients $\hat{c}_l$, but these are spread out over $2N + 1$ samples due to the downsampling before the second analysis filter. To calculate the required $c_l$ out of the input-vector $\bar{u}$ we can write:

$$\begin{bmatrix} c_l(k) \\ c_l(k-2) \\ \vdots \\ c_l(k-2N) \end{bmatrix} = L_2 \begin{bmatrix} u(k) \\ u(k-1) \\ \vdots \\ u(k-2N-N) \end{bmatrix}$$ (4.5)

with

$$L_2 = \begin{bmatrix} L_1 & 0 & \cdots & 0 \\ 0 & L_1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots \\ 0 & \cdots & 0 & L_1 \end{bmatrix}$$ (4.6)

This filter-matrix calculates only the elements of $\hat{c}_l$ needed for the next step (4.4). Combining the two steps results in:

$$c_{th}(k) = H_1 L_2 \begin{bmatrix} u(k) \\ u(k-1) \\ \vdots \\ u(k-2N-N) \end{bmatrix}$$ (4.7)

Note that $H_1 L_2$ is just a filter vector of length $2N + N + 1$ which act on the series $u(k) \ldots u(k-2N-N)$ to compute level 2 coefficient $c_{th}(k)$. The filter coefficients are products of filter coefficients from $H(z)$ and $L(z)$:

$$H_1 L_2 = \begin{bmatrix} h_1 l_2(0) & h_1 l_2(1) & \ldots & h_1 l_2(2N + N) \end{bmatrix}$$ (4.8)
Such construction can be made to compute a new coefficient on all levels of the decomposition. For example, \( c_{ll}(k) = L_1 L_2 \hat{u}(k) \) with:

\[
L_1 = \begin{bmatrix}
L(0) & L(1) & \ldots & L(N)
\end{bmatrix}
\]

(4.9)

It can be proven that the total number of old input samples needed for computation of a new coefficient at a certain decomposition level is defined by:

\[
l_u = \sum_{p=1}^{P} N 2^{p-1}
\]

(4.10)

\( P \) is the decomposition level and \( N \) the order of the original filters.

### 4.3 The synthesis bank

In general the reconstruction filters have the same order as the analysis filter. Only for bi-orthonormal wavelets (presented in Appendix B.2) the reconstruction filter can have a different order. After processing and downsampling the coefficients are denoted by \( c' \). For the level 1 reconstruction of only the highpass coefficients \( c'_h \) we can write:

\[
y(k) = \hat{u}(k - N) = \begin{bmatrix}
h'(0) & h'(1) & \ldots & h'(N)
\end{bmatrix}
\]

(4.11)

\[
\begin{bmatrix}
c'_h(k) \\
c'_h(k - 2) \\
\vdots \\
c'_h(k - N)
\end{bmatrix}
\]

We choose to use the newest coefficient \( c'_h(k) \) as the first input value for the filter to minimize the delay. Synthesis from coefficients on moments \( k \) to \( k - N \) outputs a reconstruction of the input at moment \( k - N \). Indeed, for a wavelet filter the delay is equal to the order \( N \) as reported in Appendix A.4. The zeros are caused by the down- and upsampling which is needed for a good reconstruction. This means throwing away all odd\(^1 \) coefficients \( (k - 1, k - 3, \ldots) \).

\[
\begin{bmatrix}
c(k) \\
c(k - 1) \\
\vdots \\
c(k - N)
\end{bmatrix}
\downarrow 2 \uparrow 2
\]

\[
\begin{bmatrix}
c'(k) \\
c'(k - 2) \\
\vdots \\
c'(k - N)
\end{bmatrix}
\]

(4.12)

However this does not mean that the odd samples are redundant; the next sample moment all coefficients shift one sample up so they become then even coefficients. This is why we cannot throw away any coefficients before storing them. We can see this back in Fig. 4.1 where the downsampling is applied in the synthesis bank, after storing and processing the coefficients.

If \( N \) is odd, a unit delay can be gained in the reconstruction process. We can use the following trick, caused

---

\(^1\) In this real-time application where the signal horizon is always moving forward we cannot really talk about even or odd samples. We define even samples as \( k, k - 2, \ldots \), so including the newest received sample/coefficient.
4.3 The synthesis bank

by the up/downsampling. The level 1 reconstruction (as in (4.11)) can also be written as:

$$\hat{u}(k - N) = \begin{bmatrix} h'(0) \\ 0 \\ h'(2) \\ \vdots \\ h'(N - 1) \\ 0 \end{bmatrix} \cdot \begin{bmatrix} c'_h(k) \\ c'_h(k - 1) \\ c'_h(k - 2) \\ \vdots \\ c'_h(k - N + 1) \\ c'_h(k - N) \end{bmatrix}$$

(4.13)

We see that the last filter coefficient $h'(N)$, or the wavelet coefficient $c'_h(k - N)$ is not used in the reconstruction (so actually we do not have to store that coefficient). If we now move our filter one sample forward, we gain one sample delay:

$$\hat{u}(k - N + 1) = \begin{bmatrix} h'(1) \\ 0 \\ h'(3) \\ \vdots \\ h'(N) \end{bmatrix} \cdot \begin{bmatrix} c'_h(k) \\ c'_h(k - 1) \\ c'_h(k - 2) \\ \vdots \\ c'_h(k - N + 1) \end{bmatrix}$$

(4.14)

Finally, we can write (4.14) in reduced form, using only the even wavelet coefficients, as stored in $\tilde{c}'_h$ in Fig. 4.3:

$$\hat{u}(k - N + 1) = H'_1 \begin{bmatrix} c'_h(k) \\ c'_h(k - 2) \\ \vdots \\ c'_h(k - N + 1) \end{bmatrix}, \quad H'_1 = [ h'(1) \ h'(3) \ \ldots \ h'(N) ]$$

(4.15)

This shift can be applied at any decomposition level without making errors in the reconstruction. Looking at higher level coefficients in the ordinary wavelet tree Fig. 4.4, we see that they are downsampled more times. Since we store wavelet coefficients at all times we need to watch out in the reconstruction process. For example look at the level 2 coefficients (say $\tilde{c}_{1h}$). After up and downsampling, using the ordinary structure, they are available for reconstruction as:

$$\begin{bmatrix} c_{1h}(k) \\ c_{1h}(k - 1) \\ \vdots \end{bmatrix} \Downarrow \Downarrow 2 \Rightarrow \begin{bmatrix} c_{1h}(k) \\ c_{1h}(k - 2) \\ \vdots \end{bmatrix} \Downarrow \Downarrow 2 \Rightarrow \begin{bmatrix} c'_{1h}(k) \\ c'_{1h}(k - 2) \end{bmatrix}$$

(4.16)
Just as in (4.15) we can reconstruct $c'_l$ (Fig. 4.4) coefficients from $c'_{lh}$:

$$c'_l(k - 2(N - 1)) = H'_1 \begin{bmatrix} c'_{lh}(k) \\ c'_{lh}(k - 4) \\ \vdots \\ c'_{lh}(k - 2(N - 1)) \end{bmatrix}$$  \hspace{1cm} (4.17)

Looking at the delay this reconstruction we see it is increased; we do not use more coefficients, but the time span is enlarged. For arbitrary decomposition level $p$ we have to use coefficients $c'_l(k)$, $c'_l(k - 2^p)$, $\ldots$, $c'_l(k - 2^p(N - 1))$ to calculate the 1-level higher coefficient $c'(k - 2^p(N - 1))$. For reconstructing the output from $c'_l$ (level 1) coefficients we need all even coefficients as in (4.15). Because $c'_l(k - 2(N - 1))$ is the newest coefficient available at that time the delay increases:

$$\begin{bmatrix} c'_l(k - 2(N - 1)) \\ c'_l(k - 2(N - 1) - 2) \\ \vdots \\ c'_l(k - 3(N - 1)) \end{bmatrix} = H'_2 \begin{bmatrix} c'_{lh}(k) \\ c'_{lh}(k - 2) \\ \vdots \\ c'_{lh}(k - 3(N - 1)) \end{bmatrix}$$  \hspace{1cm} (4.18)

with,

$$H'_2 = \begin{bmatrix} h'(1) & 0 & h'(3) & \cdots & h'(N - 1) \\ 0 & h'(1) & 0 & h'(3) & \cdots & h'(N - 1) \\ \vdots & & & & & \vdots \\ h'(1) & 0 & h'(3) & \cdots & h'(N - 1) \end{bmatrix}$$  \hspace{1cm} (4.19)

At any level we will always need all successive odd intervening coefficients in the range $c'_l(k)$, $c'_l(k - 2^p)$, $\ldots$, $c'_l(k - 2^p(N - 1))$. Because all coefficients are stored, after processing and downsampling they are available as $c'_{lh}$ in (4.11): $c'_l(k)$, $c'_l(k - 2)$, $\ldots$. This means that zeros have to be inserted in all reconstruction matrices (like $H'_2$). The number of zeros will raise proportionally with the reconstruction level as $2^p - 1$.

From the row of even level 1 coefficients in (4.18) we can reconstruct the output as in (4.15):

$$\hat{u}(k - 3(N - 1)) = L'_1 \begin{bmatrix} c'_l(k - 2(N - 1)) \\ c'_l(k - 2(N - 1) - 2) \\ \vdots \\ c'_l(k - 3(N - 1)) \end{bmatrix}$$  \hspace{1cm} (4.20)

with

$$L'_1 = \begin{bmatrix} l'(1) & l'(3) & \cdots & h'(N) \end{bmatrix}$$  \hspace{1cm} (4.21)
4.4 Assembling the total system

So the total reconstruction from level 2 coefficients looks like:

\[ \hat{u}(k - 3(N - 1)) = L_1' H_2' \begin{bmatrix} c'_{l_2}(k) \\ c'_{l_2}(k - 2) \\ \vdots \\ c'_{l_2}(k - 3(N - 1)) \end{bmatrix} \]  \hfill (4.22)

We see the total delay is raising rapidly if we use higher level decompositions. Using \( c_{l_2}(k) \), the newest level 2 coefficient, we can only reconstruct the output at \( k - 3(N - 1) \). For the total delay of a \( P \) level decomposition and filter order \( N \) we can find:

\[ l_d = \sum_{p=1}^{P} (N - 1)2^{p-1} \]  \hfill (4.23)

The gain of delay caused by the odd filter order \( N \) can be expressed as \( l_u - l_d \), using (4.10) and (4.23). Till now we have only reconstructed parts of the output (only contributions from \( c_k \) and \( c_{l_2} \)). For a full reconstruction we have to sum the contributions of all coefficients. Construction of all the matrices becomes very complex for high levels. Basically it is only placing zeros and filter-coefficients at the correct places. The process is automized in a MATLAB routine for arbitrary decomposition level \( P \).

4.4 Assembling the total system

From the previous section it is clear how to calculate the wavelet coefficients and the reconstruction at any level with a minimal delay. Now it is time to assemble the analysis and reconstruction bank. Using the system structure in (4.2), the state vector is defined as:

\[ \hat{x} = \begin{bmatrix} \hat{u} \\ \hat{c}_h \\ \hat{c}_{l_2} \\ \hat{c}_{l_3} \\ \vdots \\ \hat{c}_{l_{P-1}} \\ \hat{c}_{l_P} \end{bmatrix} \]  \hfill (4.24)

The states consists of old inputs \( \hat{u}(k) \) and old wavelet coefficients \( \hat{c}_{\cdot}(k) \) of several levels. The input vector at time \( k - 1 \) consist of \( l_u \) old inputs: the number of old inputs needed to calculate all coefficients at time \( k \).

\[ \hat{u}(k - 1) = \begin{bmatrix} u(k - 1) \\ u(k - 2) \\ \vdots \\ u(k - l_u) \end{bmatrix} \]  \hfill (4.25)

The vectors \( \hat{c}_{\cdot}(k) \) consist of most recent and \( l_d \) old wavelet coefficients of the different levels:

\[ \hat{c}_{\cdot}(k) = \begin{bmatrix} c_{\cdot}(k) \\ c_{\cdot}(k - 1) \\ \vdots \\ c_{\cdot}(k - l_d) \end{bmatrix} \]  \hfill (4.26)
Note that only for the highest decomposition level $P$ all coefficients are needed. For lower level reconstructions only the oldest coefficients are needed to ensure a synchronized synthesis of all levels. In spite of this we have to store coefficients of all levels from the most recent coefficient $k$ up to time $(k - l_d)$. The total delay for a certain wavelet filter is then $l_d$.

In the analysis step from (4.2) only the $A$ and $B$ are involved:

$$\tilde{z}(k) = A\tilde{z}(k - 1) + Bu(k)$$ (4.27)

First the square matrix $I_0$ is defined which moves down the elements of the input vector. This is needed to store the old inputs/coefficients preserving their time order.

$$I_0 = \begin{bmatrix} 0 & \cdots & 0 & 0 \\ 1 & \mathbf{0} & 0 \\ \vdots & \vdots & \vdots \\ \mathbf{0} & 1 & 0 \end{bmatrix}$$ (4.28)

Now we can fill $A$, which is responsible for shifting down old inputs/coefficients and calculating the influence of the old inputs on the new coefficients. Vector $B$ determines the influence of the new input on the new coefficients. Now (4.27) can be written as:

$$\begin{bmatrix} \tilde{u}(k) \\ \tilde{c}_b(k) \\ \tilde{c}_{th}(k) \\ \vdots \\ \tilde{c}_{p'}(k) \end{bmatrix} = \begin{bmatrix} I_0 & A_{H_1} & I_0 \\ A_{H_1L_2} & I_0 \\ \vdots & \vdots & \vdots \\ A_{L_{1...L_P}} & I_0 \end{bmatrix} \begin{bmatrix} \tilde{u}(k - 1) \\ \tilde{c}_b(k - 1) \\ \tilde{c}_{th}(k - 1) \\ \vdots \\ \tilde{c}_{p'}(k - 1) \end{bmatrix} + \begin{bmatrix} B_u \\ B_{H_1} \\ B_{H_1L_2} \\ \vdots \\ B_{L_{1...L_P}} \end{bmatrix} u(k)$$ (4.29)

Only the first (most recent) element of each level is influenced by submatrices $A_{..}$ and subrows $B...$. To illustrate this, i.e. $A_{H_1L_2}$ looks like:

$$A_{H_1L_2} = \begin{bmatrix} h_{1L_2}(1) & h_{1L_2}(2) & \cdots & h_{1L_2}(2N + N) & 0 & \cdots & 0 \\ \cdots \end{bmatrix}_{l_a + 1}$$ (4.30)

The first row consist of the filter coefficients from (4.8) acting on previous inputs. Note that for the highest levels ($c_{p' - 1}b$ and $c_{p'}$) the first row is totally filled with filter coefficients. Consequently the subrows in $B$
only consist of the first filter coefficient filled with zeros. For example, $B_{H_1L_2}$ looks like:

$$B_{H_1L_2} = \begin{bmatrix} h_1L_2(0) \\ 0 \\ \vdots \\ 0 \end{bmatrix}_{u+1}$$ (4.31)

The buffer-matrices $I_b$ with size $[l_d + 1, l_d + 1]$ on the diagonal of $A$ only move up the coefficients of a certain level. The first $I_b$ acting on the input memory has size $[u, l_u]$. Further $B_u$ looks like:

$$B_u = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}_{l_u}$$ (4.32)

All other elements of $A$ are zeros so $A$ is very sparse, especially for high decomposition levels and high filter orders. The total number of states, using (4.25) and (4.26), can be expressed as:

$$n_x = l_u + (l_d + 1)^{p+1}$$ (4.33)

From Fig. 4.5 we see that output matrix $C$ is responsible for the reconstruction of the original signal from the state vector:

$$y(k) = C\tilde{z}(k)$$ (4.34)

Upsampling in the reconstruction process is implemented in the reconstruction filter matrices (i.e. $H'_2$ in (4.19)). However, the downsampling process must be included in the reconstruction matrix $C$. Therefore a downsampling-matrix $D_2$ is defined as:

$$D_2 = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & 1 \end{bmatrix}_{l_u+1}$$ (4.35)

Because we only use odd ordered filters $H(z)$ and $L(z)$, $\frac{m+1}{2}$ is an integer. Now (4.34) can be written as:

$$y(k) = \begin{bmatrix} 0 & C_{H_1D_2} & C_{L_1H_2}D_2 & \cdots & C_{L_1\ldots L_mD_2} \end{bmatrix} \begin{bmatrix} \tilde{u}(k) \\ \tilde{e}_h(k) \\ \tilde{e}_{h'}(k) \\ \vdots \\ \tilde{e}_{h'}(k) \end{bmatrix}$$ (4.36)

For a synchronized synthesis we need only the oldest coefficients for the lower level coefficients. So the filter coefficients are preceded by zeros to give all subcolumns in $C$ length $l_d + 1$. For example $C_{L_1H_2}$ looks like:

$$C_{L_1H_2} = \begin{bmatrix} 0 & \cdots & 0 & L'_1H'_2 \end{bmatrix}_{l_u+1}$$ (4.37)
Again for the highest levels \((c_{L'_{1-1k}}\;\text{and}\; c_{L'_{P}})\) this rowvector is totally filled with filter coefficients. Using this output matrix \(C\) the output will be a delayed and possibly filtered version of the input according to:

\[
g(k) = \hat{u}(k - l_d)
\]  

\((4.38)\)

Before reconstructing the coefficients in the state vector can be processed. The total process of constructing the system matrices \(A, B\) and \(C\) is implemented in a MATLAB routine \texttt{rtwavfilt.m} which is presented in Appendix C. It is also possible to look separate to each reconstruction. Then an adapted output-matrix \(C\) can be used:

\[
y(k) = \begin{bmatrix}
\mathcal{O} & C_{H_1_1} \times D_2 & \mathcal{O} & \cdots & \mathcal{O} \\
\mathcal{O} & \mathcal{O} & C_{L_1_1H_2_2} \times D_2 & \cdots & \mathcal{O} \\
\vdots & \vdots & \ddots & \ddots & \vdots \\
\mathcal{O} & \cdots & \mathcal{O} & C_{L_1_1\cdots L_{P_P}} \times D_2 & \mathcal{O}
\end{bmatrix}
\begin{bmatrix}
\hat{u}(k) \\
\hat{c}_h(k) \\
\hat{c}_{lh}(k) \\
\vdots \\
\hat{c}_{lP}(k)
\end{bmatrix}
\]

\((4.39)\)

In this way a multiple output system is created. If the sequences are summed the input will be reconstructed. It is also possible to output the reconstructions using the shortest delay times for each level. This can be used for time-critical analysis applications.

It is also possible to output directly the new coefficients at all levels without storing them. In this way online analysis is possible with the shortest delay time for each decomposition level. Now DWT-waveforms of Appendix B.2 can be used for analysis purposes. This structure can build with \texttt{rtwavana.m}, also presented in Appendix C. The routine outputs a filter matrix \(F\) which can be implemented as a multiple output FIR filter. So a real-time wavelet analyzer is created.

\[
F = \begin{bmatrix}
H_1 \\
H_1 \times L_2 \\
\vdots \\
L_1 \times \cdots L_{P_P}
\end{bmatrix}
\]

\((4.40)\)

Besides that the presented structures can be used on-line they have another advantage. Because that all coefficients are updated every sample hit, the time-resolution is increased. For both the filter structure as the analyzer the time duration can be expressed as:

\[
\Delta t = \frac{1}{f_s}
\]

\((4.41)\)

This is equal to the sample time of the system, the minimal value for a discrete-time system. Especially for higher decomposition levels the resolution is increased compared with the ordinary DWT algorithm, where these levels are highly downsampled. So this non-redundant approach, because of the lack of downsampling, creates a real-time wavelet filter and analyzer with maximum time-resolution.
Applications in control engineering

Having explored the general application fields for wavelets it is now time to look if problems in control can be tackled with either wavelet analysis or the wavelet filter technique. First the advanced filter characteristics of the DWT are presented and denoising is explained in an example. Then the possibilities for in-the-loop applications are investigated, using the results from the previous chapter. After that, two worked-out applications are presented. The first application uses the filter capacity of the DWT for the denoising of a special class of signals. In the second the analyzing properties of the wavelet transform are used in a real-time feature detection application.

5.1 Wavelet filter properties

It is clear now that the DWT process can be seen as filtering (Chapter 3). Most (discrete) filters normally used in control engineering are linear time-invariant filters. The magnitude of the transfer function changes with frequency and depends linear on the amplitude: $|H(s = 2\pi f)|$ or $|H(z = e^{2\pi i f})|$ in discrete time. Normally they have nonlinear phase behaviour. They can have linear phase but then some group delay will be present.

With a wavelet filter (and also a filter bank) it is possible to define an arbitrary decay for the amplitude in a certain frequency band. So it is possible to create almost any shape in the magnitude-plot of the transfer function of the filter. In the DWT this is a function of the frequency band (decomposition level $p$) and the values of the coefficients, representing the amplitudes of the inputs:

$$|H(f)| = \left| \sum_p H_p(c_p) \right|$$  \hspace{1cm} (5.1)

Nonlinear transfer functions are possible, however very exotic designs are not found in literature, which is probably due to amplitude and phase disturbances at the borders of the frequency bands. The alias and disturbance cancellation for QMF structures is only guaranteed if the coefficients are not processed. Every filter action will create disturbances at the borders. Contrary to most LTI designs, wavelet filters have linear phase behaviour which results in a certain group delay.
5.1.1 DWT example: denoising

In general denoising is accomplished using thresholding. On every level of the decomposition a threshold $\delta$ is applied. Coefficients below the threshold are omitted in the reconstruction. By doing this, the low-amplitude regions in certain frequency bands, presumably due to noise, are suppressed. The noise power is assumed to be smaller than the signal power but also then, it is impossible to filter out all the noise without affecting the signal.

The MATLAB Wavelet Toolbox offers an easy-to-use tool for denoising with wavelets. It includes algorithms that can automatically select the various threshold levels assuming a certain noise structure, i.e. white noise. There are several other threshold parameters that can optimize the denoising, which are clarified in the toolbox help files and in [MMOP00]. It is also possible to set the threshold values for every level manually. The lower branch of the tree, which iterates to the scaling function, is usually not changed for denoising purposes. In the Wavelet Toolbox this level is called the Approximation; the other levels are Details, numbered with the decomposition level $p$.

Denoising is always done using soft-thresholding (Fig. 5.2.b). Only ignoring coefficients below the limit $\delta$ without affecting values just above causes irregularities in the reconstruction. This is called hard-thresholding (Fig. 5.2.a). To make a continuous transition at the threshold, all other coefficients are scaled. Soft-thresholding causes a little shrinkage of the signal since all coefficients are decreased a little. However, if the threshold is low relative to the average coefficient level this effect is negligible. Suppose we want to clean the real-life signal in Fig. 5.3. This signal has wide frequency support which makes it not easy to denoise it. Before the decomposition can start, the waveform and decomposition level have to be chosen. The decomposition determines the lowest frequency that can be observed in the DWT. All noise under this frequency remains unchanged and is reconstructed. With $P$ the decomposition level, this frequency is...
5.1 Wavelet filter properties

There are plenty of DWT-waveforms that can be used for denoising. The waveform determines the quality of decomposition since the wavelet coefficients are a measure for the resemblance between signal and wavelet. The best results are obtained when the waveform fits good to the signal. For most signals, especially in control, a smooth waveform must be chosen. Most measurements in control are produced by causal systems which have lowpass behaviour, so their outputs will be smooth. However, there will always be a trade-off between smoothness of the waveforms and computation time, since higher order wavelets are smoother. In this example, more or less arbitrary, the db4 wavelet (see Appendix B.2) is chosen. The 8-level decomposition is shown on the next page. From level $C_1$, which projects the highest frequencies, it can be seen that there is not only noise in this frequency band. So it would be impossible to denoise this signal with a lowpass filter. From the Approximation it is clear that the basic shape of the signal is composed of frequencies below 19.5313 Hz (Fig. 5.4). This signal is obtained by reconstructing only the lowest channel, so all levels in Fig. 5.5 are completely suppressed. This could also be the result of a lowpass filter.

From the decomposition it is clear where the thresholds must be placed. Using the automatic thresholding routine from MATLAB tuned for white noise, a reasonable reconstruction is obtained. Setting the thresholds by hand can improve the result. The horizontal lines in Fig. 5.5 show the threshold levels, set by hand. Reconstruction with soft-thresholding result the signal of Fig. 5.6. Apart from the interval $5.5 < t < 7.5$, all high-frequency content (noise) is cancelled. In Fig. 5.7 the reconstruction of a high-frequent part is visible. The error (Fig. 5.7.b) shows only a slight disturbance due to the soft-thresholding.

This technique has the advantage over traditional filtering in that it can remove noise at all frequencies. The threshold values can be adjusted separately for every level, so the noise may have different intensities in the several frequency bands. Furthermore, the wavelet denoising show better behaviour for signals with short bursts. A normal lowpass filter (and also filter banks) would smear out such short phenomena. Because of the good time resolution for high frequencies of the DWT, such bursts can be located (and reconstructed) very well. Furthermore, wavelet filters are especially suitable for denoising signals with a large spread in frequency content.
Fig. 5.5: Coefficients of the 8-level DWT decomposition with a db4 wavelet. The coefficients represent the signal in the indicated frequency bands.
5.2 Wavelets in control loops

In the previous section a real-time algorithm is derived. Now wavelet filters can be used on-line. Unfortunately, the group delay of wavelet filters is considerable (4.23). The next table shows the delays using a 5th-order wavelet (db3):

<table>
<thead>
<tr>
<th>Decomposition level ( p )</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delay ( t_d )</td>
<td>4</td>
<td>12</td>
<td>28</td>
<td>60</td>
<td>124</td>
<td>252</td>
</tr>
</tbody>
</table>

Little decomposition depth already results in huge delays. With this, the hope to use them as controllers disappears. Performance of controllers deteriorate dramatically even with small delays. Using higher sample rates in the controller to decrease the delay times has no effect. Since oversampling shifts up the frequency regions of the filter (Fig. 3.11), a larger decomposition level \( p \) is needed to reach frequency bands of interest and the apparent profit in delay is cancelled. Also extrapolation of measurement data for control purposes has not been successful in the past, so building controllers with wavelet filter is not possible. Normal filter banks are also unsuitable for control purposes. They only have one decomposition level, so the delay is fairly small compared with the DWT. However, if the number of channels increases the filter order must also increase to construct good filters [Sch95]. Except for controllers there are on-line structures possible where a real-time wavelet implementation can succeed. It can be useful in supervisory loops (Fig. 5.8) to adapt the actual controller based on certain features in the measurement. On-line denoising outside the control loop can also be interesting. The adaptation loop in an adaptive control setup may have some delay. Using a wavelet filter for advance denoising of the measurement, it can be used in a self-tuning regulator (Fig. 5.8.b), if the controller output \( u \) is delayed. An extended Kalman filter as parameter estimator benefits greatly from a noise-free input. The wavelet filter can also be used for other adaptive control strategies as model reference adaptive control (see [dJLV96]).
5.3 Application I: encoder-quantization denoising

Encoders, widely used in motion systems, always generate noise which is especially annoying when derivatives of the measurement have to be calculated. Using a lowpass filter can give good results in some cases but not always. To find out if wavelet filters can be appropriate for denoising encoder signals, some study in the principles of encoders and quantization is done.

5.3.1 Encoders and quantization

An encoder uses a disc or strip with a certain graduation, the measuring standard. From one side light is projected onto the pattern and received on the other side with photovoltaic cells. Moving the measuring standard produces a periodic fluctuation of light intensity, which is detected by the photo-cells. Such scanning of a single periodic graduation results in an incremental (counted) measurement. Since most applications require some absolute reference, most encoders have at least one reference mark. This way, the absolute reference position is permanently assigned to exactly one measuring step. The fluctuation in intensity results in a periodic voltage from the photocells. Most encoders provide at least two sinusoidal, 90° phase-shifted measuring signals (Fig. 5.10.a). For use with DSP systems this signal is converted to a TTL signal: a pulse train (Fig. 5.10.b). The only actual information on position or angle is a counter, which is raised if a grating bar on the measuring standard is crossed. Then the position or angle is changed with $\Delta$, the encoder stepsize. The raw encoder signal is used for accurate off-line position and velocity reconstruction. It can be used to study friction behaviour in motion systems (i.e. presliding). Actually the signal in Fig. 5.3 is a raw signal from a pre-sliding measurement.
5.3 Application I: encoder-quantization denoising

5.3.2 Quantization errors

The encoder TTL-output is similar with a quantiser in digital systems: information of a signal is only presented on discrete values (appendix A.5). When processing measurements, usually quantization errors are handled as input-signal independent, uniformly distributed white noise. Most of the quantization effects can be cancelled (together with other noise phenomena) by lowpass-filtering the data. This treatment fails in case signal is changing slowly. In fact, quantizing errors are not random but deterministically related to the input of the quantiser, the actual position/angle. To obtain more information on quantizing errors, some experiments are carried out with artificial signals (i.e. Fig. 5.11) that are quantized (Fig. 5.12). Investigating the spectrum of the quantizing error for harmonic input signals shows that this are related to the frequencies of the harmonics. Also the amplitude of the harmonics has influence on the frequency of the error: the bigger the amplitude of the signal, the bigger the frequency in the error signal. It can be concluded that the frequency of the quantizing error globally depends on the velocity of the input signal and the resolution (5.3). Indeed, this justifies the fact that lowpass filtering can be used successfully with high velocities and a small quantiser stepsize. The frequencies of the quantization noise are high enough in comparison to the frequency content of the signal.

\[ f_q = \frac{\dot{x}}{\Delta} \]  

(5.3)
Although quantization errors are deterministic, their spectrum is nonlinearly related to the spectrum of the input signal. So it is not easy to find LTI-filters with good overall performance, because they only act frequency-dependent.

In the field of signal processing a technique is available to suppress the errors in critical situations (low velocities): dithering. Dithering means to tremble or to vibrate the quantization process. To disturb the deterministic behaviour of the quantiser (which has fixed decision and representation levels) the noise is raised and so the uncertainty of the quantized signal. After quantization this effect virtually raises the quantization resolution. In order to obtain this effect, the dither must have a random nature (more information on dither is presented in [Sch95]). The problem with encoders used as quantisers is that the input of the quantiser is not available. The only way to influence the encoder before information is quantized is to dither the measurement standard or the photocells, which may be another interesting research-subject.

Note that time sampling is completely unrelated to the errors coming with quantization effects. If our input signal is strictly band-limited according the Nyquist criterion, this time quantization is no limitation for reconstructing the signal. However if the sample frequency is lowered, the quantization effects become more random, but there will be information loss because of undersampling. It is assumed that the sample frequency is high enough in our measurements.
5.3 Application I: encoder-quantization denoising

5.3.3 Filtering quantization effects with the DWT

The artificial signal in Fig. 5.11 has a wide spectrum and also a wide range of velocities $\dot{x}$. The quantization error in Fig. 5.13 also has a wide frequency spectrum which in uncorrelated with that of the original signal. Indeed, Fig. 5.14 shows a wide noisy spectrum.

Understanding the nature of quantization errors, the DWT seems the perfect filter for denoising encoder signals since it can filter out noise at all frequencies. We only have to determine the thresholds for the several decomposition levels. A very interesting property on quantization errors which is not mentioned yet is that their amplitude is always the same, equaling the encoder stepsize $\Delta$. Therefore, the threshold $\delta$ can be the same for all levels. The actual value is a little bit dependent on the used waveform. If the waveform shows good resemblance with the quantization-error signal, this value must be close to $\Delta$. This
can be derived from the orthonormality property of DWT-wavelets (Appendix A.2). Experiments show that the choice of the waveform is critical in this denoise process: using arbitrary smooth wavelets did not result in noise-free reconstructions. The search for an optimal waveform is automated using a cost-function on the reconstruction error $e_q = x_r - x$. The best results are obtained with the bi-orthogonal bior5.5 wavelet (Appendix B.2).

Fig. 5.15: Coefficients of a 6-level decomposition using bi-orthonormal 5.5 wavelet

Because of the low speeds in signal $x$, 6-level decomposition is needed, which is presented in Fig. 5.15. The reconstructed signal is shown in Fig. 5.16. The wavelet filter is able to cancel quantization errors at very low speed and relative high stepsize (Fig. 5.17.a). At high speeds there is less progress (Fig. 5.17.b). Here the quantization errors are already small, and the DWT creates more amplitude distortion due to the soft thresholding.
5.3 Application I: encoder-quantization denoising

Fig. 5.16: Details of the reconstruction using coefficients of Fig. 5.15, uniformly thresholded at \( t = \Delta = 0.2 \). Original signal is dotted.

Fig. 5.17: Errors between reconstruction \( x_r \) and original signal \( x \).

The results at high-speed are somewhat disappointing. Actually, these speeds are very high. Comparing the result with the ordinary lowpass filtering of encoder signals in Fig. 5.18, the results are not bad at all. For low velocities the output of an LTI-filter follows the encoder steps. The encoder steps are too far away from each other which can be seen as a combination of unit-step signals. The result is a cumulative set of stepresponses created by the filter. To cancel this effect the cutoff-frequency must be lowered, but then high-velocity parts will be totally suppressed.
The denoising using DWT also works because of the multiresolution property. If the filter length would not be increased for lower frequencies it would be impossible to detect low frequencies in a quantization-error signal. It is expected that filter banks are not able to cancel quantization errors in such wide range of speeds. For very low speeds, if the amplitude of the input signal is comparable to the quantiser stepsize, the DWT cannot bring help. In that case only dithering can help, or the raw encoder signal should be used.

To conclude this section, the total procedure for denoising a TTL-encoder signals is summarized in three steps:

- Determine the lowest speed in the signal. With this speed the lowest frequency in the quantization error can be estimated using (5.3). Now the decomposition level \( P \) can by rewriting (5.2).
- Adjust threshold-levels at all levels to the encoder stepsize: \( \delta = \Delta \).
- Select the bior5.5-wavelet for decomposing the signal. Apply soft-thresholding and reconstruct the signal.

5.4 Application II: real-time feature detection on a CD-player setup

5.4.1 Feature detection

In general, feature detection is a technique to extract or isolate information from a signal that is important to a certain application. It is well known in the field of image processing were algorithms to detect edges (one dimensional feature detection) and corners (two dimensional feature detection) are used.

A time-frequency representation can act as detector. If events differ from each other by their frequency content they can be separated. The disadvantage of the STFT is that the features can only be discriminated by their frequency content. Besides a spectrum, features have a certain time pattern which is more characterizing and discriminating then their spectrum. Wavelets have different shapes which can be more discriminating to some signals then sinusodials. The multiresolution approach also benefits detection: short features have high frequency content while slow and long lasting events consist of low frequencies. Because of these properties the WT seems suitable for this task.

Wavelets are already used in several fields for this purpose, even in engineering (i.e. [CA98]). However, all applications found in literature are not time critical: the signals are processed off-line or quasi on-line in a...
delayed security loop. This application shows an on-line wavelet-based feature detection technique using the real-time wavelet analyzer from Chapter 4. It can be implemented in a supervisory loop (Fig. 5.8.a) i.e. to adapt the controller if certain unwanted events disturb the closed-loop.

5.4.2 Real-time feature detection

For successful detection the detector must own good indicator properties: there should be a minimum number of false warnings and all actual features should be isolated. For successful on-line detection there is another requirement: fast detection. Because the features normally decrease the performance of the system, interference by adapting the controller must be done as fast as possible. On the other side, the accuracy of the detection will be optimal if the feature can be totally detected. So there will always be a balance between accuracy of detection and detection speed. Another part of feature detection is the design of a decision rule, that can decide whether a feature is detected based on information of the indicator. In the case of a wavelet filter the decision rule can be a simple threshold: if the value of the coefficients of a certain level rise above a threshold value $\theta$, the feature is said to be detected.

Some simulations with closed-loop systems to which artificial disturbances are applied, have been carried out. The real-time wavelet filter performs very well in fast detection of high frequent disturbances, even below the noise floor. In the next section, the filter is tested on a real-life setup: a CD-player.

5.4.3 CD-player setup

A CD-player is a complex system in which high-tech mechanical, control and digital signal processing techniques are combined. Although a perfect system in sound experience and data storage, there is always something left to be desired. Compact discs suffer from imperfections due to reckless handling: scratches on the surface are no exception. Furthermore there are situations in which the read process is subject to dynamic excitations (i.e. in portable and car CD-players). In these cases the read-out process of the data on the disc can be disturbed severely.

In this process two servo-loops are involved: one for focusing and one for radial positioning of the laser-unit. The radial positioning loop controls the following of the track and receives input from a reflected laser beam, which is also used for reconstructing the data on the disc. Scratches on the disc have much influence on this measurement, the radial-error signal ($y_m$ in Fig. 5.19), and disturb the radial-positioning servo-loop. Since the actual mechanical system is not disturbed, the controller gets wrong input: disc-scratches can be seen as measurement noise $d_m$. On the other hand, dynamic excitations, as shocks against the frame of the CD-player, cause actual errors in the read-out process. External shocks can be modeled as system disturbances $d_s$. In bad cases the normal controller is not able to correct for the shocks and as a result the track is lost. For both $d_m$ and $d_s$ it would be useful to detect each of these disturbances in an early stage. This way the controller can be adapted in a dedicated way to cancel the consequences of both effects.

![Fig. 5.19: Proposed setup with on-line feature detection in the servo-loop (C: controller, P: plant, W: supervisory controller with wavelet analyzer).](image)
5.4.4 Characterizing disturbances

To develop a better understanding of the nature of the different disturbances, a variety of error signals \( y_{m} \) of the servo-loop are investigated (i.e. Fig. 5.20). To investigate the real-time properties of the wavelet analyzer from Chapter 4 in quasi real-time, a MATLAB Simulink model is used.

\[
\begin{align*}
F(z) & = 1 \\
\end{align*}
\]

**Fig. 5.20:** Example of an error signal of the radial positioning servo-loop, sampled at 25.6 kHz. The effects of shocks as well as scratches are present in the signal.

First, focus is put at surface-defects. These are best isolated in the level 1 decomposition coefficients: disc-scratches have high-frequent influence on the radial servo-loop. The analysis is executed with several waveforms. The Haar-wavelet (db1), with first order filters, provides the shortest delay time in the wavelet analyzer but is not able to separate between the effects of shocks and scratches. The Daubechies db3 waveform (5th-order filter) has more delay, but shows good resemblance with the effects of disc-scratches. Based on the level 1 coefficients \( C_1 \) in Fig. 5.22, a threshold is determined. Again this is a compromise between speed of detection and accuracy. The close-up in Fig. 5.23 shows that the filter reacts on the feature before the error signal shows significant rise. The wavelet filter provides faster detection than a simple threshold-based decision on the original error signal.
5.4 Application II: real-time feature detection on a CD-player setup

Fig. 5.22: Wavelet analysis of the radial-error signal presented in Fig. 5.20. Decomposition depth \( p = 6 \) using the db3-wavelet. Real-time wavelet analyzer is used: coefficients are presented with minimal delay for each level and are updated every sample.

Shocks are best detected in level 3 or 4. From Fig. 5.24 it is clear that the db3-wavelet gives too much delay for early detection. The waveform db3 is not sufficiently related to the pattern created by the shocks. Maybe the number of cycles in the db3-wavelet is too large. Now the goal is finding a good waveform for detecting shocks.
5.4.5 Building dedicated waveforms

In order to get maximum performance with respect to feature detection it is desirable to use waveforms that show good resemblance with the features in the signal. What a wavelet filter actually does is measuring resemblance between its impulse response and a part of the signal. The best detection is possible if the impulse response is exactly equal to the inverse of the feature’s time-pattern. So it must be possible to build own waveforms that own better properties than the standard waveforms from appendix B.2 or B.1. All the special properties (i.e. orthonormality) are not needed if only analysis is concerned.

Not all possible waveforms can be chosen. Just as all wavelets, the mean value must be zero (actually they have to fulfill (2.19)). The best waveform is a fingerprint of the characteristic of the whole feature. However for this application detection delay is also an issue. As mentioned before, a trade-off between the analyzing length and the accuracy of the detection has to be made. The effects of scratches on the radial error signal can easily be determined and isolated to construct some good waveforms. Disc scratches show a defined pattern in the error signal that starts to oscillate around the sample frequency. It may be
noticed that this effect could be caused by the sample frequency, but this is not true in this case. Different measurements of the effect of one special scratch show that the oscillations are characteristic for the shock (Fig. 5.25). Processing different scratches show similar results. However, only averaging them cancels out the specific characteristics. Therefore, a dedicated waveform is derived a little pragmatic (Fig. 5.26). It is based on the db3 filter, which performs not bad at all.

Unfortunately the outcome of external shocks on the error signal is harder to detect. An additional set-up is built to apply conditioned shocks with the intention of creating reproducible measurements. Averaging these measurements will reveal the deterministic influence of the shocks and suppress other closed-loop disturbances (effects due to the disc characteristics or system/measurement noise).

An electro-dynamic shaker is driven by a power amplifier that receives a step-signal from a DSPT Siglab™. The generated stepped signal is transformed by the shaker dynamics into an impulse response of its second order systems. The shaker is designed for harmonic excitations which means that step-like signals cannot be transduced. However, the shaker is not fixed on the CD-player frame. Adjusting the distance between shaker and frame to realize contact at almost full-stroke will produce an impulse-like force on the frame. A softened tip will conduct only low frequencies which shows more resemblance with real-life shocks. A capacitive accelerometer provides feedback on the actual shock that is presented to the CD-player mechanism. Using this measurement the presented step signal is tuned to prevent multiple contact moments between shaker and frame.

Operating in standard mode (using the default servo-loops for the focusing and positioning) the shocks are presented. Using a trigger on the step-signal will enable good reproduction. The experiment is performed for 3 shaker-positions using different shock amplitudes. Fig. 5.28 shows information on the acceleration of the frame due to the shocks. Averaging 10 measurements each time cancels out other non-deterministic influences. Capturing about 50 different measurements (altering amplitude of the shocks, disc and attack
angle), show that the resulting influence on the radial-error signal is almost the same. Results are presented in Fig. 5.29. Now a dedicated waveforms is derived to make early detection possible, which is shown in Fig. 5.30. For shock-detection much lower sample rates are sufficient for accurate detection compared with scratch detection.

![CD-player setup-cd for investigating shock detection](image)

Fig. 5.27: CD-player setup-cd for investigating shock detection

---

(a) Acceleration pattern of the frame. (b) Frequency spectrum of the acceleration (peaks around 90 Hz, 155 Hz, 180 Hz and 235 Hz).

Fig. 5.28: Information on the conditioned shocks.
This waveform is validated on several data sets with real-life shocks. Fast detection is possible only with shocks that show much resemblance with the shocks presented by the shaker. Disturbances with other frequency content show this in the radial error signal: the main frequencies in the shocks are also visible in the radial-error signal. This makes it not easy to find one dedicated wavelet for overall shock detection. A solution to this problem could be a parallel setup of several filters for the same feature. It would have been nice if radial-error showed a specific pattern at the begin of shocks (indicated with ‘?’ in Fig. 5.29). Then very early detection would be possible. However, this is not the case, else it had been isolated by the averaging and would have been visible in Fig. 5.29.

Another possibility is to use an adaptive feature detector. In such setup the analyzing filter is adapted to the specific features presented at that time. The detector will be able to perform better if certain features occur periodically, which is often the case with scratches and shocks.
Conclusions & Recommendations

After reading this report it may be clear that wavelet analysis is a powerful tool for frequency analysis and feature detection. Although feature detection is not really a control issue, it is related to certain control problems. With the wavelet analyzer from Chapter 4 it is possible to detect disturbances in real time with maximum time-resolution. It is possible to adapt controllers in a very early stage. The feature detector can be seen as an identification technique for disturbances, which is a hot topic in control engineering. For periodical disturbances the optimal waveform in the detector can be adapted, and an adapted feature detector is born. A possible next step is to implement the analyzer in a complete supervisory system on an existing control system, i.e. a CD-player setup. An adaptation algorithm for the controller can be developed and tested.

On-line detection of scratches in the CD-player setup is possible and shows hopeful results. However, for robust detection more data are needed (i.e. use of various CD-players). Shock detection is less promising: for accurate detection there are at least more parallel detectors needed. The nature of shocks is a little inapplicable, and it is not clear if a robust feature detector can be build for this type of disturbances. Although real-time analysis is possible, the presented routine is not optimized at all for usage on a DSP-system. The created algorithm is just a first step to make real-time calculation possible. In special the memory handling can be improved a lot, if the algorithm is properly programmed.

Just as the DFT for Fourier analysis, the DWT can be seen as the most efficient, and therefore widely used version of the wavelet transform. Moreover, the reconstruction capability gives the DWT very nice properties if it is used as a filter. However, the delay time is considerable and a wavelet filter cannot be used for control purposes. The optimized denoising technique for encoder signals shows good results. Also for big quantization-levels a smooth output is obtained. Besides thresholding, other possible methods for processing the coefficients can be considered. Therefore, further investigation in the possibilities of nonlinear filtering is desirable.

The field of signal processing owns a lot of interesting techniques, also for control engineers. Filter banks are only shortly described; their variety is big and they offer great possibilities for filter purposes. Just as with wavelets, real-time filter banks are possible. In this report only perfect reconstruction orthonormal filter banks and wavelets are mentioned. But with some more freedom on the design rules their possibilities increase. After all, efficiently is not the number one requirement. So more research to dedicated filter banks is recommended. To construct filter banks, the theory must be studied closely which will take a lot of time. Besides this, MATLAB does not provide a Filter Bank Toolbox for an easy start in this field. Consulting someone with more signal processing background will accelerate the quest to optimal filter banks.

Finally, the Matlab Wavelet toolbox is a good start to explore the possibilities of wavelet analysis and wavelet filtering. It is equipped with the most standard wavelet routines needed for general applications. All routines are implemented for off-line usage. From a signal processing kind of view this is convenient. For on-line purposes in control-loops, delay times are critical. Therefore, the real-time algorithm is maybe
the most renewable in this report. It can be used for future research to find dedicated wavelet applications in control.
References


REFERENCES

[GR01] Roger Ghanem and Francesco Romeo.
A wavelet-based approach for model and parameter identification of non-linear systems.

An introduction to wavelets.

Tutorial on continuous wavelet analysis of experimental data.
The preparation of this document was supported in part by NASA-Dryden (TM: Martin Brenner) through Creare Inc (Drs. Miller and Magari), April 1995.

[Mal89] Stephane Mallat.
A theory for multiresolution signal decomposition: the wavelet representation.

[MMOP00] Michel Misiti, Yves Misiti, Georges Oppenheim, and Jean-Michel Poggi.
*Wavelet Toolbox Users Guide*.
Wavelet Toolbox, for use with MATLAB.

[MMPW97] Mark J. Mears Marios M. Polycarpou and Scott E. Weaver.
Adaptive wavelet control of nonlinear systems.

On-line architecture for the s.mallat algorithm.

[Pol99] Robi Polikar.
The wavelet tutorial.
The Engineer's ultimate guide to wavelet analysis.

*Digital Signal Processing*.

[RV91] Olivier Rioul and Martin Vetterli.
Wavelets and signal processing.

[Sch95] D.W.E. Schobben.
Dithering and data compression.

*Wavelets and Filter Banks*.

[Swe96] Wim Sweldens.
Wavelets and the lifting scheme: A 5 minute tour.

[TC98] Christopher Torrence and Gilbert P. Compo.
A practical guide to wavelet analysis.

[Vai93] P.P. Vaidyanathan.
*Multirate Filter Banks and Wavelets*.
Appendix A

Theoretical backgrounds

A.1 Uncertainty principle

The inequality (A.1) derives its name (Heisenberg’s) uncertainty principle from its interpretation in quantum mechanics. Observations of subatomic particles show they behave like waves with spatial frequency proportional to particle momentum. The classical laws of mechanics enable prediction of the future of a mechanical system by extrapolation from the currently known position and momentum. But because of the wave nature of matter, with momentum proportional to spatial frequency, such prediction requires simultaneous knowledge of both the location and the spatial frequency of the wave. This is impossible; hence the word uncertainty.

\[ \Delta f \Delta t \geq \frac{1}{4\pi} \]  

(A.1)

For the time-frequency representation this means that it is impossible to accurately localize both time and frequency. Although it is easy to verify the uncertainty principle in many special cases, it is not easy to deduce it. The difficulty begins from finding a definition of the width of a function that leads to a tractable analysis. One possible definition (presented in [RV91]) uses a second order moment of the weighting function \( g(t) \), that is defined by:

\[ \Delta t = \sqrt{\frac{\int t^2 |g(t)|^2 dt}{\int |g(t)|^2 dt}} \]  

(A.2)

This value is known as the time duration or RMS duration of the function \( g(t) \). Practically this means that two pulses in time can be discriminated only if they are more than \( \Delta t \) apart. The spectral bandwidth is defined likewise:

\[ \Delta f = \sqrt{\frac{\int f^2 |G(f)|^2 df}{\int |G(f)|^2 df}} \]  

(A.3)

Given window function \( g(t) \) and its Fourier transform \( G(f) \), the bandwidth \( \Delta f \) defines the frequency-difference of two sines which will just be discriminated. With these definitions, Dennis Gabor prepared a widely reproduced proof for (A.1) which is omitted here. The Gaussian function is often used because it meets the bound \( \Delta f \Delta t = \frac{1}{c} \). A Gaussian has the same shape in both time and frequency domain so it is most concentrated in both domains. It can be proven that for both short-time Fourier analysis and the wavelet transform the bandwidth-time product is constant:

\[ \Delta f \Delta t = c \]  

(A.4)
A.2 Orthonormal bases

Two vectors $v, w$ are said to be orthogonal if their inner product equals zero:

$$\langle v, w \rangle \equiv \sum_n v_n w_n = 0 \quad \text{(A.5)}$$

Similarly, two (real) functions $f_1(t)$ and $f_2(t)$ are said to be orthogonal to each other if their inner product is zero:

$$\langle f_1(t), f_2(t) \rangle \equiv \int f_1(t)f_2(t)dt = 0 \quad \text{(A.6)}$$

Orthonormality means that two (or a set of) vectors/functions are all mutually orthogonal and their energy is 1. For function we can express this as:

$$\int f_n(t)f_m(t)dt = \begin{cases} 1 & \text{if } n = m \\ 0 & \text{if } n \neq m \end{cases} \quad \text{(A.7)}$$

If a set of functions fulfill (A.7), they form an orthonormal base. As with vectors we can also project a signal onto a set of basis functions:

$$c_n = \int x(t)f_n(t)dt \quad \text{(A.8)}$$

The result is a set of coefficients $c_n$. If we use an orthonormal base, an arbitrary signal can then be represented exactly as a weighted sum of the basis function. This is called reconstruction:

$$x(t) = \sum_n c_n f_n(t) \quad \text{(A.9)}$$
This also holds for discrete-time signals, which can be seen as vectors actually. An orthonormal projection implies that there is no redundancy in the transformation. On the other hand, a non-redundant transformation does not automatically mean that the used base was orthonormal/orthogonal.

Advantages of an orthonormal projection are:

- the coefficients are independent of each other.
- there is no energy loss: the energy of the original signal is exactly stored in the coefficients.
- easy computation: the projection to coefficients and reconstruction to the original signal can be separately calculated for each single coefficient.

Bi-orthonormality looks very similar, only the used bases for projection and reconstruction are not the same. If projection looks like:

\[ c_n = \int x(t)f_n(t)dt \tag{A.10} \]

and reconstruction equals:

\[ x(t) = \sum_n c_n g_n(t) \tag{A.11} \]

the projection is bi-orthonormal if:

\[ \int f_n(t)g_m(t)dt = \begin{cases} 1 & \text{if } n = m \\ 0 & \text{if } n \neq m \end{cases} \tag{A.12} \]

Now (A.7) is not satisfied and:

\[ \int f_n(t)f_m(t) = 0, \tag{A.13} \]
\[ \int g_n(t)g_m(t) = 0 \]

is not expected for every \( n \neq m \).

### A.3 System representations

The normal state-space representation for discrete-time systems:

\[
\begin{align*}
\hat{x}(k+1) &= A\hat{x}(k) + Bu(k) \\
y(k) &= C\hat{x}(k) + Du(k)
\end{align*}
\tag{A.14}
\]

Alternative layout:

\[
\begin{align*}
\hat{z}(k) &= \Gamma z(k-1) + \Phi u(k) \\
y(k) &= \Psi \hat{z}(k)
\end{align*}
\tag{A.15}\]
System from (A.15) can be expressed in the standard representation using:

\[
\begin{align*}
A &= \Gamma, & B &= \Phi \\
C &= \Psi\Gamma, & D &= \Psi\Phi
\end{align*}
\]  

(A.16)

A normal discrete-time state space model can also be written as (A.15). Rewriting this system to (A.14), a unit delay can be gained compared with the normal state-space representation.

### A.4 Design rules for QMF banks

Wavelet filters are based on QMF structure (Fig. 3.1). Perfect reconstruction is a crucial property. To show that this is possible in spite of non-ideal filters with transition bands some basic design rules are shortly presented (more explanation on design rules in [SN96], chapter 4). In general and also in the case of wavelet filters the filters \( H(z), L(z) \) are FIR filters defined as:

\[
H(z) = \sum_{n=0}^{N} h(n)z^{-n} \quad L(z) = \sum_{n=0}^{N} l(n)z^{-n}
\]  

(A.17)

If the sampling operators \((\downarrow 2)\) and \((\uparrow 2)\) were not present, a reconstruction without delay would mean that:

\[
H(z)H'(z) + L(z)L'(z) = z^{-N}
\]  

(A.18)

So the filter reconstruct the input with a delay of \( z^{-N} \). In general a QMF structure with \( N^{th} \) order filters produce a delay of \( N \) samples. If the combination of \((\downarrow 2)\) followed by \((\uparrow 2)\) is applied to a signal, all odd-numbered components are replaced by zeros. It can be proven that up- and downsampling of a filtered signal \( X(z) \) in the \( z \) domain looks like

\[
(\uparrow 2)(\downarrow 2)H(z)X(z) \leftrightarrow \frac{1}{2}\left( H(z)X(z) + H(-z)X(-z) \right)
\]  

(A.19)

The second term \( H(-z)X(-z) \) causes aliasing. This term is multiplied by \( H'(z) \) at the reconstruction step. This alias has to cancel the alias form the other channel: \( L'(z)L(-z)X(-z) \). Combining the expressions, we obtain an alias cancellation condition:

\[
H'(z)H(-z) + L'(z)L(-z) = 0
\]  

(A.20)

The sampling operators also produce a change in (A.18). From (A.19) it can be proven that perfect reconstruction with down- and upsampling is possible when:

\[
H(z)H'(z) + L(z)L'(z) = 2z^{-N}
\]  

(A.21)

This design condition ensures distortion cancellation. These two design condition still give much freedom in the design. It can be proven that the QMF structure is orthogonal if the highpass and lowpass filters are coupled as

\[
H(-z) = z^{-N}L(z^{-1})
\]  

(A.22)
In words this means the filter is reversed and every second coefficient is multiplied by $-1$. In this way it is possible to construct a complementary highpass filter if the lowpass filter is known. Perfect reconstruction is obtained if the reconstruction filters equal:

$$L'(z) = H(-z) \quad H'(z) = -L(-z)$$  \hspace{1cm} (A.23)

In this way the reconstruction and analysis filters are equal, besides a flip of the coefficients. Filter banks using these filters are paraunitary. Wavelet filters are a special class of orthogonal filters and the design rules are complicated. The fact that they converge to regular waveforms (see subsection 3.2.2) is related with the placing of the zeros (FIR filters only have zeros and no poles) of the filters. Design rules of several wavelet filters are presented in [SN96]. Their filter order $N$ is always odd as a consequence of the design rules.

### A.5 Quantisers

A quantiser is defined as a device with input $x$ and output $x_q = Q(x)$, which maps input intervals $(\ldots, \langle \alpha_{i-1}, \alpha_i \rangle, \langle \alpha_i, \alpha_{i+1} \rangle, \ldots)$ on discrete output levels $(\beta_i)$. The quantiser error signal $e$ is given by $e = Q(x) - x$. The quantisers here discussed will be assumed to be uniform and infinite, so that the input signal is never clipped. In general the decision levels $(\alpha_i)$ and the representation levels $(\beta_i)$ are given by:

$$\forall i \left\{ \begin{array}{l} \alpha_i = (i + 0.5)\Delta + \kappa \\ \beta_i = i\Delta + \kappa \end{array} \right.$$  \hspace{1cm} (A.24)

where $\kappa$ is a constant which holds $-\frac{\Delta}{2} < \kappa \leq \frac{\Delta}{2}$. The quantiser stepsize is defined with $\Delta$, inversely related to the resolution. This holds that the quantizing error never exceeds $\frac{\Delta}{2}$. The output of the quantiser becomes:

$$Q(x) = \Delta \left\lfloor \frac{x - \kappa}{\Delta} + 0.5 \right\rfloor + \kappa$$  \hspace{1cm} (A.24)

The brackets $\lfloor \rfloor$ represent to floor operator. If $\kappa = 0$, the quantiser has a properly defined zero output level and is called a mid-tread quantiser. The quantiser is of the mid-riser type in the case that $\kappa = \frac{\Delta}{2}$.

![Fig. A.2: A linear mid-tread quantiser with unit stepsize ($\Delta = 1$, $\kappa = 0$).](image)
Appendix B

Some wavelet families

In this appendix the most common waveforms are presented. Waveforms in the first section are only used for analyzing purposes. They are characterized by a wavelet function in time domain only. The second group of wavelet families are designed for use in the DWT. They are characterized by the four QMF filters and posses a wavelet and a scaling function in time domain.

B.1 Continuous WT waveforms

The following waveforms, also called crude wavelets, are common in wavelet analysis. They are not orthogonal or biorthogonal compactly supporting but in theory they possess the perfect reconstruction property.

B.1.1 Morlet

The Morlet wavelet is possibly the original wavelet. Conceptually related to the STFT, the Morlet is a locally periodic wavetrain. It is obtained by taking a complex sine wave (as in the Fourier transform), and by localizing it with a Gaussian (bell-shaped) window, already presented in (2.16). An often used combination is $f_c = \frac{\pi}{2}, f_b = 2$, but these values can be modified. The Morlet wavelet can also be used as a real waveform providing only amplitude information.

![Morlet Wavelet](image)
B.1.2 Meyer

The Meyer wavelet is defined in the frequency domain. Contrary to the most waveforms for continuous WT analysis, the Meyer wavelet possesses a scaling function which is also defined in the frequency domain. The wavelet and scaling function can be truncated to obtain a FIR filter approximation which can be used in DWT. This discrete Meyer wavelet is close to perfect reconstruction.

B.1.3 Gaussian

The Gaussian-derivative family is built starting from the Gaussian function \( f(x) = C_p e^{-x^2} \). The constant \( C_p \) is such that the 2-norm of the \( p^{th} \) derivative of \( f(x) \) is equal to 1. Waveforms are obtained by taking derivatives of \( f(x) \):

\[
\psi_p(t) = \frac{d^p}{dx^p} f(x)
\]  

(B.1)

Just as the Morlet wavelet Gaussian waveforms can also be used as complex analysis functions. Wavelets are then derived from the complex Gaussian \( f(x) = C_p e^{-i\pi x^2} \). Another well known waveform is the Mexican hat, which is proportional to the second derivative of the Gaussian.
B.2 DWT waveforms

The next families are especially designed to be used in DWT. These wavelets are compactly supported and orthogonal or biorthogonal. They are characterized by a highpass and lowpass analysis and synthesis filter which can form a QMF structure. Each family is designed according to certain design rules that lead to the four filters. The filters can be constructed for several orders, but most families are restricted to lower orders. From these filters a wavelet and a scaling function can be derived using the structure from Fig. 3.14 in subsection 3.2.2. The waveforms may look not smooth at all, but note that the wavelet filters all have smooth passbands in the frequency domain. In general, for higher orders the functions become smoother in the time domain which makes them more suitable for analyzing real-life signals (at least in control engineering). For the most popular families the wavelet and scaling function are presented next.

B.2.1 Daubechies

This family is called after Ingrid Daubechies. She invented the compactly supported orthonormal wavelet which made wavelet analysis in discrete time practicable. The first order member of this family is also known as the Haar wavelet.
B.2.2 Coiflets

Coiflets were also build by Ingrid Daubechies at the request of Ronald Coifman. They are near from symmetry and have nice shapes.
B.2.3 Symlets

Symlets are nearly symmetrical wavelets proposed by Daubechies as modifications to the Daubechies family. The properties of the two wavelet families are similar.

![Waveform plots for sym2, sym3, sym4, sym5, sym6, sym7, sym8](image)

B.2.4 Biorthogonal

This family contains compactly supported biorthogonal spline wavelets. Their filters are symmetrical, which ensures symmetrical wavelets and scaling functions. This symmetry means that filters have linear phase characteristics which is useful (i.e. in avoiding dephasing in image compression). It is well known that symmetry and exact reconstruction are incompatible if the filters are paraunitary (equal for the analysis and the synthesis bank). The only exception on this is the first order 'Haar' or 'db1' wavelet, which is also the first order member of this family. Symmetry is only possible if the analysis and synthesis filters differ. It is possible that even the order of reconstruction and analysis filters is different. Therefore each member of this family has two wavelets and two scaling functions: a set for decomposition and one for reconstruction. There exists a reverse biorthogonal family which uses the synthesis filters for decomposition and vise versa. The double numbering stand for the order of respectively the analysis filter and the synthesis filter. Wavelets 4.4, 5.5 and 6.8 are such that reconstruction and decomposition functions and filters are close in value.
Some wavelet families

Analysis

Synthesis
B.2 DWT waveforms

- bior3.1 $\psi(t)$
- bior3.1 $\phi(t)$
- bior3.1 $\psi(t)$
- bior3.1 $\phi(t)$

- bior3.3 $\psi(t)$
- bior3.3 $\phi(t)$
- bior3.3 $\psi(t)$
- bior3.3 $\phi(t)$

- bior3.5 $\psi(t)$
- bior3.5 $\phi(t)$
- bior3.5 $\psi(t)$
- bior3.5 $\phi(t)$

- bior3.7 $\psi(t)$
- bior3.7 $\phi(t)$
- bior3.7 $\psi(t)$
- bior3.7 $\phi(t)$

- bior3.9 $\psi(t)$
- bior3.9 $\phi(t)$
- bior3.9 $\psi(t)$
- bior3.9 $\phi(t)$

- bior4.4 $\psi(t)$
- bior4.4 $\phi(t)$
- bior4.4 $\psi(t)$
- bior4.4 $\phi(t)$
Some wavelet families
Appendix C

MATLAB functions

C.1 Real-time wavelet filters: rtwavfilt.m

% RTWAVFILT makes it possible to use the discrete
% wavelet transform in real time.
%
%
% This "wavelet filter" is composed in
% discrete state-space form and uses the matrices
% A, B and C to build the system.
%
% The state consists of old data points (inputs)
% and coefficients needed for reconstruction.
%
% CI contains the indices of the most recent
% coefficients at each level in the state-vector.
% They are ordered from low to high and the last
% indices refer to the approximation coefficient.
%
% DL gives the total number of unit delays of the
% filter. For MODE 3 this is a vector containing the
% delays for each decomposition level.
%
% The waveform can be specified by the 4 QMF filters:
%
% L: the decomposition low-pass filter
% H: the decomposition high-pass filter
% LR: the reconstruction low-pass filter
% HR: the reconstruction high-pass filter
%
% P is the desired decomposition level.
%
% MODE determines the output of the system:
%
% MODE 1: reconstruction of the input.
%
% MODE 2: separate reconstruction of all decomposition
% levels. Equal delay times for all levels
% (direct signal reconstruction possible).
% MODE 3: MODE2, but with minimal delay for each level.
% For MODE 2 and 3, the output order is the same as
% the order in the state-vector:
% y(1):    level 1 reconstruction
%         :
% y(end-1): level P reconstruction
% y(end):  reconstruction of the approximation
% %
% Maurice Schneiders
% August 2001
% %
% force into row-vectors
% L=L(:)’; H=H(:)’; Lr=Lr(:)’; Hr=Hr(:)’;
% Filter order
N=length(L)-1;
% l_u=0; l_d=0; llu=[]; lld=[]; for p=1:P
   l_u=l_u+N*2ˆ(p-1);
   l_d=l_d+(N-1)*2ˆ(p-1);
   llu=[llu l_u];
   lld=[lld l_d];
end
% l_u: number of old inputs u needed
% l_d: delay (number of samples)
% A=zeros(l_u+(l_d+1)*(P+1),l_u+(l_d+1)*(P+1)); B=zeros(l_u+(l_d+1)*(P+1),1);
% B(1)=1; C=zeros(1,l_u+(l_d+1)*(P+1));
% Shift all old inputs
A(2:l_u,1:l_u-1)=eye(l_u-1,l_u-1);
% Shift all old coefficients
for i=0:P
   A(l_u+2+i*(l_d+1):l_u+2+i*(l_d+1)+l_d-1,...
      l_u+1+i*(l_d+1):l_u+2+i*(l_d+1)+l_d-2)=eye(l_d,l_d);
end
% Computing level 1 highpass coefficient
A(l_u+1,1:N)=H(2:end); B(l_u+1)=H(1); T=zeros(1,N); T(1:2:end)=Hr(2:2:end);
C(l_u+2+l_d-N:1_u+1+l_d)=T;
if P==1
% Computing level P lowpass coefficients
   A(l_u+2+l_d,1:l_u)=L(2:end);
   B(l_u+2+l_d)=L(1);
   T=zeros(1,N);
   T(1:2:end)=Lr(2:2:end);
   C(end-N+1:end)=T;
else
% Computing other coefficients
% build biggest matrix needed

sa=llu+1;
Al=zeros(sa(end-1),sa(end));
for k=1:sa(end-1)
    Al(k,k*2-1:k*2-1+N)=L;
end

% analysis coefficients
% computing highpass coefficients LH,LLH,...
% and computing lowpass coefficients L^N

HL=H;
LL=L;
for p=2:P
    HL=HL*Al(1:sa(p-1),1:sa(p));
    LL=LL*Al(1:sa(p-1),1:sa(p));
    A(l_u+1+(p-1)*(l_d+1),1:llu(p))=HL(2:end);
    B(l_u+1+(p-1)*(l_d+1),1)=HL(1);
    A(l_u+1+P*(l_d+1),1:llu(p))=LL(2:end);
    B(l_u+1+P*(l_d+1),1)=LL(1);
end

% synthesis coefficients
% highpass
ss=(lld+2)/2;
HLr=Lr(2:2:end);
for p=2:P
    THr=zeros(ss(p-1),ss(p));
    TLr=zeros(ss(p-1),ss(p));
    for k=1:ss(p-1)
        THr(k,k:2^(p-1):end-ss(p-1)+k)=Hr(2:2:end);
        TLr(k,k:2^(p-1):end-ss(p-1)+k)=Lr(2:2:end);
    end
    T=zeros(1,l_d+1);
    T(end-2*ss(p)+2:2:end)=HLr*THr;
    C(l_u+1+(p-1)*(l_d+1):l_u+1+(p-1)*(l_d+1)+l_d)=T;
    HLr=HLr*TLr;
end

% lowpass
T=zeros(1,l_d+1);
T(1:2:end)=HLr;
C(l_u+1+P*(l_d+1):end)=T;
end

d=[lld lld(end)]; l=llu(end);
ci=zeros(P+1,1); for k=0:P ci(k+1)=(l+k*(d(end)+1)+1); end
if exist('mode')~=1
    mode=1;
end
Cms=zeros(P+1,1);
if mode==1
    dl=d(end);
elseif mode==2
    for p=1:P+1

C.2 Real-time wavelet analysis for DWT-wavelets: rtwavana.m

function F=rtwavana(L,H,P)
% RTWAVANA makes it possible to use wavelet
% analysis in real time with wavelet which are
% defined by their QMF filters (used for DWT).
% % F=RTWAVANA(L,H,P)
% % % This "wavelet analyzer" is composed in a
% % filter matrix F. The rows of F contain of
% % scaled waveforms defined by the 2 decomposition
% % filters of the wavelet (L,H). P defines the
% % decomposition depth (also the number of outputs)
% % % There is minimal delay for each scale.
% % Actually F can be seen as a
% % set of FIR filter and can be used in a
% % Simulink "discrete filter block" as numerator
% % with the denominator set to 1.
% % %
% % Maurice Schneiders
% % August 2001

L=L(:)'; H=H(:)';
% Filter order
N=length(L)-1;

l_u=0; llu=[]; for p=1:P
    l_u=l_u+N*2^(p-1);
    llu=[llu l_u];
end
sa=llu+1; F=zeros(P,sa(end)); Al=zeros(sa(end-1),sa(end)); for k=1:sa(end-1)
    Al(k,k*2-1:k*2-1+N)=L;
end
F(1,1:N+1)=H(1:end);

HL=H; LL=L; for p=2:P
    HL=HL*Al(1:sa(p-1),1:sa(p));
    LL=LL*Al(1:sa(p-1),1:sa(p));

    F(2,1:sa(p))=HL(1:end);
end