Time series analysis in coastal engineering

Literature study

T.J.P. de Rooij

Delft University of Technology, Deltares

TU Delft

Deltares

Time series analysis in coastal engineering

Literature study

by

T.J.P. de Rooij

a literature study in preparation of the thesis
to obtain the degree of Master of Science
at the Delft University of Technology.

April 25, 2017

Student number:	4077903	
Project duration:	February 1, 2017 – October 31, 2017	
Thesis committee:	Prof. dr. ir. C. Vuik,	TU Delft, supervisor
	Ir. J. Kramer,	Deltares

TU Delft Delft University of Technology

Deltares Enabling Delta Life

Contents

1 Introduction

When was the last time you took a picture with your mobile phone? Did you realize you used wavelets to store this picture efficiently? *Excuse me, wavelets?* Yes wavelets! Wavelets are short wavelike functions which are used to analyze, decompose and compress signals [18]. This technique is called wavelet analysis and finds its base in the late 1980's. The impact of this technique is so large that its main founder, Yves Meyer, won the Abel prize[1] just a few weeks ago [3]. An everyday application of wavelet analysis is the compression of pictures using the JPG-format [18, 27] which has been the standard format to save pictures efficiently since 1992, even for the newest mobile phone today. Other examples of applications are the analysis of electric fields produced by lightning [6], electron spin resonance analysis [35], speech enhancement [19], digital fingerprint compression by the FBI [27] and the analysis of seismic signals for detecting e.g. earthquakes and oil layers [39]. These are just a few examples to highlight the large applicability of wavelets.

Wavelet analysis has thus found its way into many research areas. However, in the field of coastal engineering wavelets are virtually unexplored. Deltares is a Dutch independent institute for applied research in the field of water and subsurface[2] that has marked this technique as an opportunity to improve time series analysis in the coastal engineering field. Coastal researchers and engineers have to deal with complex time dependent physical processes every day. At Deltares the main focus of research lies within areas where land and water meet, such as deltas, coastal regions and river basins. Sea level rise threatens economically important and densely populated areas all over the world, therefore this research is indispensable. Currently, time series analysis in the field of coastal engineering is mostly performed through Fourier analysis, in combination with noise reducing techniques. Fourier analysis however has its limitations, especially concerning non stationary signal analysis [15]. This literature study will focus on the analysis of time series in the coastal engineering field. These signals contain time evolution of wave heights, forces, pressures, etc. The goal of Deltares' assignment (see Appendix D) is to research the added value of applying wavelet analysis using existing wavelets and related techniques to time series, compared to Fourier analysis.

Overview of the thesis

This has been the starting point of this literature thesis, which has been conducted at Deltares in the department Coastal Structures and Waves (CSW). After addressing the preliminaries and notation at the end of this chapter, the second chapter will give a short problem description, with a set of example signals. In Chapter 3, the view of time series analysis is broadened: what ways are there to analyze signals and how do they relate? In this chapter three techniques are highlighted which will be addressed in Chapter 4, 5 and 6. In these chapters also the advantages and disadvantages of the different methods are discussed. In Chapter 6 extra attention is paid to wavelet analysis, for it is the technique that still has to be explored. The applicability and comparison of these different techniques in a set of signal processing applications is addressed in Chapter 7. Finally, this thesis concludes with a summary and a conclusion in Chapter 8. In this conclusion the main research question of my masters thesis: "How can wavelet analysis improve time series analysis in the field of coastal engineering?", will be presented. Following this main question some research question are derived.

[1]The Abel prize is considered to be the Nobel prize in mathematics.
[2]Website Deltares: `www.deltares.nl`.

1.1. Preliminaries and notation

In this section preliminaries and notation used in this literature thesis are shortly addressed. In the layout of this thesis, two different blocks can be found.

Example 1.1 Examples are shown in grey boxes; this is an example of an example.

Extra information, such as used code, can be found at the bottom.

And some wavelets are introduced in blue boxes.

Mathematical notation

Functions (of time) or signals (in the time domain) will be given with a lowercase letter, e.g. $s(t)$ or $x(t)$. When discrete signals are addressed, the function is given with hooked brackets, e.g. $x[n]$. Signals the other discussed domains, Fourier and wavelet, will be given with capitals: respectively $\mathcal{F}\{x\}(\omega) = X(\omega)$ and $\mathcal{W}\{x\}(a,b) = X(a,b)$. Furthermore, matrices will also be expressed with a capital. Here the ω represents the frequency, a the scale and b the translation. The Greek letter ϕ is used for refinable functions only, and ψ only for wavelets only. Their Fourier transforms are given by their capitals, respectively Ψ and Φ. Vectors are given in bold type $\boldsymbol{v} = [v_1, v_2, \ldots, v_N]^\mathsf{T}$.

In both mathematics and computer science, the order symbol $\mathcal{O}$, also known as the Landau O symbol is used. The letter O is used because it refers to *order*. The mathematical notation describes the limiting behavior of a function. In this literature thesis and computer science it is used to express the complexity of algorithms. It expresses the number of floating point operations (flops) for the parameter going to infinity.

Mathematics

Some complex calculations are stated throughout this thesis, therefore some complex calculus will be shortly revised. The imaginary number i is defined as $i^2 = -1$. A complex number $z = x + iy$ consists of a real part $\mathrm{Re}(z) = x$ and an imaginary part $\mathrm{Im}(z) = y$. The modulus $|z| = \sqrt{x^2+y^2}$ and the argument $\theta = \arg(z) = \arctan(y/x)$, such that the complex number can be written in polar form as $z = |z|e^{i\theta}$. The complex conjugate $\overline{z} = \mathrm{Re}(z) - i\mathrm{Im}(z)$. The conjugate transpose of a matrix A is denoted as $A^* = \overline{A^\mathsf{T}}$, so $(A^*)_{ij} = \overline{A_{ji}}$.

The inner product for two functions $f, g \in L^2$ is defined as

$$\langle f, g\rangle = \int_{-\infty}^{\infty} f(t)\overline{g(t)}\,\mathrm{d}t. \tag{1.1}$$

The block function $\mathbb{1}_I(x)$ is given by

$$\mathbb{1}_I(x) = \begin{cases} 1 & \text{if } x \in I \\ 0 & \text{elsewhere} \end{cases}. \tag{1.2}$$

Furthermore some quantities will be expressed in decibels, which are defined as

$$x_{\mathrm{dB}} = 10 \cdot \log_{10} x_{\mathrm{magnitude}}.$$

Power decibels are common in signal analysis, those are defined by

$$x_{\mathrm{powerdB}} = 10 \cdot \log_{10} x^2_{\mathrm{magnitude}} = 20 \cdot \log_{10} x_{\mathrm{magnitude}}.$$

Discrete operations

In Chapter 6 some algorithms will be discussed, which rely on the following discrete operations. If $\boldsymbol{x}$ is a sequence $\boldsymbol{x} = \{\ldots, x_{-1}, x_0, x_1, \ldots\}$, the notation $(-)\boldsymbol{x}$ denotes the reverse sequence:

$$(-)\boldsymbol{x} = \{\ldots, x_1, x_0, x_{-1}, \ldots\} \tag{1.3}$$

The notation $(\downarrow 2)\boldsymbol{x}$ stands for down sampling of the sequence, an operation that throws the odd-numbered vector entries out and renumbers the even-numbered ones, so

$$((\downarrow 2)\boldsymbol{x}) = x_{2k}. \tag{1.4}$$

The opposite operation is upsampling, which is denoted as $(\uparrow 2)\boldsymbol{x}$, where between each pair of adjacent entries of the vector a zero is added, and the new vector is renumbered:

$$((\uparrow 2)\boldsymbol{x})_{2k} = x_k, \tag{1.5}$$

$$((\uparrow 2)\boldsymbol{x})_{2k+1} = 0. \tag{1.6}$$

Programming languages

In this thesis two programming languages are addressed. The MATLAB programming language by The MathWorks, Inc. will be referred to as Matlab. The MathWorks, Inc. also delivers a tool to apply wavelet analysis with: Wavelet Toolbox™. To this will be referred to as the Wavelet Toolbox. The other language is Python™ by the Python Software Foundation, simply referred to as Python. No commercial toolboxes using the Python programming language are used.

2
Problem description

Researchers in the field of coastal engineering want to improve the understanding of physical processes associated with waves, wave structure interaction, stability of structures or the influence on morphology recorded by various measurements techniques like time sampling, lasers scanning and photography. These techniques are employed to capture instant information on wave conditions, forces, currents, erosion and accretion for further detailed analysis. In this chapter an overview of the challenges for and questions about time series analysis concerning these measurements is given. At the end of this literature thesis some of these will be answered or used as basis of research questions for the masters thesis. The measurements are conducted at the Deltares facilities or in the field. Sometimes time series are the result of computer models.

Deltares' main testing facilities are the Zoutzoethal, the Geohal and the Delta Flume. In the Zoutzoethal scale models of coastal structures such as harbors, dikes, windmill foundations, shipping locks are tested. In the Delta Flume full size testing for some structures is possible. Tests in the Geohal are of a more geological nature or concern land structures, which are common not within the field of the department coastal structures and waves (CSW). The experiments result in continuous measurable quantities which are sampled and thus discretized in order to save the results. Results from computer models already are discrete. This limits the scope of this literature thesis to digital signals. A selection of used measurement equipment consists of cameras, laser distance meters, pressure sensors, force meters, capacitive water height meters etc. The research goals of conducted measurements are very broad. The challenge is to determine underlying signal in the sometimes noisy measurements. This can vary from determining wave parameters such as speed, direction, period, height, frequency etc. to the determination of peak values of forces or pressures. In the next section an example will be discussed. Often encountered problems by data analysts at Deltares are:

- Different analysts use different analyzing methods. Therefore interpretation of results may slightly differ per analyst.
- Some measured signals have large noise contributions, how can these be filtered out without affecting the underlying signal?
- What are the effects on the measured signals by known disturbances: e.g. resonance frequency and power supply frequency (50 Hz)?
- Can such known of expected effects be quantified and filtered from the signal?
- How to identify different types of noise, such as noise contributed by the measurement equipment and unwanted effects in the measured signal?
- Determine the timing of jumps or discontinuities and reconstruct them.
- The separation of incoming and reflected waves.

Some of these problems are described briefly in the following example section.

2.1. Schelde Flume measurement

There are a lot of different set-ups at the Deltares facilities. In this section one example of a setup in the Schelde Flume, a flume in which waves with a height of half a meter can be generated, is discussed. The measurements measure three different quantities at a sample rate of 3 kHz:

- The GHM (Wave Height Meter) is a self build piece of equipment which measures the conduction between an anode and a cathode, to determine the water (or wave) height. The amplifying filters in the GHM have cut-off frequencies of 85 Hz and 130 Hz.
- The pressure sensors are made from Wien bridges build with pressure sensitive resistors. Filters applied to the output have a cut-off frequency about 1000 Hz.
- The force sensors do more or less have the same construction as the pressure sensors.

All equipment is connected to one analog-to-digital converter (ADC) which is connected to the computer. The output of the pressure and force sensors is amplified by 30 year old amplifiers (KWS), which do not have any filter implemented to prevent delays between the different signals. A measurement set-up can have over 30 different sensors; for the analysis it is important that the alignment in time is as close to perfect as possible. An overview is given in Figure 2.1. Measurements start 30 seconds before the experiment starts, to check whether the measurement equipment is working. This can also be useful in determining off-set and noise contributions. The duration of a measurement can be over half an hour, which leads to about 6 million samples.

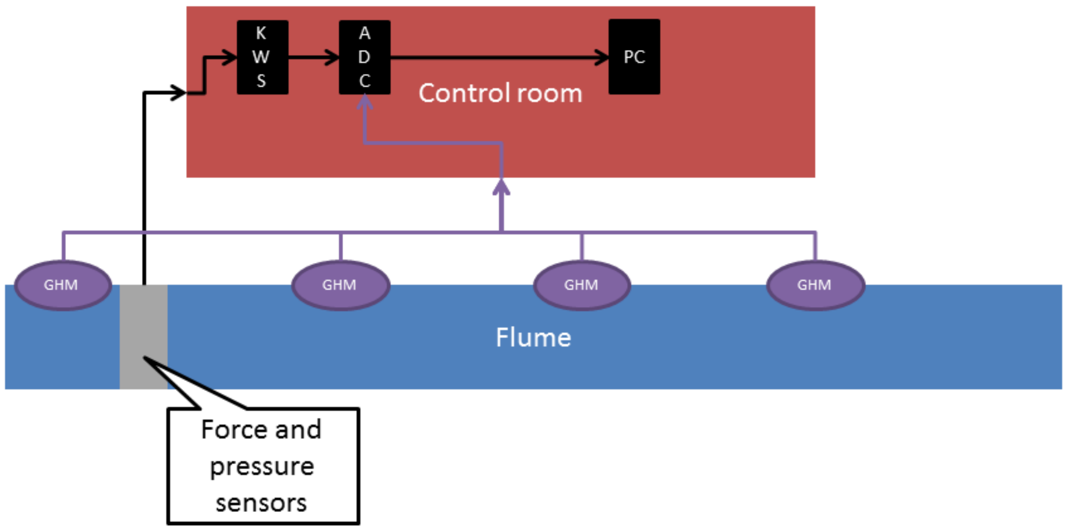


Figure 2.1: Measurement setup in the Schelde Flume

In Appendix C a set of time signals from the measurements are given in Figure C.1. At the end of the flume a breakwater is installed, behind this breakwater a transshipment box is placed to catch the water going over the breakwater. The effects of the waves on the breakwater are measured using force and pressure sensors. The water height measurement in the Figure C.1 displays the amount of water in the box. Both the pressure and the force measurement from Figure C.1 are placed in the frontal area of the wall. The peaks in their signals denote the arrival of a wave.

The spectra of the signals are given in Figure C.2. Although the time series of the force and pressure look a bit noisy, the spectrum shows little presence. For just 30 seconds of the signal (Figure C.3) the pressure and force signals still have a noisy appearance. If one looks at the water heights in the flume in Figure C.4, the water level is much less noisy than the pressure and force measurement. The spectra know large contributions of low frequency components. This is expected for these measurements. Further we notice that the noise in the water height measurements has a much smaller signal to noise ratio than the other two shown measurements.

All spectra (Figure C.2) show a clear peak around 50 Hz. This peak was expected and is the result of the power supply frequency in the Netherlands: 50 Hz. Furthermore the spectra show some mirrors at $(100\ell + 50)$ Hz for $\ell \in \mathbb{N}$. The force meters also shows peaks around 100ℓ Hz. In the measured force signal these mirrors are much more present than in the pressure and wave height measurement. The contribution of this distortion to the wave height measurement can be neglected on basis of this data. The source of these mirrors can for instance be the ACDC converter (alternating current to direct current converter) of the measurement equipment.

The different data sets have different problem formulations. For the water height in the box, the question is: how can we apply signal analysis to extract only the jumps in the signal? The goal is to determine the amount of water transshipped, which is hard to read from the time signal by the sloshing of the water. In the case of the force and pressure measurements the peak value of the underlying signal is an important design parameter. This is clearly influenced by some noise. What filter do we need to apply to suppress this noise, but not affect the underlying 'real' signal? In this literature thesis the possibilities within the large area of signal analysis will be explored. Starting with the overview of the area in the following chapter.

3 Analysis of signals

In the last chapter a problem description is given: time series which result from measurements at Deltares are distorted by all kinds of effects. The question for this literature thesis is to give an overview of the possible ways for the analysis of these signals. In this chapter different 'observation options' of temporal information are presented. These "data types" could also be applied to spatial information, this will not be considered. The goal is to find an easy retrieval of both temporal and frequency information of a signal $x : \mathbb{R} \to \mathbb{C}$. An example comprehensible for readers with different backgrounds will be used here: a guitar player, striking only just three strings separately.

Time domain

The receiver of a time signal, for instance an ear receiving sound waves, a cellphone receiving the 4G signals or laser equipment measuring wave heights at Deltares, always receives an amplitude at a given time. Mathematically this is seen as a function: $x(t)$, mostly being referred to as a (time) signal in signal analysis. From this signal a lot of properties of the transmitter, transmission and receiver can be derived using the right techniques. In Section 3.3 the characteristics retrievable from this time signal $x(t)$ will be discussed. However, in transmission the signal $x(t)$ can be disturbed, making it hard to draw the right conclusions based on the time signal. An example of a guitar player recording can be found in Figure 3.1c.

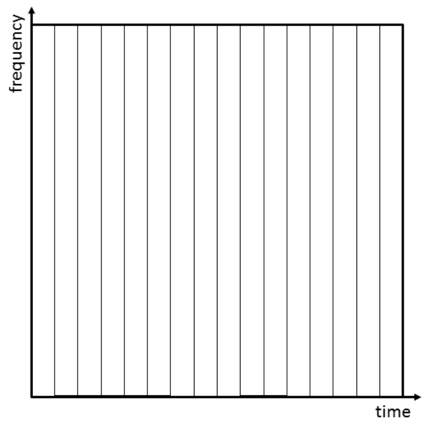
Fourier transform

The signal $x(t)$ does contain more information then just the amplitude at a given time. With the right mathematical transformations, an insight in the different frequencies of the signal can be given. The *Fourier transform* (FT) is a common and much used transformation to derive the energy density per frequency of a signal. This transformation represents the signal in the frequency domain and will be discussed in Chapter 4.

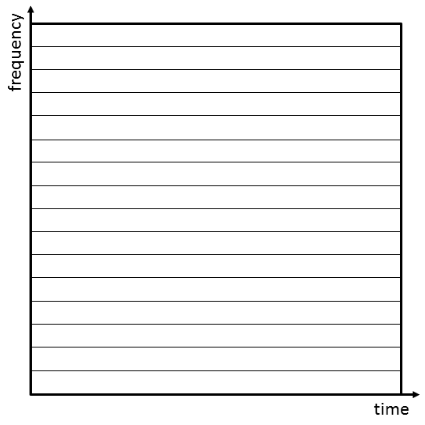
Think of the guitar player again. If only the time domain signal of the sound is considered, one cannot know which notes were played. However we do know at what times the strings were struck, by observing the amplitude differences in time. If only the Fourier transform of the signal is considered, the representation of the signal in the frequency domain, one can tell exactly which notes were played, but not in which order. This can be observed in the spectrum in Figure 3.1d.

Short term Fourier transform

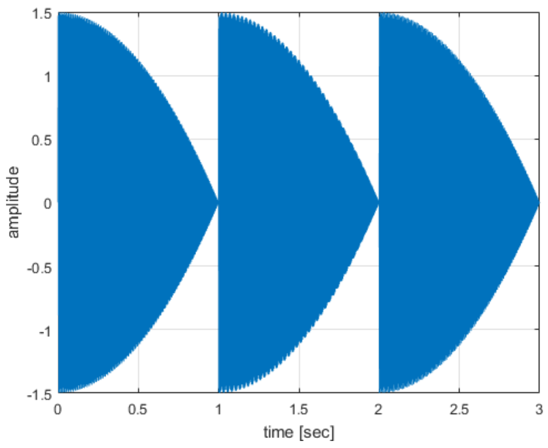
A (practiced) listener however can tell both the notes and the order in which these notes were played! This is impossible when only the time domain or in the frequency domain of the signal is reviewed. A visual representation of these short comings of these domains is shown in Figure 3.1. In the early twentieth century Gabor [4] was the first to act on these short comings, developing the Gabor transform. He used the combination of a window function and the Fourier transform to derive a coupling of the temporal and frequency domain. Hence, this method gives insight in the occurrence of frequencies in distinct time intervals. Later his work was placed in the framework of the *short term Fourier transform* (STFT, Figure 3.1e) which will be covered in Chapter 5. A disadvantage of this method is the relative large loss of both temporal and frequency information, which is explicable from the uncertainty principle (Section 5.3.1). This can be observed in Figure 3.1g.



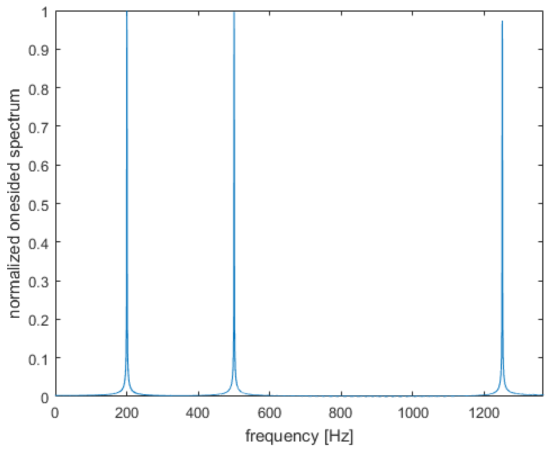
(a) Time domain: a high resolution of the energy distribution over the time axis, no resolution over the frequency axis.



(b) Frequency domain: opposite of the time domain: high resolution over frequency axis, no resolution over the time axis.



(c) Three notes played separately by a guitar player.



(d) The spectrum of the notes, showing us that the played notes are at 200, 500 and 1250 Hz. The spectrum does not provide any information about the timing of these notes.

Figure 3.1: Schematic view of the four main ways to analyze a signal, including examples.

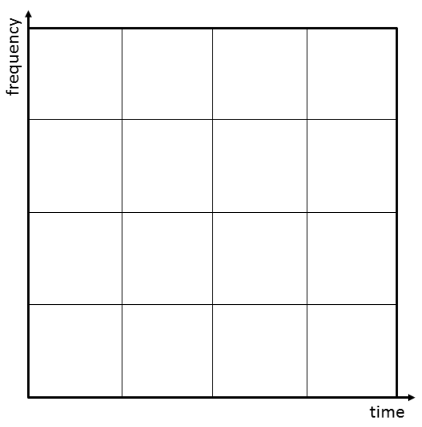
Wavelet analysis

These losses are minimized by the application of wavelet analysis. In the 1960's the basis of the wavelet transform was developed in both mathematics and physics. Wavelet analysis is a tool that cuts the function $x(t)$ up into different frequency components, studying each component with a resolution matching its scale [11], alike the STFT. In the late 1980's the approaches from different disciplines were combined, which led to the theory of wavelet analysis as known today. As shown in Figure 3.1f, for different frequencies the time spacings are different, this again finds its explanation in the uncertainty principle. This results in a different representation as shown in Figure 3.1h.

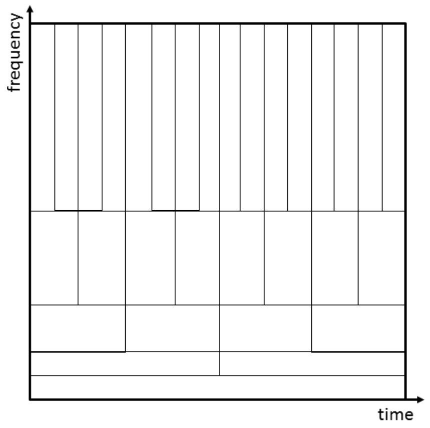
Now note that wavelet analysis approach resembles the human interpretation of sounds fairly good: high pitches are often played short and recognized quickly, where low pitches are usually played longer. In fact, Daubechies [11] shows the human ear uses a wavelet analysis type of signal processing in the early stages of analyzing sounds. This makes wavelet analysis a great basis for compression of sounds, undetectable to the human ear. This is one of the many examples the wavelet transform is very useful for.

3.1. Analysis and synthesis

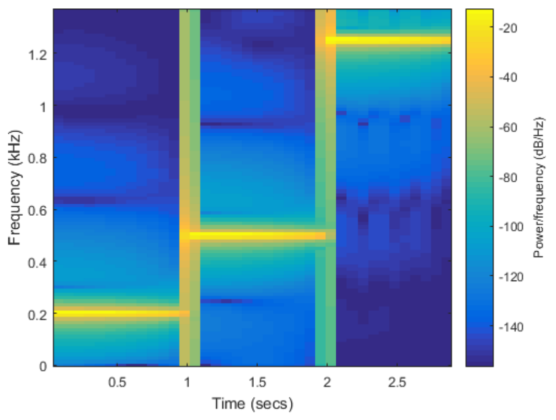
An important theme in analysis is the representation (or approximation) of a known or unknown function (or time signal) $x(t)$ by special known functions [4]. This representation is the basis of Fourier analysis, which in his term is the basis of the STFT and the wavelet transform. These special functions are chosen from a family of functions, for instance monomials: $t \mapsto t^k$, $k \in \mathbb{N}$. These functions are well understood and preferably



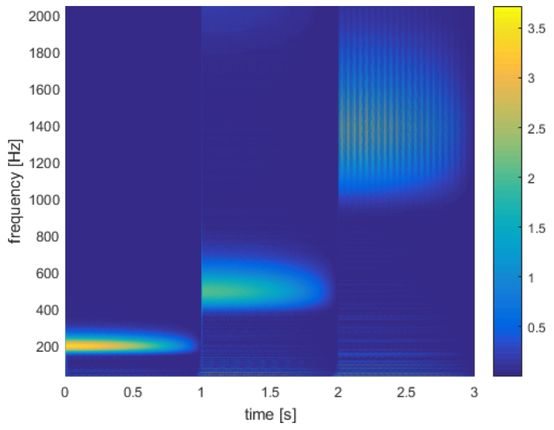
(e) **Short Term Fourier Transform (STFT)**: trade off in time and frequency, less resolution in both time and frequency direction, but linking the two domains.



(f) **Wavelet transform**: same trade off as of the STFT, different implementation. Note the areas of the blocks are the same, only their frequency-time spacing differs.



(g) The spectrogram of the signal from (c): the timing of the notes can be read.



(h) The scalogram of the signal from (c): the timing of the notes can be read.

Figure 3.1: Schematic view of the four main ways to analyze a signal, including examples.

easy to compute. Most often their analytical properties tend to show the evident or hidden symmetries of the considered function $x(t)$.

In this literature thesis the function

$$x : \mathbb{R} \to \mathbb{C}$$

is considered. Often this function is referred to as *(time) signal*, which is a broader concept of *a function or sequence that represents information* [15]. Further a family of so-called *basis functions* is chosen. The family given by $\{e_\alpha | \alpha \in I\}$ with the basis functions $t \mapsto e_\alpha(t)$. The index set I can be discrete (as in the example of the monomials), but also continuous. An arbitrary function $x(t)$ then can be represented in the form

$$x(t) = \sum_{\alpha \in I} c_\alpha e_\alpha(t) \tag{3.1}$$

in the discrete case. This expression (3.1) is called a *representation*. In the continuous case it can be expressed as an integral:

$$x(t) = \int_I c(\alpha) e_\alpha(t)\, \mathrm{d}\alpha.$$

Here the c_α is the coefficient belonging to the basis function $e_\alpha(t)$. The set of coefficients $\{c_\alpha | \alpha \in I\}$ is called the *analysis* of x with respect to the family $\{e_\alpha | \alpha \in I\}$. The inverse operation that takes a given coefficients as input and returns the function itself as output is called the *synthesis* of x by means of e_α. There are tons of different families of basis functions to choose from. The Taylor polynomial for instance uses higher and higher derivatives of the function $x(t)$. In this literature thesis two much used different families will be discussed. These both share the property of orthonormality, which makes the coefficients relatively easy to compute [4]. These two are the Fourier analysis (Chapter 4) and the wavelet analysis (Chapter 6).

3.2. Signals and systems

In the last section a signal was explained as function or sequence that represents information. The goal of signal analysis is to find information unknown to the receiver. The applications as mentioned in Chapter 2 are all measurement results, done using a computer. $x(t)$ therefore will not be a continuous signal of function, but a *discrete* signal $x[n]$, often in the form of a vector $\mathbf{x} = \{x[n]\}_{n=0,\dots,N}$. For the numerical work discretization is indispensable. On the one hand a computer can only do a finite amount of computations and so handle a finite amount of basis functions. On the other hand the time signal $x[n]$ will be discrete and finite. In the following chapters always both the continuous and the discrete cases will be reviewed. The focus however will be on the discrete cases, for the scope of this literature thesis are mainly *digital* signals, discrete in both time and amplitude. Signals are classified *deterministic* if their behavior is known and can be represented by e.g. a formula. *Stochastic* signals are discussed further in Chapter 7: their amplitudes cannot be defined by formulas or graphical elements. These signals are most described in terms of their expected values (mean, variance etc.).

So, there are a lot of different characteristics on signals. We have learned that signals represent information. So if for instance we have a signal of a measurement of the force on a monopile hit by a single wave. The input (the wave) and the output (the measurement) together contain information about the monopile, e.g. its resonance frequency. The monopile in this example is a system, where the formal definition of a system is: *the abstraction of a process or object that puts a number of signals into some relationship* [15]. The study of these systems is called *System Theory*. This study is complete for so called *linear* and *time-invariant* systems. The different analyzing methods are very good applicable to these type of systems.

Linear

The response of a system is the description of the output for a certain input. This response can be written in the form of a function. A function $f(\boldsymbol{x})$ is called *linear* if the *superposition principle* is applicable [15]:

$$f(A\boldsymbol{x} + B\boldsymbol{y}) = Af(\boldsymbol{x}) + Bf(\boldsymbol{y}).$$

Often the linearity of a system is assumed for simplification, although it is not linear.

Time invariance

A second important characteristic of a system is known as *time invariance*. This holds that *the response to a delayed input signal results in a corresponding delayed output signal* [15]. Systems that are both linear and time invariant are abbreviated to *LTI* systems. There are three techniques to model continuous time systems. The mathematical representation is in the form of a set of differential equations to relate the output to the input. Block diagrams and state models are both graphical representations of that relationship.

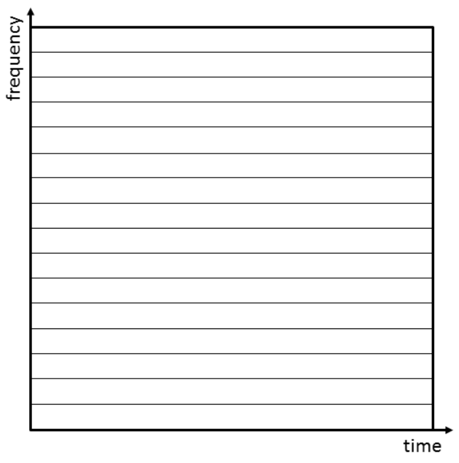
3.3. Time series analysis

Why undertake the hassle of transforming a signal, when we can also derive a lot of information from the time domain? A lot of information in the time domain is very important. One can think of minimums and maximums, but also of maximum derivatives, mean load etc. Often this information is wanted, however it might be distorted by effects that are hard to recognize in the time domain. For instance think of the power supply effects shown in Chapter 2. There are means to address these issues in the time domain, as will be elaborated further in Section 7.2.1. These are however applicable in a small set of situations. The feasibility of these means is discussed in the same section as well. In this literature thesis, *deterministic signals* are considered: deterministic signal can be modeled completely as a function of a variable. From Section 7.1 *random signals*, such as the measurement results, are discussed.

3.3.1. Other analyzing techniques

Of course there are much more analyzing techniques than just the few discussed here. Four important other techniques are briefly discussed here, starting with the Laplace transform, which can be seen as a predecessor of Fourier analysis [15]. The Laplace transform is still being used in a lot of engineering fields. This transform is also only suitable for frequency domain representation of a signal. The second one is the z-transform, which is more of a discrete implementation of the Fourier transform. It is not named after a famous mathematician, but after the letter z, which is used for the complex frequency variable. The z-transform is important in digital Fourier analysis and filter design in this field. The third one is the Hilbert transform [15], which will be shortly addressed in its combination of the Hilbert-Huang transform (Section 7.2.3). Finally the Wigner-Ville distribution cannot be forgotten. This is also an time-frequency transform, much alike the short term Fourier transform. It is a technique finding its base in physics from the 1930s. Before the introduction of wavelet analysis it has found its application is a lot of different sectors, however it suffers greatly from cross terms and is not used much for this reason [32]. In the next chapter the focus will be on the mathematical background and practical applicability of Fourier analysis.

4 Fourier Analysis



In the formal Fourier analysis, the basis functions as described in Chapter 3

$$t \mapsto e^{ikt} = \cos(kt) + i\sin(kt), \qquad k \in \mathbb{Z} \tag{4.1}$$

are chosen. This is a family of 2π-periodic, orthonormal functions. For any $f \in L^2(\mathbb{R}/2\pi)$, measurable functions $f : \mathbb{R} \to \mathbb{C}$, the formal definition of the Fourier transform is built. This theory expanded to cover not only these specific functions, but all functions on $\mathbb{R}$. The resulting sum, known as the representation (3.1), is called the *Fourier series*:

$$f(t) := \sum_{k=-\infty}^{\infty} c_k e^{ikt}. \tag{4.2}$$

In this chapter the analysis using this series will be clarified. The next chapter will expand the Fourier Analysis technique to the Short Term Fourier Transform (STFT). That chapter will conclude with the limitations, advantages and disadvantages of the use of the Fourier Analysis.

4.1. CFT - Continuous Fourier Transform

The Fourier transform (abbreviated to FT, or CFT where the C stands for continuous) of a time signal, or function, $x \in L^1$ is defined by the integral

$$\mathcal{F}\{x(t)\}(\omega) = \mathcal{F}x(\omega) = X(\omega) := \int_{-\infty}^{\infty} x(t)e^{-i\omega t}\,\mathrm{d}t, \qquad \omega \in \mathbb{R}.$$

Here ω denotes the frequency in Hz (s^{-1}). This transform gives a representation in the complex domain of the frequency content of the time signal $x(t)$. Often the modulus of the Fourier transform is displayed against the frequency ω to indicate the energy density over the spectrum. As discussed before, information concerning time-localization cannot be read off from $X(\omega)$. So it is hard to determine at what time, which frequencies are present in the signal. However when one is looking out for one frequency, like radars which respond to very specific frequencies, this characteristic of the Fourier transform is very convenient.

$X(\omega)$ is called the two-sided spectrum of $x(t)$. The original time signal can be calculated form this spectrum by using the inverse Fourier transform

$$x(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(\omega)e^{i\omega t}\,\mathrm{d}\omega,$$

the linear combination of the pure oscillations of all frequencies $\omega \in \mathbb{R}$. A few important Fourier transform theorems can be found in Appendix A. Note that for a periodic, continuous time signal the CFT is discrete in frequencies, for the CFT exists of a finite set of basis functions. But for a general signal it is nor discrete nor periodic.

Example 4.1 (Spectrum) The CFT of a rectangle pulse is a nice example to show the continuous Fourier transform. The rectangle pulse is defined as

$$\text{rect}(at) = \mathbf{1}_{[-a/2,a/2]}, \qquad \text{for } a > 0.$$

To determine the Fourier transform for the case $a = 1$, we use the definition:

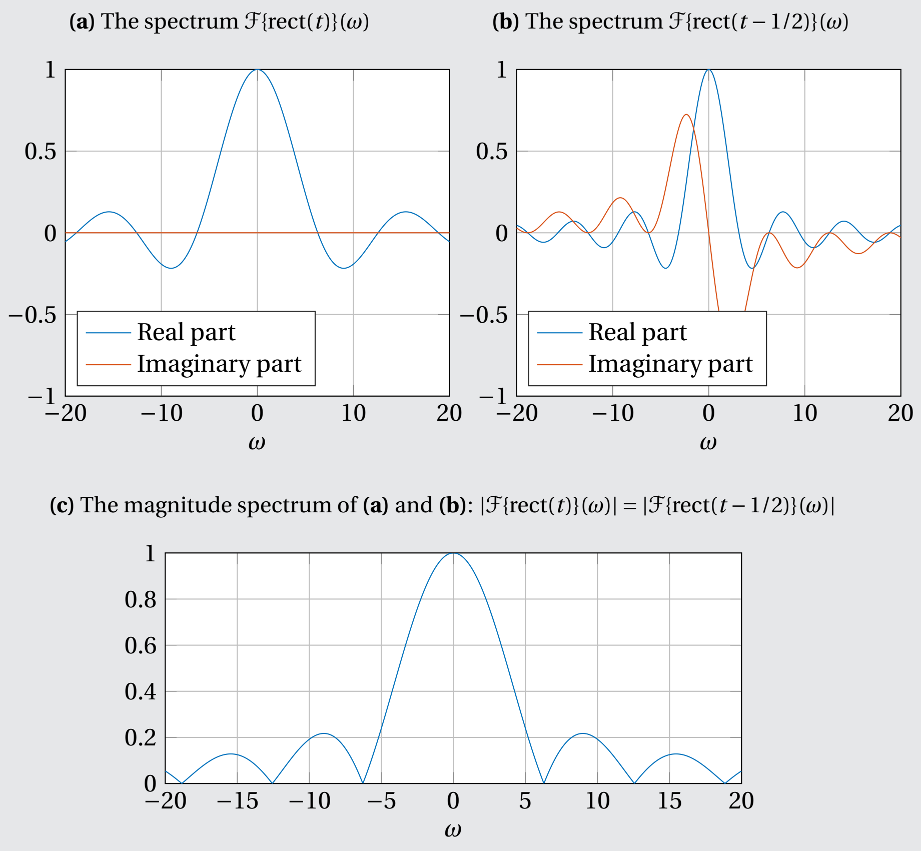
$$\begin{aligned}\mathcal{F}\{\text{rect}(t)\}(\omega) &= \int_{-\infty}^{\infty} \mathbf{1}_{[-1/2,1/2]} e^{-i\omega t}\,\mathrm{d}t = \int_{-1/2}^{1/2} e^{-i\omega t}\,\mathrm{d}t \\ &= \frac{1}{-i\omega} e^{-i\omega t}\Big|_{-1/2}^{1/2} = -\frac{1}{i\omega}\left(e^{-i\omega/2} - e^{i\omega/2}\right) \\ &= \frac{\sin(\omega/2)}{\omega/2} := \text{sinc}\left(\frac{\omega}{2}\right).\end{aligned}$$

This result is known as the sinc function. A plot of the Fourier transform can be found in Figure 4.1a. If we translate the rectangle pulse with $1/2$, we could use the translation property of the Fourier transform (A.2) to find that

$$\mathcal{F}\{\text{rect}(t-1/2)\}(\omega) = e^{-i\omega/2}\text{sinc}(\omega/2).$$

The Fourier transform of this translated pulse is depicted in Figure 4.1b.

Figure 4.1: Example: the CFT of a regular and a shifted rect-function.



Spectrum

In the last example we have seen that the Fourier transform of a signal can consists of a real and an imaginary part. We could decompose it like $\mathcal{F}x(\omega) = X(\omega) = Y(\omega) + iZ(\omega)$. This is known as the *Cartesian* or *quadrature form* [7]. An other well known form is the *polar form*: $X(\omega) = |X(\omega)|e^{i\theta(\omega)}$. This is also known as the *magnitude-phase form*, where the real functions $|X(\omega)|$ and $\theta(\omega)$ denote the magnitude and the phase of the signal. To determine whether frequency components are present in a signal, one could examine only the *magnitude spectrum* $|X(\omega)|$, often referred to as the *spectrum* of $x(t)$.

Example 4.2 (Magnitude spectrum) In Example 4.1 the Fourier transforms of two different rect-functions has been shown. When the modulus of these two signals is observed, something special is exposed in Figure 4.1c. Both the 'regular' and the shifted rectangle pulse have the same *magnitude spectrum*! From this we can conclude that the *phase* of the Fourier transform adds information about the time shift of the frequencies. This shift is added, by the used formula (A.2).

Physical waveforms

Physical waveforms $x(t)$, of finite energy and length (time duration of the signal is T), may be represented by the Fourier series over the interval $a < t < a + T$:

$$x(t) = \sum_{n=-\infty}^{\infty} c_n e^{in\omega_0 t}, \qquad \text{with the Fourier coefficients } c_n = \frac{1}{T}\int_a^{a+T} x(t)e^{-in\omega_0 t}\,\mathrm{d}t,$$

where $\omega_0 = 2\pi/T$ [7]. If $x(t)$ is a periodic function with period T, this representation is valid for all times, for the resulting Fourier transform also has period T. This transformation maps the waveform of time T to the formal Fourier basis (4.1).

Quadrature Fourier Series

The quadrature Fourier series [7] uses the property that the complex exponential can be written as a sum of a sine and a cosine, which leads to another representation of any physical waveform $x(t)$ (over the interval $a < t < a + T$):

$$x(t) = \sum_{n=-\infty}^{\infty} a_n \cos n\omega_0 t + b_n \sin n\omega_0 t \tag{4.3}$$

$$\text{where } a_n = \begin{cases} \frac{1}{T}\int_a^{a+T} x(t)\,\mathrm{d}t & n = 0 \\ \frac{2}{T}\int_a^{a+T} x(t)\cos n\omega_0 t\,\mathrm{d}t & n \geq 1 \end{cases} \qquad \text{and } b_n = \begin{cases} 0 & n = 0 \\ \frac{2}{T}\int_a^{a+T} x(t)\sin n\omega_0 t\,\mathrm{d}t & n \geq 1 \end{cases}.$$

These Fourier series can also be described in polar form [7], but this form is beyond the scope of this literature thesis. The Fourier transform is sometimes referred to the decomposition of a signal into sines and cosines. This interpretation comes from the quadrature series.

Convolution

The convolution of two signals $x(t)$, $y(t)$ is defined by

$$(x * y)(t) := \int_{-\infty}^{\infty} x(\tau)y(t-\tau)\,\mathrm{d}\tau = \int_{-\infty}^{\infty} x(t-\tau)y(\tau)\,\mathrm{d}\tau = (y * x)(t). \tag{4.4}$$

What the convolution does is 'mapping' $x(t)$ everywhere where $y(t)$ has a value which is not zero. This is shown graphically in Figure 4.2a. It is important to notice that the convolution of two signals in the time domain is equivalent to the multiplication of their transforms in the frequency domain [27], so

$$\mathcal{F}(x * y)(\omega) = X(\omega) \cdot Y(\omega). \tag{4.5}$$

This relation is known as the convolution theorem. Computing the convolution may be very time consuming for the high number of floating point operations needed. Therefore this relationship is sometimes used to compute the convolution of two signals, by multiplying their Fourier transforms.

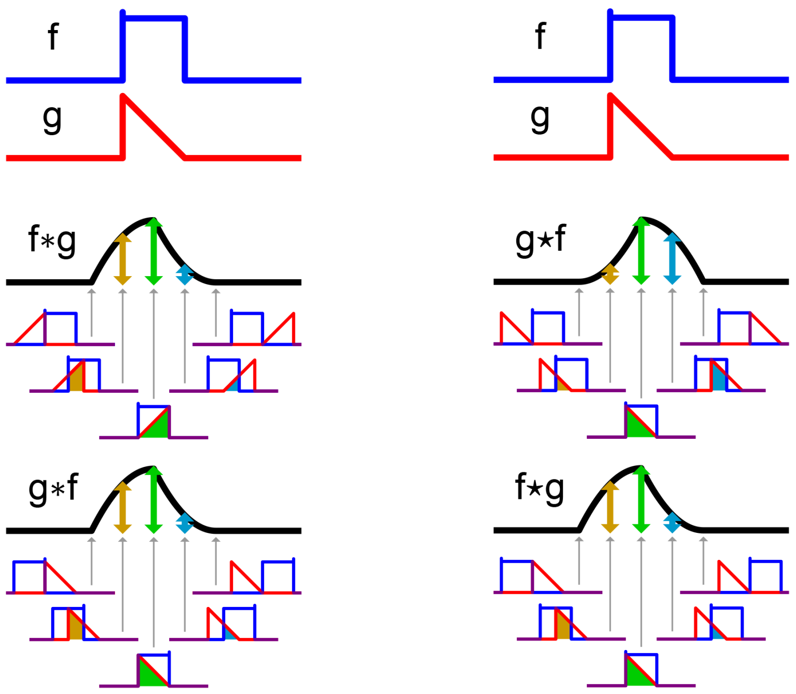
Cross correlation

An other operation of two signals, very similar to the convolution is the cross correlation [15], which is defined for complex signals by

$$(x \star y)(t) := \int_{-\infty}^{\infty} \overline{x(\tau)}\, y(t+\tau)\, \mathrm{d}\tau = \overline{x(-t)} * y(t). \tag{4.6}$$

Note that this operation is not commutative, as is the convolution. The cross correlation 'assesses' how similar two functions are. If there is a high cross correlation, the signals show much similarities, and vice versa, as shown in Figure 4.2b.

The *auto correlation* of a function $x(t)$ is the cross correlation of this function with itself, $(x \star x)(t)$. The cross correlation can be computed in the same manner as the convolution, but the signal g has to be mirrored. Because the Fourier transform is symmetric, the cross correlation satisfies $\mathcal{F}(x\star y)(\omega) = \overline{X(\omega)}\, Y(\omega)$. The autocorrelation thus satisfies $\mathcal{F}(x\star x)(\omega) = \overline{X(\omega)}\, X(\omega) = |X(\omega)|^2$. This Fourier transform of the autocorrelation is known as the *power spectral density*, which has the units of watts per Hertz [7].



(a) The convolutions of two functions f and g. **(b)** The cross correlations of two signals f and g.

Figure 4.2: The difference between the convolution and cross correlation operation[45].

Delta Dirac function

The Delta (Dirac) function is a theoretical function with a support reduced to $t = 0$, but with an integral of 1. In some areas this is referred to as the Delta pulse. This theoretical function simplifies computations, leaving convergence issues aside [21]. The Delta function $\delta(t)$ associates any continuous function $f(t)$ to its value at $t = 0$:

$$\int_{-\infty}^{\infty} \delta(t - t_0) f(t)\, \mathrm{d}t = f(t_0). \tag{4.7}$$

Note that by this property that

$$f * \delta(t) = f(t).$$

Example 4.3 (CFT of $\boldsymbol{\delta(t - t_0)}$) The Fourier transform of the Delta function can be derived from the property above, combined with the convolution property (4.5). However, te derivation from the definition is very short:

$$\mathcal{F}\{\delta(t-a)\}(\omega) = \int_{-\infty}^{\infty} \delta(t-a)e^{-i\omega t}\,\mathrm{d}t = e^{-i\omega a}.$$

For a Delta peak at $t=0$, we find the magnitude spectrum of the Delta function as $\mathcal{F}\{\delta(t)\}(\omega) = 1$, constant over the whole frequency domain. In general a Delta peak results a constant magnitude spectrum, because $|\mathcal{F}\{\delta(t-a)\}(\omega)| = 1\ \forall a \in \mathbb{R}$.

Example 4.4 (CFT of a sine) The best example for the CFT is the transform of a sine or a cosine, for these infinite and C^{∞} functions result in a very nice spectrum. The spectrum of an arbitrary sine $x(t) = \sin(\omega_0 t)$ is:

$$\begin{aligned}\mathcal{F}\{\sin(\omega_0 t)\}(\omega) &= \int_{-\infty}^{\infty} \sin(\omega_0 t)e^{-i\omega t}\,\mathrm{d}t = \int_{-\infty}^{\infty} \frac{1}{2i}\left(e^{i\omega_0 t} - e^{-i\omega_0 t}\right)e^{-i\omega t}\,\mathrm{d}t \\ &= \frac{1}{2i}\int_{-\infty}^{\infty} e^{it(\omega_0-\omega)} - e^{-it(\omega_0+\omega)}\,\mathrm{d}t = \frac{1}{2i}\left[\int_{-\infty}^{\infty} e^{-it(\omega-\omega_0)}\,\mathrm{d}t - \int_{-\infty}^{\infty} e^{-it(\omega+\omega_0)}\,\mathrm{d}t\right] \\ &= \frac{1}{2i}[\delta(\omega+\omega_0) - \delta(\omega-\omega_0)].\end{aligned}$$

In this derivation we used characteristic (A.7). The result is two delta peaks at the frequencies $-\omega_0$ and ω_0. So the signal can thus be explained as a combinations of two harmonics, one at $-\omega_0$ and one at ω. This corresponds with our input signal. Note that both the sine signal and its Fourier spectrum have an integral over the whole domain equal to zero.

4.2. From continuous to discrete

The described theory was perfectly applicable to the early days of signal processing. Most signals were continuous, such as AM radio signals. However nowadays more and more applications switch to digital implementation, such as DAB+ in case of the radio example. Also at Deltares all measurements are done using computers, as described in Chapter 2. Computer sampling leaves us with so called *sampled signals*.

4.2.1. Sampling Theorem

These sampled signals are described using the delta function (4.7), discussed before. A discrete signal may be represented as a sum of delta functions. For this we assume a *uniform sampling*, sampled with the *sampling frequency* $f_{\text{sample}} = \Delta t^{-1}$ [Hz]. Assume a continuous signal $x(t)$ is being sampled, then the uniform sampling of this signal is described as

$$x_{\text{sampled}}(t) = \sum_{k=-\infty}^{\infty} x(t)\delta(t-k\Delta t) = \sum_{k=-\infty}^{\infty} x(kT)\delta(t-k\Delta t).$$

Now calculate the Fourier transform of $x_{\text{sampled}}(t)$, using the linearity property (A.1) of the Fourier transform and the Fourier transform of the Delta function (A.9) to derive the *discrete time Fourier transform* (DTFT, discrete in time, continuous in frequency):

$$\mathcal{F}x_{\text{sampled}}(\omega) = X_{\text{sampled}}(\omega) = \sum_{n=-\infty}^{\infty} x(k\Delta t)e^{-i\omega n\Delta t}.$$

Most sampled signals $x(kT) = x[k]$ are finite. The drawbacks of this approach are discussed in Section 4.3. For now, we will work with the abbreviated expression for a signal of length N

$$X(\omega) = \sum_{k=0}^{N-1} x(k\Delta t)e^{-i\omega k\Delta t}. \tag{4.8}$$

LTI systems

Fourier analysis is a tool used a lot for LTI systems. Not all systems are coverable by Fourier analysis, which sometimes forces signal analysts to use *Laplace transformations* or *Z-transformations* [7]. Fourier analysis is a much used tool because the Fourier transform is a linear operator (A.1), so linearity of such systems can be studied both in the time and the frequency domain.

4.2.2. DFT - Discrete Fourier Transform

Expression (4.8) can be simplified further, by discretization of the frequency domain. Mallat [21] notes that the family of basis functions

$$\left\{ e_n[k] = \exp\left(\frac{i2\pi nk}{N}\right) \right\}_{0\le n\le N} \tag{4.9}$$

is an even orthonormal basis of signals with period N. Using this basis, the *discrete Fourier transform* (DFT) is defined. This transform is discrete in both time and frequency. Now again consider a discrete time signal, given by $x[k]$, with $k \in \mathbb{N}$ and finite with a duration of time T, then the DFT becomes

$$\mathcal{F}_{\mathrm{d}} x[n] := X[n] = \sum_{k=0}^{k=N-1} x[k] e^{-i2\pi nk/N}, \qquad \text{where } n = 0, 1, 2, \ldots, N-1, \tag{4.10}$$

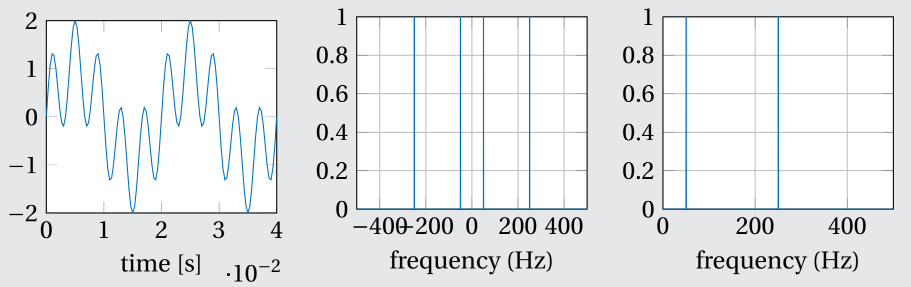
and the inverse operation is given by

$$\mathcal{F}_{\mathrm{d}}^{-1} X[k] := x[k] = \frac{1}{N} \sum_{n=0}^{N-1} X[n] e^{i2\pi nk/N}, \qquad \text{where } k = 0, 1, 2, \ldots, N-1.$$

From now on the index k is used for indexing the discrete time domain and the index n for the discrete frequency domain. Note that the basis functions (4.9) are independent of the sampling frequency or the total duration of the signal $x[n]$. The resulting DFT has domain $[-f_{\text{sample}}/2, f_{\text{sample}}/2]$, which will be elaborated further in Section 4.3.

Example 4.5 (DFT of a sum of sines) In this first example we will pay attention to the function $x(t) = \sin(100\pi t) + \sin(500\pi t)$. The resulting spectrum can be found in Figure 4.3b. Note that this spectrum is two sided, as predicted. One half of the information of this spectrum is superfluous. Therefore, one half can be discarded, leading to the one sided spectrum in Figure 4.3c.

The MATLAB-code for this example: Listing B.1.



(a) Two periods of the signal. **(b)** The two sided normalized spectrum. **(c)** The one sided normalized spectrum.

Figure 4.3: Spectrum of the signal from Example 4.5.

Discrete convolution and cross correlation

The discrete convolution of two discrete time signals $x[n]$ and $y[n]$ (assume both length N) is given by:

$$(x * y)[n] = \sum_{k=-\infty}^{\infty} x[k] y[n-k], \qquad \text{for } 0 \le n < N, \tag{4.11}$$

also a commutative operation as the continuous convolution (4.4). Again, the discrete cross correlation can be computed in the same way as (4.6): $(x \star y)[n] = \overline{x[-n]} * y[n]$. To compute the discrete convolution, information from $x[n]$, $n \geq N$ is needed, but the signal does not exist there. The *circular convolution* considers these two signals as periodic, such that $x[N] = x[0]$, $x[N+1] = x[1]$ etc. The computation (4.11) can be abbreviated to

$$(x \circledast y)[n] = \sum_{k=0}^{N-1} x[k]y[n-k],$$

where the $\circledast$ denotes the *circular* or *cyclic* convolution. If in $x[k]$ from (4.10) the values of $x[0]$ and $x[N-1]$ are very different, the big transition in the 'periodic signal' will lead to high amplitude Fourier coefficients at high frequencies. Again as for the continuous case of the convolution (4.5) there is a coupling between the DFT and the convolution: Mallat [21] notices that the eigenvectors of the circular convolution operation are the same as the family of basis functions described in (4.9). So for a circular convolution of two signals: $g[k] = f \circledast h[k]$ the DFT of $g[k]$ is obtained by multiplying the DFTs of the two signals $\mathcal{F}_{\mathrm{d}}g[n] = \mathcal{F}_{\mathrm{d}}f[n]\mathcal{F}_{\mathrm{d}}h[n]$.

FFT - Fast Fourier Transform

When (4.10) is considered, notice that the direct computation of the DFT takes N complex multiplications and N complex additions per element $X[i]$, $i = 0, \ldots, N-1$ to find the whole frequency spectrum $X[n]$. his implies the computation is of $\mathcal{O}(N^2)$ arithmetic operations. The fast Fourier transform (FFT) is a faster algorithm to evaluate the DFT [7], where the number of computations is of $\mathcal{O}(N \log_2 N)$ [27]. The FFT algorithm breaks the convolutions used in the DFT computation in shorter convolutions, to lower the number of computations.

Kronecker delta

The discrete case of the delta function is known as the Kronecker delta, this one is also denoted with a δ. It is defined as

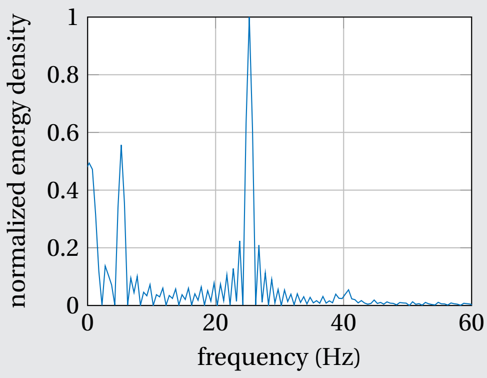
$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{elsewhere} \end{cases} . \tag{4.12}$$

Example 4.6 (DFT of transient signals) To show the largest short coming of the Fourier transform a self composed signal is made, see Figure 4.5. This signal is composed of a number of transient signals. What do we expect for the Fourier transform of this signal? A block wave leads to a sync-reaction, the other parts are all perfect harmonics, which should result in excitations around those frequencies. Because these signals do not all have the same energy contribution, differences in energy densities will occur.

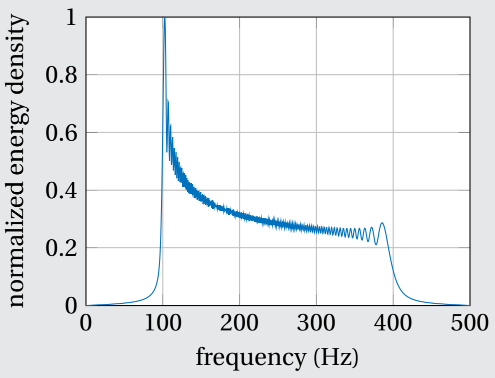
The spectrum is depicted in Figure 4.4a. The expectations can be found in this spectrum. The largest contributor to the signal is the 25 Hz part of the signal, followed by the 5 Hz part. The blockwave also has a large contribution, the sync can also be recognized in this spectrum. Around 40 Hz there is a little excitation. If this signal would have less amplitude or be of shorter duration, it would not be noted. So we can 'rebuild' the signal from this spectrum, but at what times did these signals occur? This information is completely lost.

The MATLAB-code for this example: Listing B.2.

Example 4.7 (DWT of a chirp) The last example of this chapter is made using the `chirp`-function from Matlab. Here we have created a so called quadratic chirp from 100 to 500 Hz: this is a signal which increases in frequency over time. In this case it increases from 100 to 400 Hz in 2 seconds. Quadratic means this signal doubles in the first second to 200 Hz and in the second second to 400 Hz. The resulting spectrum (Figure 4.4b) shows the presence of all these signals and their decreasing energy density. This is all corresponding to the chirp-signal. However again we cannot tell how this signal is composed.



(a) The spectrum of the signal from Example 4.6.



(b) The spectrum of a chirp (Example 4.7).

Figure 4.4: Spectra of the signals from Example 4.6 and 4.7.

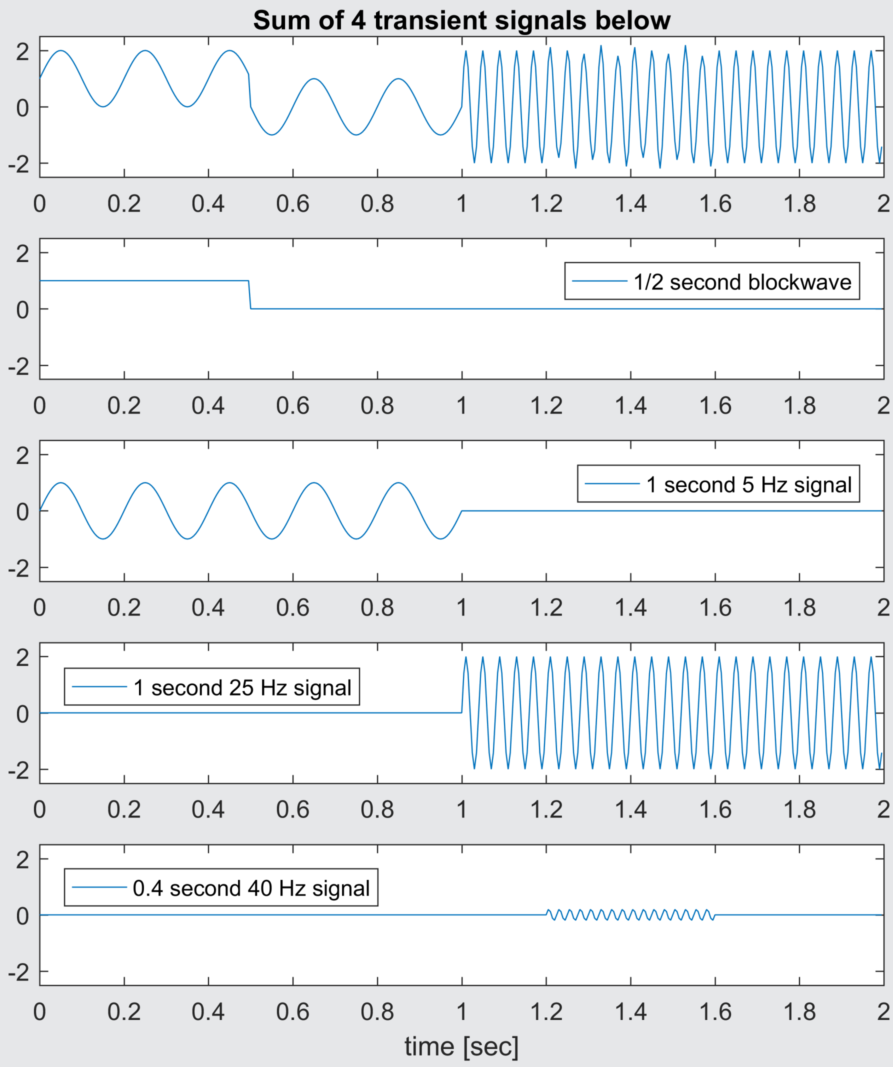


Figure 4.5: The upper signal is composed of four transient signals. The spectrum of this signal is discussed in Example 4.6.

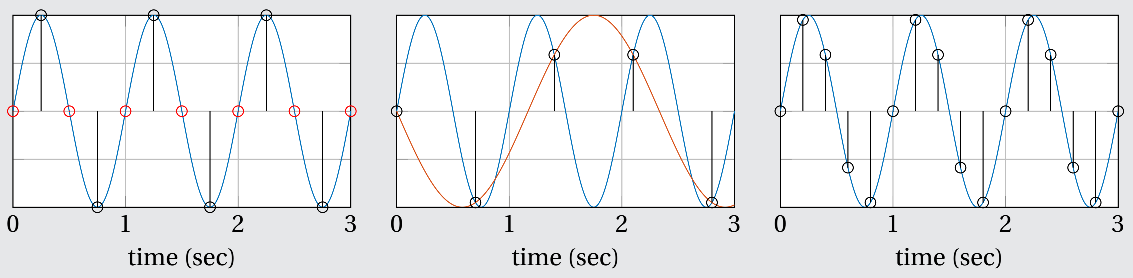
4.3. Nyquist-Shannon Sampling Theorem

As mentioned before the sampling of a signal can be seen as multiplying the sampled signal $x(t)$ by a set of Delta impulses (Section 4.2.1). The Nyquist-Shannon theorem relates properties of the signal to the properties of the sampling frequency and vice versa. This theorem mainly has effect in the Fourier approach used in both the Fourier transform as the short term Fourier transform. For a signal of an interval T, we find the sampling frequency as $f_{\text{sampling}} = 2\pi/T$ [27], resulting in the DFT (4.10). The spectrum of the original signal $x(t)$, $X(\omega)$, repeats at every frequency interval of f_{sampling} in the spectrum of the sampled signal $X_{\text{sampled}}(\omega)$ [27]. These repetitions are called *images*. However if the original spectrum $X(\omega)$ is not limited by a maximum frequency (also known as *bandwidth*) $f_{\max} \leq f_{\text{sampling}}/2$, the repetitions of $X(\omega)$ result in overlapping. The high frequency components of $X(\omega)$ are added to its low frequency components, resulting in a distorted $X_{\text{sampled}}(\omega)$. The repetitions of the spectrum can be cut out by using a *low pass* filter. In this case the choice for a filter selecting the domain $[-f_{\text{sampling}}/2, f_{\text{sampling}}/2]$ is made. This however does not remove the overlapping components.

This distortion of the recovered signal due to insufficient sampling frequency is known as *aliasing* [27]. A clear illustration of aliasing is given in Example 4.8. Signals with sudden transitions or noticeable noise often contain frequency components of which $\omega \to \infty$, these are therefore impossible to catch by any sampling frequency and will always lead to aliasing effect. These might be small however. To recover an analog signal properly from the sampled signal, $f_{\text{sampling}} \geq 2f_{\max}$ [7, 27]. This minimum sampling frequency for a signal is known as the *Nyquist frequency*, defined as $f_{\text{N}} = 2f_{\max}$ [7]. Sampling a signal under the Nyquist frequency is called *undersampling*, sampling with a frequency over the Nyquist frequency is called *oversampling*. Is oversampling a problem? No it is not, but working with an oversampled signal, results in doing computations that are not strictly necessary. However, when the Nyquist rate is based on the bandwidth of the sampled signal, oversampling could reduce noise, aliasing and improves the resolution of your signal. These effects are all explained by a theorem called the Nyquist-Shannon or Sampling Theorem. A nice proof of this theorem is given by Couch [7].

Example 4.8 (Sampling a 1 Hz sine) In Figure 1 four examples of the sampling of a 1 Hz signal ($x(t) = \sin(2\pi t)$) are presented. In 4.6a sampling at the Nyquist rate of 2 Hz is done, with two different starting points. One would say on basis of the black sampling, the signal can be reconstructed. However, the red sampling at the same frequency shows this could lead to some fatal mistakes, where in this case the signal cannot show in the sample.

In 4.6b, the aliasing is made clear. The sampled signal of the 1 Hz sine, can also be interpreted as a sine with frequency 7/6 Hz. This is where the aliasing comes from. Note that the aliasing results in images for $f_{\text{sampled}} > f_{\text{N}}$, but then they do not disturb the spectrum. At last, in 4.6c, a sampling with $f_{\text{sampling}} > f_{\text{N}}$ is shown.



(a) Two different samplings at the Nyquist frequency $f_{\text{N}} = 2$ Hz.

(b) $f_{\text{sampling}} < f_{\text{N}} = 2$ Hz

(c) $f_{\text{sampling}} > f_{\text{N}} = 2$ Hz

Figure 4.6: Sampling of a 1 Hz sinusoidal signal: $x(t) = \sin(2\pi t)$.

4.3.1. Filtering

The images of a spectrum can be taken out of the spectrum using a gate function $G(\omega)$. This function in signal analysis is referred to as a filter. Filters can be applied using computers, but often also *analogue filtering* (in the form of electronics) is used. An audio amplifier for instance often contains a set of analogue filters to determine the audio signals for the low, mid and high speakers. Often measurement equipment apply filtering effects to the measurements. A *low pass filter* is a filter that passes all frequencies under the cutoff frequency ω_m, for instance $G(\omega) = \mathbb{1}_{[-\omega_0,\omega_0]}(\omega)$. In this same way, *bandpass* and *high pass* filters are used to respectively pass signals in a certain bandwidth and above a certain cutoff frequency. This filtering can be interpreted as a multiplication in the frequency domain: $X_\text{filtered}(\omega) = X_\text{sampled}(\omega)G(\omega)$. If cut off windows are used, the *Gibbs phenomenon* or *ripple effect* arises [27]. This is best shown with an example.

Example 4.9 (Gibbs phenomenon) For this example three estimations of the block function $x(t) = \mathbb{1}_{[-7,7]}(t)$ using only harmonics have been made. These estimations are shown in 5. By Example 4.1 we know that the function $x(t)$ has an infinite spectrum. We clearly see that the more harmonics are used to estimate the original, the better the estimation looks like the original. However, all estimations show oscillations near the edges of the block wave. Note however that the overshoot of these oscillations is of equal size for all estimations. This effect is known as the *Gibbs phenomenon.*

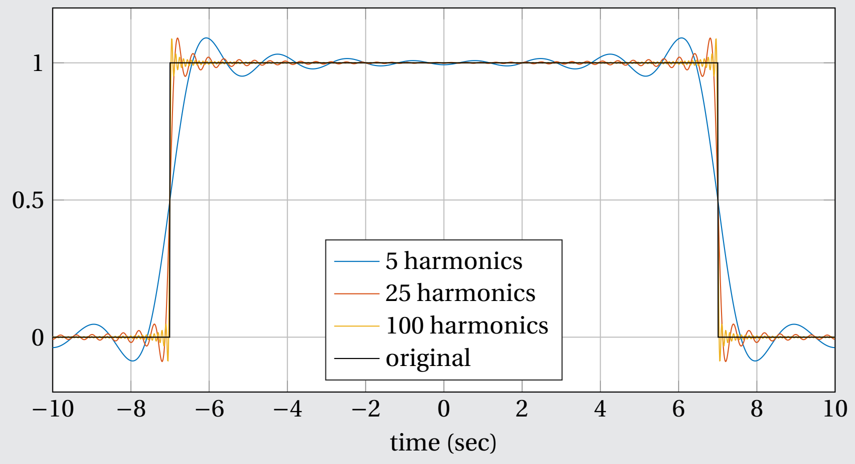


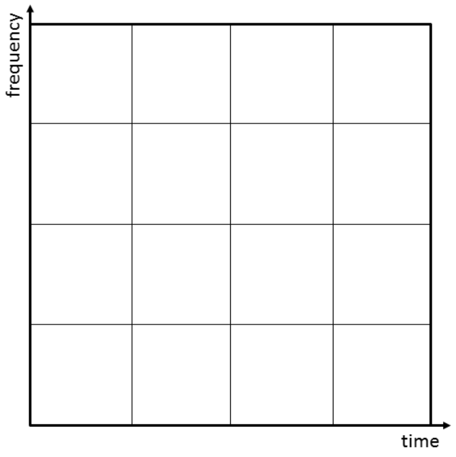
Figure 4.7: An example of the Gibbs phenomenon for the signal $x(t) = \mathbb{1}_{[-7,7]}(t)$.

By truncating the spectrum of the signal as in Example 4.9, a lot of relative large ripples appear. Choosing a longer window leads to increased ripple frequency, with no effect on the ripples magnitude [27]. A solution to this problem is choosing smoother windows, however, this results in loss of frequency resolution. The filter determines the quality of the signal approximation [27]. Because the DFT is done using a computer, the signal is of finite length T. This can be interpreted as multiplying the original infinite signal with a block window $w(t) = \mathbb{1}_{[0,T]}$. By the inversion property of the Fourier transform (A.7), this will result in rippling effects in the spectrum, as observed in the in Figure 4.4.

4.3.2. Fourier transform requirements

The Fourier transform cannot just be applied to all functions existing. In mathematical terms the CWT only exists for functions in the L^2 space [15]. This ensures the convergence of the integral in Equation 6.4. In physical terms one would speak of functions with a finite energy [7]. Discrete signals can be interpreted as truncated signals, which will thus always be of finite energy. For these, the Fourier transform should be restricted to the representation of smooth, 2π-periodic functions [30]. When used to represent non-periodic functions, or functions with discontinuities, the Gibbs-effect will be present in the spectrum, and convergence around the boundaries will be non-uniform. There are efficient ways to weaken this effect [30], but this is a large disadvantage of the Fourier transform. Vice versa, when signals in the Fourier domain are truncated, the Gibbs effect shows in the time domain.

5 Temporal and frequency information



In the previous chapter the extraction of frequency information from a signal has been discussed. As mentioned in Chapter 3, the goal is to extract a connection between time and frequency information from a time signal. In this chapter the steps of this process will be discussed. First the windowed Fourier transform (WFT) is explained, from which the short term Fourier transform (STFT) is deducted. The next chapter will use the same theory to derive the wavelet transform.

5.1. WFT - Windowed Fourier Transform

The WFT makes use of a so-called *window function* $g : \mathbb{R} \to \mathbb{R}_{\geq 0}$ [4]. This window function should have the property that it has compact support containing 0, or at least a maximum at $t = 0$, decaying fast for $|t| \to \infty$ and $\int_{-\infty}^{\infty} g(t)\,\mathrm{d}t = 1$. A widely used window is the *Gabor transform window*, after the physicist who was one of the first to use the WFT systematically. He remarked that the window $\mathcal{N}_{\sigma,0}$ has some optimizing properties. The Gabor transform window is given by

$$g(t) = \mathcal{N}_{\sigma,0}(t) := \frac{1}{\sqrt{2\pi}\sigma} e^{-t^2/2\sigma^2}, \qquad \text{with } \sigma \text{ constant.} \tag{5.1}$$

The choice for a window like (5.1) instead of a rectangular pulse for instance is obvious to the reader familiar with Fourier transform properties. The sharp cut off by a rectangular window, will lead to rippling effects, described in Section 4.3. These contributions will disturb the results. The WFT is the CFT of the signal $x(t)$ multiplied with the window function $g(t)$.

5.2. STFT - Short Term Fourier Transform

The chosen window g will be slid over the signal $x(t)$ to not select the full signal, only parts of it. Therefore the *window transform* is defined as

$$g_s : \quad t \mapsto g(t-s), \tag{5.2}$$

a translation by $s \in \mathbb{R}$ of the window g. Note that for $s > 0$ the window is translated to the right. Now define:

$$\mathcal{G}x(\omega, s) := \int_{-\infty}^{\infty} x(t) g(t-s) e^{-i\omega t}\,\mathrm{d}t = \int_{-\infty}^{\infty} x(t) g_s(t) e^{-i\omega t}\,\mathrm{d}t.$$

This use of the translated window transform is widely known as the s*hort term Fourier transform* (STFT) because the multiplication by $g(t-s)$ localizes the Fourier integral in the neighbourhood of $t = s$ [21]. The value of $\mathcal{G}x(\omega,s)$ represents again the complex amplitude by which the pure harmonic $e^{i\omega t}$ is present in the signal $x(t)g_s(t)$ for that particular s. By the redundancy of information of the signal $x(t)$ in the STFT $\mathcal{G}x(\omega,s)$, there are many inverse transformations defined [4]. The representation of this transformation is often given by a *spectrogram.* The spectrogram, denoted by $P_{\mathcal{G}}$ measures the energy of $x(t)$ in the time-frequency neighborhood of (ω, s). This neighborhood is specified by its so called Heisenberg box $h_{\omega,s}$ (see section 5.3.1). Where

$$P_{\mathcal{G}}x(\omega,s) = |\mathcal{G}x(\omega,s)|^2 = \left|\int_{-\infty}^{\infty} x(t)g(t-s)e^{-i\omega t}\,\mathrm{d}t\right|^2.$$

In the given examples this spectrogram will be discussed further. For large signals the spectrogram can be come quite hard to read. The two most applied enhancers are window overlapping and bin averaging. This last one decreases time frequency resolution even further.

5.2.1. The discrete cases

As for the DFT and DTFT, the discrete STFT and discrete time STFT, respectively DSTFT and DTSTFT, are defined. The derivation is following the same steps as the derivation of the DTFT (Section 4.2.1) and DFT (Section 4.2.2). In the discrete case a *discrete window* or *window sequence* $g[n]$ is chosen. Most of the time this is a symmetric discrete signal of period N, with unit norm $\|g\| = 1$ [21]. For the DTSTFT the discrete signal $x[n]$ is multiplied with the shifted *window sequence* $g[n-k]$, resulting in the expression

$$X(\omega,k) = \sum_{n=-\infty}^{\infty} x[n]g[n-k]e^{-i\omega n} = x[n]e^{i\omega n} * g[n].$$

The same step as for the DFT (Section 4.2.2) is done, leading to the expression for the DSTFT [21]:

$$X[n,k] = \sum_{\ell=0}^{N-1} x[n]g[n-k]\exp\left(\frac{-i2\pi n\ell}{N}\right).$$

Computation time

As discussed before, the computation time of the DFT algorithm is of $\mathcal{O}(N^2)$, which can be shortened to $\mathcal{O}(N\log_2 N)$ using the FFT algorithm. The DSTFT algorithm from (5.2.1) calculates $X[n,k]$. The window size and the overlap of the windows have their effects in the computation time. Again consider a signal $x[k]$ of length N. If for all $0 \le k < N$ $X[n,k]$ is calculated using the FFT algorithm this results in a computation time of at most $\mathcal{O}(N^2\log_2 N)$ to compute the DSTFT of the signal $x(t)$. If there however is very little overlap between the windows, the computation time of the STFT can be close to the FFT computation time.

ISTFT - Inverse Short Term Fourier Transform

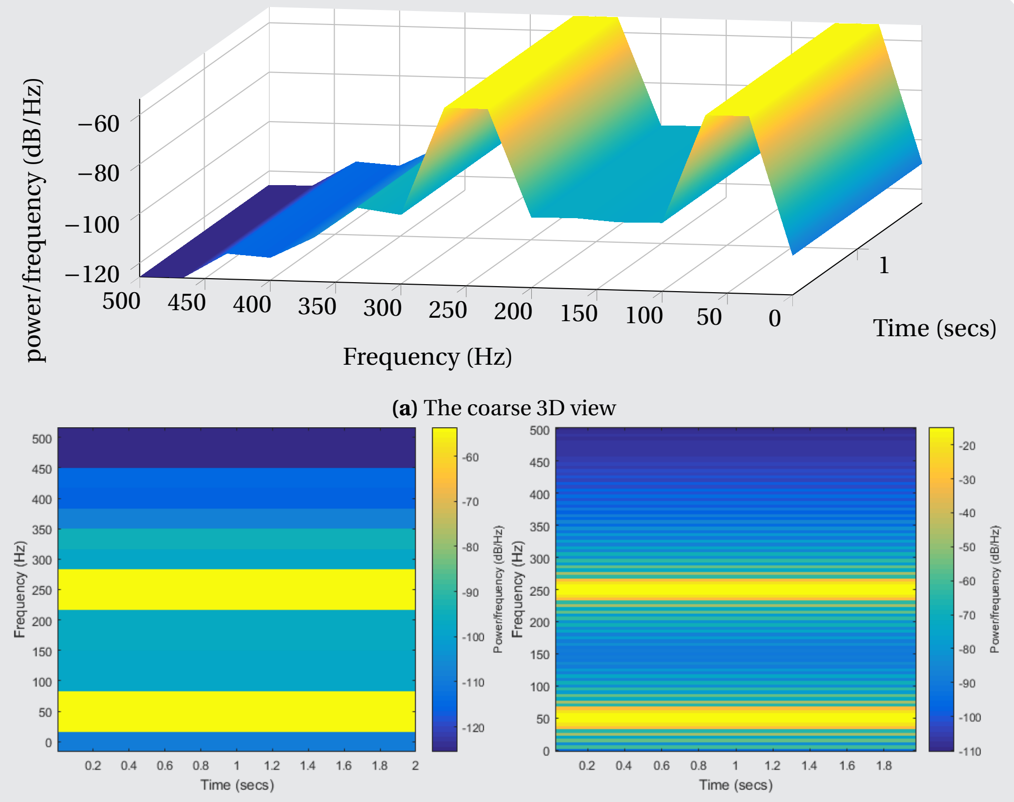
As discussed in the continuous case, there are a lot of different algorithms to compute the inverse of the STFT. Mallat [21] shows that the inverse of the DSTFT of $X[m,\ell]$ is computed using the following summation:

$$x[k] = \frac{1}{N}\sum_{m=0}^{N-1}\sum_{\ell=0}^{N-1} X[m,\ell]g[n-m]\exp\left(\frac{i2\pi\ell n}{N}\right).$$

Example 5.1 (STFT of sum of sines) In this first example the STFT of the function $x(t) = \sin(100\pi t) + \sin(500\pi t)$ will be elaborated. In Figure 5.1 the STFT of this function can be found. The result is, as expected, constant over the time domain, and different over the frequency domain. In Figure 5.1a and 5.1b the same result is shown. From now on the z-axis will not be shown, only the 2D view with a colorbar is used.

Two runs of the STFT have been made, one with a relative small resolution in both time and frequency (Figure 5.1b) and the other with a high resolution in both axis (5.1c). A higher energy density in the finer computation is the most obvious difference, which is easily explained. The energy of these two sines is very concentrated around their frequencies of 50 and 250 Hz. For higher resolution, this energy is spread over a smaller box area, resulting in a larger density. In the coarser computation, we find more energy in between the two frequencies. This is the result of the window. It is clear that the Fourier spectrum of this signal (Figure 4.3c) gives us better information than the STFT.

The MATLAB-code for this example: Listing B.3.



(a) The coarse 3D view

(b) The coarse 2D view

(c) The fine 2D view

Figure 5.1: The STFT of the signal from Example 5.1.

5.3. Regularity and decay

Till thus far the Fourier transform and the short term Fourier transform have been addressed. We have seen some nice results using the different transforms. There are some limitations however, especially for the STFT. These limitations are dependent on the regularity and decay of the chosen window function. The regularity of a signal $x(t)$ affects the decay of its Fourier transform $|X(\omega)|$ and vice versa [21]. The decay depends completely on the worst singular behaviour of $x(t)$. If there exists a constant K and $\epsilon > 0$ such that

$$|X(\omega)| \leq \frac{K}{1+|\omega|^{p+1+\epsilon}}, \qquad \text{then } x \in C^p. \tag{5.3}$$

For instance a step function, which is in C^0, results in a decay of $\mathcal{O}(1/|\omega|)$.

5.3.1. Uncertainty principle

Equation 5.3 shows that for a fast decaying spectrum $|X(\omega)|$ the signal $x(t)$ has regular variations in time. The energy of the signal $x(t)$ therefore has to be spread over a relatively large domain. The *uncertainty principle* relates the localization of energy in time with the localization in frequency. This principle is known to many as the *Heisenberg uncertainty principle*, for its implications in quantum mechanics which were discovered by Werner Heisenberg in the late 1920's. Assume a signal $x(t)$ of which the time spread is reduced, but the total energy is kept constant:

$$x_s(t) = \frac{1}{\sqrt{s}}\, x\left(\frac{t}{s}\right), \tag{5.4}$$

then its Fourier transform (use Appendix A) is $\mathcal{F}x_s(\omega) = \sqrt{s}X(s\omega)$. So the increased localization in the temporal domain ($s < 1$) has led to a decreased localization in the frequency domain. These concentrations of energy in time and frequency are therefore restricted.

This restriction is mathematically described by the uncertainty principle. The simplest explanation of this principle is that if one wants to detect a frequency, one has to observe at least one period of the signal. So for low frequencies, this takes a lot of time. For high frequencies, very small time ranges have to be considered. The uncertainty principle knows a number of different mathematical formulations [4, 15, 18, 21]. The principle [21] states that the product of the temporal variance σ_t^2 and the frequency variance σ_ω^2 of a signal $x(t)$ and its Fourier transform respectively are restricted by

$$\sigma_t^2\sigma_\omega^2 \geq \frac{1}{4}. \tag{5.5}$$

The equality only holds for special cases of the signal. In addition, if a function $f \neq 0$ has a compact support (the signal is of finite length), then its FT $\mathcal{F}f(\omega)$ cannot have a compact support and vice versa [21].

Heisenberg boxes

The so called *Heisenberg box* is the result of this principle. This box limits the temporal and frequency precision of the STFT. The temporal and frequency variance are determined by the choice of the window function $g(t)$. Assume $g(t)$ real and symmetric, with $g_{s,\xi}(t) = e^{i\xi t}g(t-s)$. Then the variances are only dependent on time and frequency, and therefore independent of the translation s and the modulation ξ (see Equation 5.6). Hence $g_{s,\xi}(t)$ corresponds to a Heisenberg box of area $\sigma_t\sigma_\omega$, centered around (s,ξ) [21], this is illustrated in Figure 5.2. Because the window function does not change, the STFT is of identical resolution across the whole time-frequency plane.

$$\sigma_t^2 = \int_{-\infty}^{\infty} t^2|g(t)|^2\,\mathrm{d}t, \qquad \sigma_\omega^2 = \int_{-\infty}^{\infty} \omega^2|\mathcal{F}\{g(t)\}(\omega)|^2\,\mathrm{d}t. \tag{5.6}$$

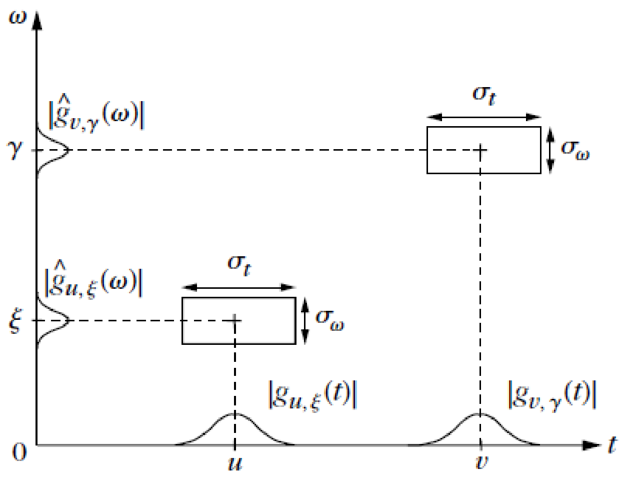


Figure 5.2: Heisenberg boxes of two windowed Fourier transforms, where $\hat{f} := \mathcal{F}f$ [8, 21].

Choice of windows

From the above, we may conclude that the resolution in time and frequency of the STFT is dependent on the spread of the window in time and frequency. Notice that from the Heisenberg uncertainty principle (5.5) follows that the minimal area of an Heisenberg box is 1/2. Mallat [21] shows that this can only be reached if g is a Gaussian window function. However, the organization of this box, width vs. height, can be arranged, to match a specific temporal or frequency resolution. This can be done by scaling the window $g(t)$ as in (5.4).

$g(t)$ finite?

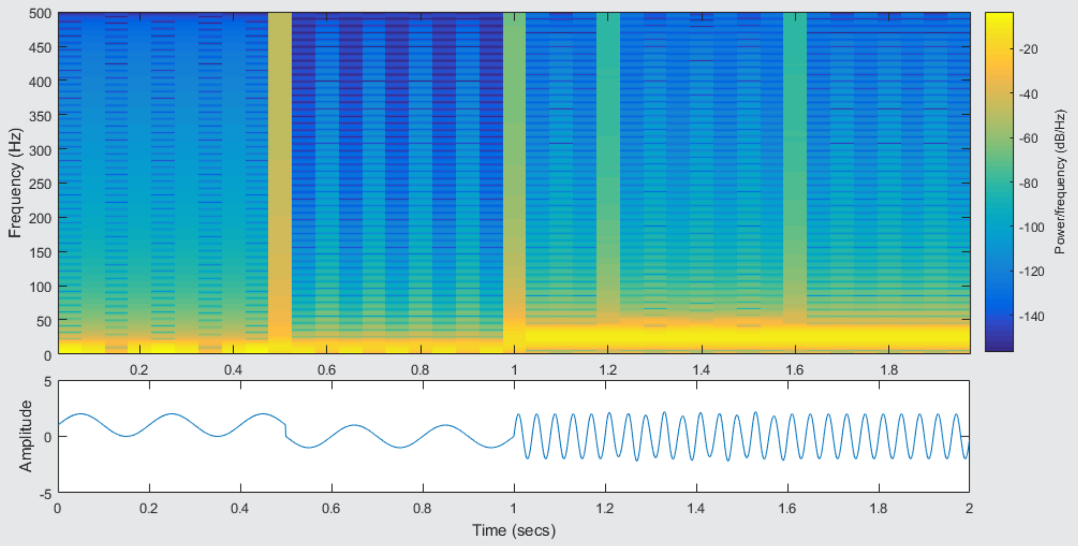
For numerical applications, $g(t)$ must have a compact support, for it is finite. However, this results in an infinite support of the window function in the frequency domain [21]. The frequency resolution of the transform is maximized by concentrating the energy of $\mathcal{F}g(\omega)$ near $\omega = 0$. Then the temporal and frequency variance σ_t and σ_ω are not to deviating. If we for instance choose to shrink σ_ω to 0 (i.e. by choosing $g(t) = \delta(t)$), this results in the normal Fourier transform: high frequency resolution, but no temporal resolution.

Example 5.2 (STFT of transient signals and chirp) Here again we consider the self made signal from Figure 4.5. The STFT shown in Figure 5.3a is hard to read, therefore we start with the example where the frequencies of the signal are increased by a factor 10. The resulting spectrogram is given in Figure 5.3b. In this figure two graphs are shown. The lower graph gives an overview of the signal. The upper graph of the STFT of this signal. On the x-axis of the STFT the time is given, on the y-axis the frequency. Figure 5.3b shows excitations at the points where there are sudden frequency changes, as does the STFT of the original signal in Figure 5.3. These sudden changes can be explained as Delta peaks, which, as we know from Example 4.3, match with a constant energy distribution in the frequency domain. The different frequency components are very clear from the spectrogram in Figure 5.3b. Their exact moments of appearance still has to be guessed, this is the result of the Heisenberg boxes.

If now again Figure 5.3 is considered, we see these results are much less clear. The limitations due to the Heisenberg boxes, which were already visible in Example 4.5, causes hard to read results. When the code is changed to a better frequency resolution, the time resolution decreases.

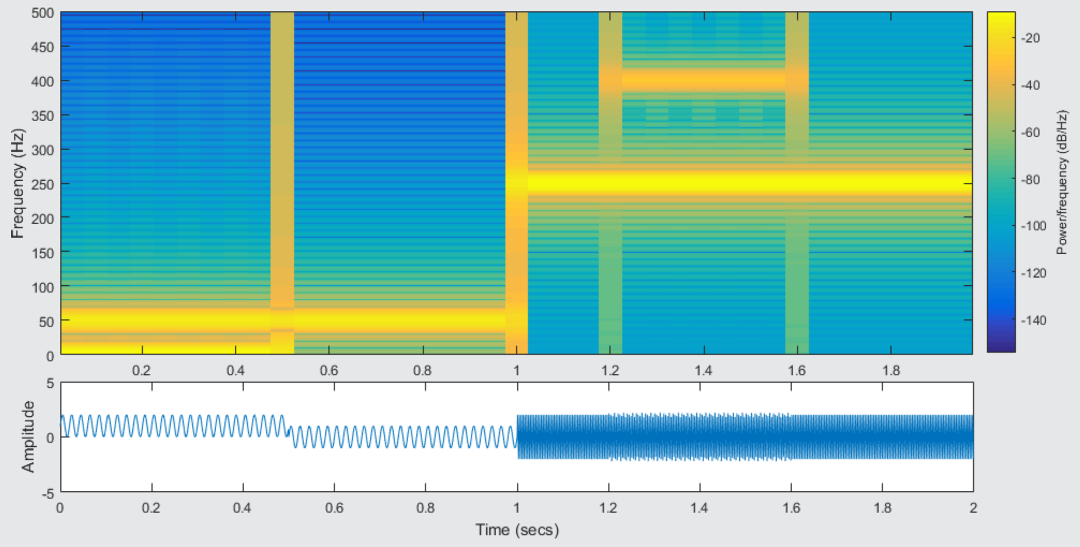
For the chirp (discussed in Example 4.7) the spectrogram is given in Figure 5.3c. Here the content of the signal is actually much clearer than from its Fourier spectrum in Figure 4.4b. As the frequencies change faster when they become higher, the deviation of the frequencies in a 'time level' is larger. In both these cases the STFT has provided much more information than the DFT. However, for frequencies close to each other, we have seen in Figure 5.3a the result is very blurry, giving us very little information.

The MATLAB-code for this example: Listing B.4.

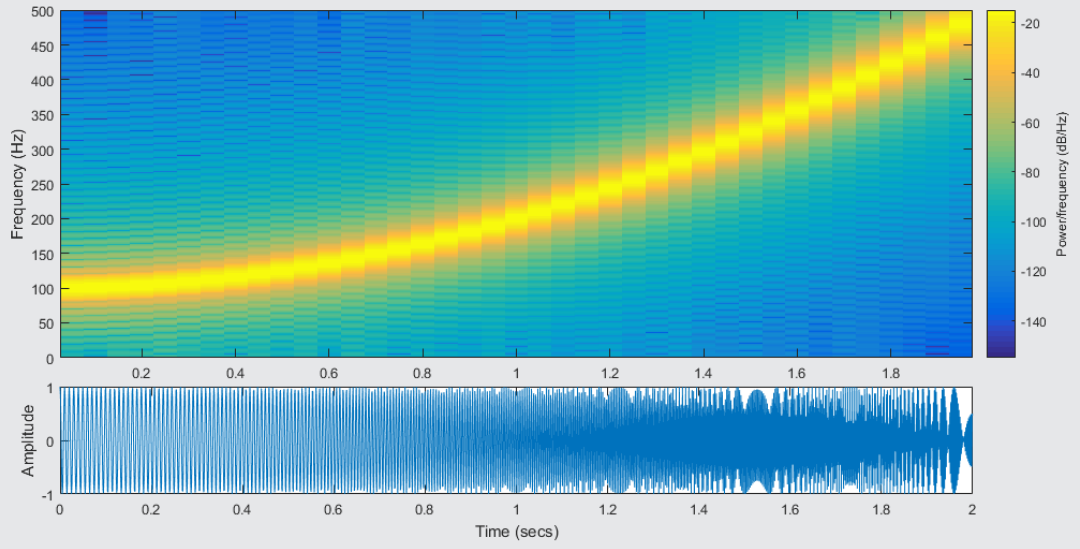


(a) The STFT for the original signal from Figure 4.5.

Figure 5.3: The STFT for the signals from Example 5.2.



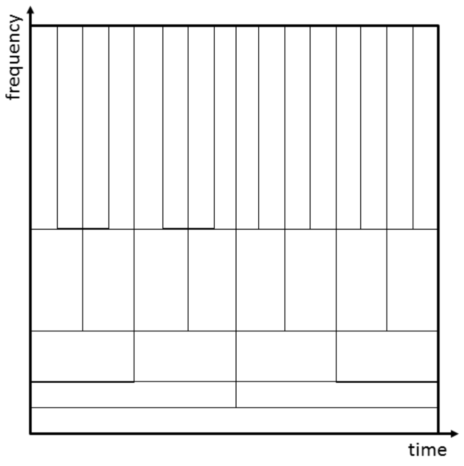
(b) The STFT for the same signal as (a), with 10 times larger frequencies.



(c) The STFT for chirp signal as described in Example 4.7.

Figure 5.3: The STFT for the signals from Example 5.2.

6 Wavelet analysis



In Chapter 5 the Fourier transform has been applied to windowed pieces of a time signal $x(t)$. This method trades resolution in the temporal and frequency domain for information about the coupling of time and frequency. This trade-off is the result of the Heisenberg uncertainty principle, which holds for all time-frequency analyzing methods, including wavelet analysis. Wavelet analysis however uses different resolutions for different frequencies.

In this chapter first the continuous wavelet transform (CWT) will be discussed. Then the discrete wavelet transform (DWT) will follow. With the application in mind, the DWT will be elaborated more thoroughly than the CWT. The DWT is explained from the definition of a multi resolution analysis (MRA, Section 6.3) where both orthogonal and biorthogonal MRAs are explained. When the reader is aware of the two types of MRA, algorithms for the DWT will be elaborated. Throughout this chapter a lot of theory to derive wavelets is addressed. Therefore this chapter will conclude with a summary of the most important equations to derive wavelets and show an example using the well known Daubechies wavelet, followed by the main characteristics of other wavelets.

In Chapter 4 we have seen that the Fourier transform approximates a signal using a sum of sines and cosines , i.e. (4.3). The only difference of the wavelet transform with the Fourier transform is the use of short waves instead of the infinite sines and cosines. Their main characteristic is having compact support; they do not last forever. Instead of multiplying the signal with an exponential function, wavelet functions $\psi(t)$ are used. The result for the Fourier transform is in terms of frequencies, whereas the wavelet transform is expressed in scale and translation. Translation is easy to understand from the equations. The scale is a bit harder to interpret. Scales can be seen as notes for a higher 'pitch' the wavelet is compressed and for a lower pitch it is stretched. This stretching leads to a longer wavelet, hence identifying the time of occurrence for low frequency components is less accurate. Conversely the lower scale wavelet will be shorter and therefore have a better localization in time. The trade off is a lower resolution in frequency, due to the Heisenberg uncertainty principle. In short, the wavelet is scaled and translated to determine the correlation of the wavelet with the signal at all scales and translations.

6.1. CWT - Continuous Wavelet Transform

We will start with the definition of a wavelet, to derive the main idea of the CWT. Later, another definition of the wavelet will be given from a discrete perspective. First the formal definition: a *wavelet* is a function $\psi : \mathbb{R} \to \mathbb{C}$ satisfying the conditions [4]:

$$\psi \in L^2, \tag{6.1a}$$

$$\|\psi\| = 1, \text{ and} \tag{6.1b}$$

$$C_\psi := \int_{\mathbb{R}\setminus\{\}} \frac{|\mathcal{F}\psi(\omega)|^2}{|\omega|}\, d\omega < \infty \tag{6.1c}$$

Here ψ is called a *mother wavelet* or simply *wavelet*. These conditions represent the bare minimum for wavelet theory to function. Wavelets occurring in practice are often L^1 functions as well. Functions that are continuous[1], differentiable and have compact support (the function has a finite length) [4]. Condition 6.1c is also known as the *admissibility condition* for a function to be a wavelet. Narasimhan et al. [27] shows that for C_ψ to be finite, $\mathcal{F}\psi(\omega) = \Phi(\omega)$ should tend to 0 as $\omega \to 0$. Note that then follows

$$\begin{aligned}\Psi(\omega)|_{\omega\to 0} &= \int_{-\infty}^{\infty} \psi(t)e^{-i\omega t}\,\mathrm{d}t\Big|_{\omega\to 0} = 0\\ \Rightarrow \int_{-\infty}^{\infty} \psi(t)\,\mathrm{d}t = 0.\end{aligned}$$

This equation is much weaker than the admissibility condition, but it is often sufficient for practical purposes. It also implies that the total area under the wavelet should be zero. The function $\psi(t)$ has to oscillate to comply to this condition, hence the name wavelet [27]. Some common wavelets are depicted in Figure 6.1, a more elaborate overview of different wavelets with their strengths and weaknesses can be found in Section 6.5.

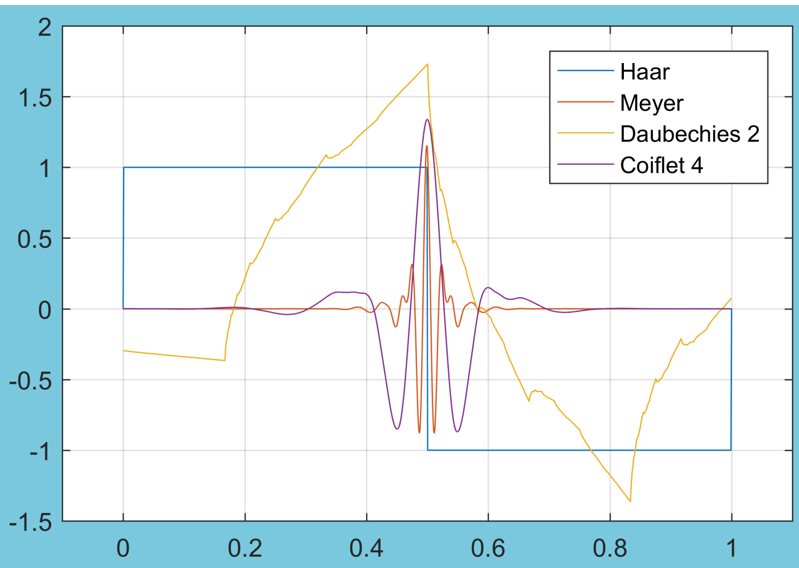


Figure 6.1: Four common real wavelets, scaled to fit in the domain [0,1] (therefore not all complying to condition 6.1b).

Scaling

Now a wavelet ψ is chosen. For an arbitrary $a \in \mathbb{R}\setminus\{0\}$ we define the *scaling* [4] (or *dilation* [27]) of the wavelet ψ as

$$\psi_a(t) := |a|^{-1/2}\psi\left(\frac{t}{a}\right). \tag{6.2}$$

This function is obtained from ψ by scaling the graph in the vertical direction (by t/a), reflecting the wavelet ψ for $a<0$ and scaling it in the horizontal direction by the factor $|a|$. This choice has been made such that requirement (6.1b) holds for all ψ_a $(a \neq 0)$, by

$$\|\psi_a\| = \int_{-\infty}^{\infty} |\psi_a(t)|^2\,\mathrm{d}t = \int_{-\infty}^{\infty} \left||a|^{-1/2}\psi\left(\frac{t}{a}\right)\right|^2 \mathrm{d}t = \frac{1}{|a|}\int_{-\infty}^{\infty}\left|\psi\left(\frac{t}{a}\right)\right|^2 \mathrm{d}t = \frac{1}{|a|}\int_{-\infty}^{\infty}\left|\psi(t')\right|^2 |a|\,\mathrm{d}t' = 1.$$

For the obvious reason this process is called the scaling of a wavelet and the variable a is referred to as the *scaling factor* or *dilation parameter*. Note this process is only valid for $a \neq 0$.

[1]Exception: the Haar wavelet shown in Figure 6.1.

Translating

After the process of scaling, the function ψ_a is *translated* along the time axis by the amount b. This translation parameter indicates the location of the wavelet in time. The translation is chosen such that for $b > 0$ the wavelet ψ_a is translated in the positive direction (as in (5.2)), from which the function

$$\psi_{a,b}(t) := \psi_a(t-b) = |a|^{-1/2}\psi\left(\frac{t-b}{a}\right) \tag{6.3}$$

is obtained. The resulting wavelets $\psi_{a,b}$ are called the *daughter wavelets* [27]. These daughter wavelets exist on the full $\mathbb{R}\times\mathbb{R}$-domain except for $\{(a,b)|a=0\}$, this set is defined as $R^2_- := \{(a,b)|a\in\mathbb{R}\setminus 0,\ b\in\mathbb{R}\}$. It is obvious that $\|\psi_{a,b}\| = 1\ \forall(a,b)\in\mathbb{R}^2_-$.

6.1.1. The wavelet transform

As said before, a trade off has to be made between detailed time and frequency information, for you cannot have both. As shown in Section 5.3.1 the scaling of the function adapts the variance in time and frequency. From this scaling follows different resolutions in time and frequency for different frequencies. Remember, the mother wavelet ψ is fixed, then the *continuous wavelet transform* (CWT) of a time signal $x\in L^2$ with respect to the chosen wavelet ψ is defined as

$$\begin{aligned}\mathcal{W}x(a,b) = X(a,b) := \langle x,\psi_{ab}\rangle &= \int_{-\infty}^{\infty} x(t)\overline{\psi_{ab}(t)}\ \mathrm{d}t = |a|^{-1/2}\int_{-\infty}^{\infty} x(t)\overline{\psi\left(\frac{t-b}{a}\right)}\ \mathrm{d}t \qquad (6.4)\\ &= \int_{-\infty}^{\infty} x(t)\overline{\psi_a(t-b)}\ \mathrm{d}t = (x*\overline{\psi_a})(b), \qquad (a\neq 0)\end{aligned}$$

From this definition we see the domain definition of $\mathcal{W}x$ is $\mathbb{R}^2_-$. In wavelet theory the a-axis is scaled vertically and the b-axis horizontally, in contrary to the (x,y) scaling the most readers are used to. This choice has been made because the translation b is connected to the time, which is expressed on the horizontal axis too.

Note the resemblance of the wavelet transform with the cross correlation (4.6). In (6.4) the wavelet transform is also written as a convolution [27]. As discussed before, the cross correlation assesses how much one function resembles the other. In the case of the wavelet transform, the resemblance of the signal $x(t)$ with the particular wavelet ψ_{ab} is expressed in the function $\mathcal{W}x(a,b)$.

Very often the domain of the transform $\mathcal{W}x$ is restricted to positive a-values. Suitable wavelets for these transforms are discussed later. If a or b is chosen from a discrete set, the wavelet transform is called discrete. For a small value of $|a|$, ψ_{ab} is of shorter duration and therefore covers a large frequency range. A larger value of $|a|$ results in a longer window and so covers a shorter frequency range. The parameter $1/a$ therefore becomes a measure of frequency. In Figure 6.2b it is clear that the translation-scale domain differs from the time-frequency domain. In the scale domain the highest frequencies are found at the bottom of the axis, where the frequency axis displays them at the top.

The inverse wavelet transform

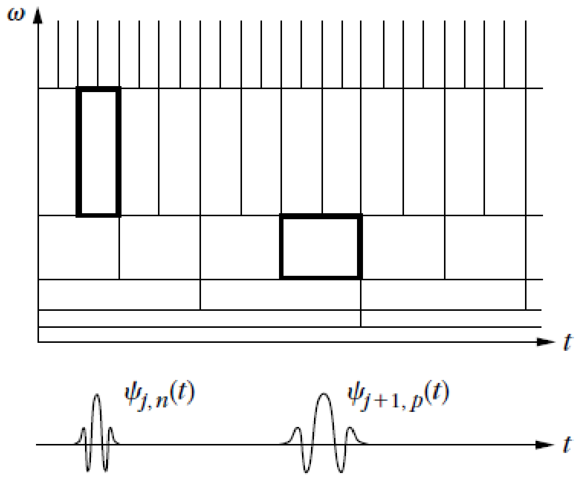
As for the Fourier transform, an inverse function for the wavelet transform is constructed as [27]:

$$x(t) = \frac{1}{C_\psi}\int_{\mathbb{R}}\left[\int_{\mathbb{R}\setminus 0} X(a,b)\psi_{a,b}(t)\frac{\mathrm{d}a}{|a|^2}\right]\mathrm{d}b, \tag{6.5}$$

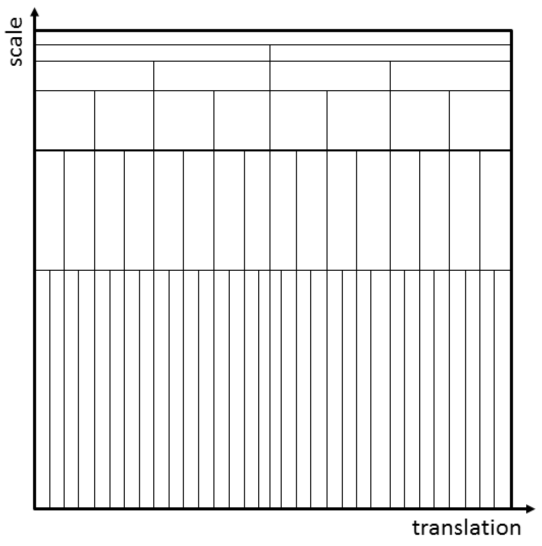
where C_ψ is defined in (6.1c). A readable proof of this inverse wavelet transform is given by Narasimhan et al. [27, page 8.5].

6.1.2. Heisenberg boxes and the scalogram

The wavelet transform does look the same as the STFT: an integral over the signal multiplied with a function ψ_{ab}. The energy spread of a wavelet ψ_{ab} corresponds to a Heisenberg box with a center, with size $a\sigma_t$ along the time axis and σ_ω/a along the frequency axis [21]. This is shown in Figure 6.2a: shorter wavelets cover less time but higher and more frequencies. Longer wavelets cover more time, but less and lower frequencies. The formal definition of the wavelet transform is in terms of scale a and translation b, which results in a Heisenberg box division of the a,b-plane as in Figure 6.2b.



(a) Heisenberg boxes of two different scalings of a wavelet [8].



(b) The arrangement of the translation-scale domain.

Figure 6.2: The time frequency domain arrangement by wavelets.

The *scalogram* [21, 28] or *wavelet power spectrum* [40] is the energy density of the wavelet transform, often used to visualize the energy density over the signal. The scalogram measures the energy of a signal $x(t)$ in the Heisenbergbox of the wavelet ψ_{ab}. For a wavelet ψ_{ab} centered around $(t,\omega) = (b, \omega = \eta/a)$ the scalogram $P_{\mathcal{W}}$ is defined as [21]:

$$P_{\mathcal{W}}x(b,\omega) = |\mathcal{W}x(a,b)|^2 = \left|\mathcal{W}x\left(b, \frac{\eta}{\omega}\right)\right|^2. \tag{6.6}$$

This is a function with two variables. The x axis represents time (or translation b), the y axis represents the scale a or as in the final part of the expression in terms of frequency. Torrence and Compo [40] has described a process to interpret the statistical significance of the scalogram to be able to quantify the wavelet transform results.

6.1.3. Computation of the CWT

The CWT of a signal often has to be computed analytically. Popular signal processing programs such as Matlab or Python posses specific wavelet analysis tools. Both these toolboxes have functions to compute the CWT of a signal. The CWT function of Python is shown in Listing B.6. In this code it becomes clear that the 'continuous wavelet transform' is computed using discrete convolutions. So not the continuous expression is computed, but an approach up to a certain level. The CWT algorithm does not have a fast computational approach [27] and this discrete approach is computational expensive. The computation for a relative short signal takes a lot of time and needs a lot of memory and is therefore considered ineffective. However, before further elaboration on this subject, a few continuous wavelet transform examples using this function from Matlab will be given, to give some insight in the results.

Complex Morlet wavelet In the following example the Complex Morlet wavelet will be used for it is a standard wavelet from MATLABs Wavelet Toolbox [25]. The complex Morlet wavelet is defined by[a]

$$\psi_{\text{cm}} = \frac{1}{\sqrt{B\pi}} \exp(2\pi i \omega_0) \exp\left(\frac{-t^2}{B}\right), \tag{6.7}$$

where B denotes the bandwidth and ω_0 the center frequency of the wavelet. An example of this wavelet with $B = 1.5$ and $\omega_0 = 1$ is given in Figure 6.3.

[a]Expression from MATLAB [26].

Example 6.1 (CWT) In Chapter 4 and 5 we have used three examples: the sum of two sines (Example 4.5, 5.1), the sum of transient signals (Figure 4.5) in Example 4.6 and 5.2 and the chirp signal from Example 4.7 and 5.2. These three examples will be discussed here as well. The wavelet transform result of these three signals can be found in Figure 6.4. The shown results are known as scalograms, following from the definition (6.6). To compute these, the `cwt()`-function of Matlab has been used [26]: see Listing B.6.

The first result to consider is the sum of sines in Figure 6.4a. Here the same kind of behavior as in the spectrogram is observed: the coefficients are constant in time. In the frequency range two 'bars' are observed, one around 50 Hz and one around 250 Hz. However, the spread of the 50 Hz signal is much smaller than the spread of the 250 Hz signal. This is the result of the different shapes of the Heisenberg boxes: these have a larger frequency variance for larger frequencies.

The same is observed for the quadratic chirp (Figure 6.4b), where also some 'boundary effects' appear at 2 seconds. The source of these effects will be discussed later. The last example (Figure 6.4c) is the wavelet transform of the sum of transient signals. Here we clearly see at $t = 1$ second that for the larger frequencies the resolution in time is better. The start of the 250 Hz signal can be marked at this time, but the end of the 50 Hz signal is not as clear. The 400 Hz part of the signal is harder to distinguish, but the start and end are even sharper. However the frequency spread for this signal is already very large. The most striking difference with the STFT is the absence of the block wave signal in this wavelet transform. This can be explained by a characteristic of the Morlet wavelet; its infinite number of vanishing moments [21]. This is addressed in Section 6.2.1. Note that power density of the STFT example is expressed in decibels, in contrary to this example.

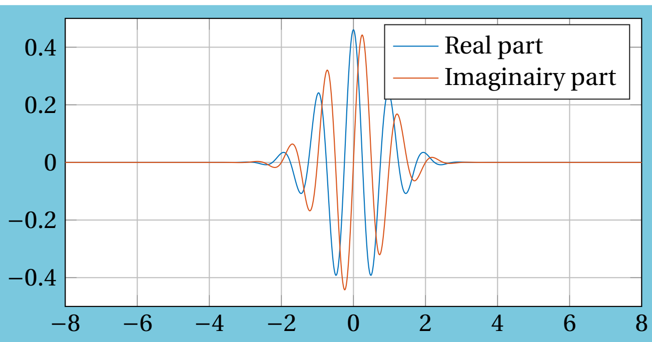


Figure 6.3: The complex Morlet wavelet $\psi_{\text{cm}}(t)$, (6.7) with $B = 1.5$ and $\omega_0 = 1$.

6.2. Wavelet characteristics

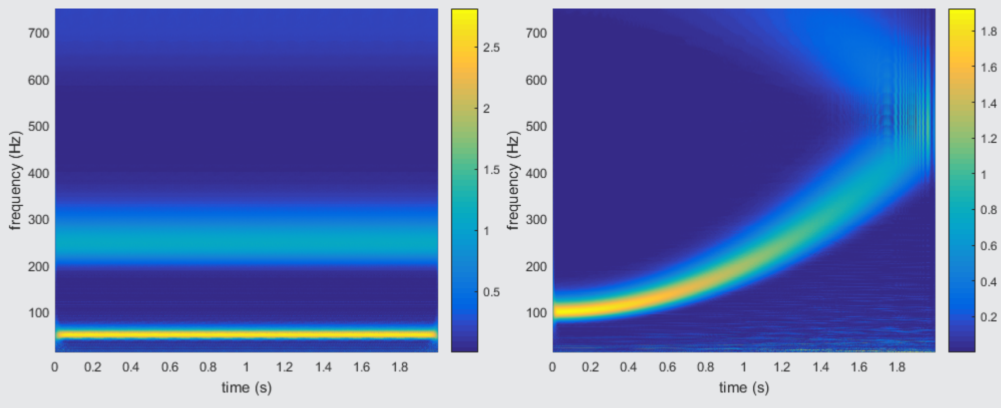
Till thus far we have seen the formal definition of a wavelet (6.1). An other approach will be presented in Section 6.3.2. These wavelets do have the same characteristics. But what characteristics do make a wavelet a good wavelet? Sweldens [37] gives a very general description of wavelets to start with:

Wavelets are building blocks that can quickly decorrelate data.

This sentence contains the three main characteristics of a wavelets. A wavelet is a *building block*, the first characteristic, for a general function or time signal. This is consistent with the MRA approach: the mathematical description of a basis fits this building block description.

The second characteristic is the *power to decorrelate*. This has not been addressed much yet, but the main message is that we can get an accurate approximation of the original signal $x(t)$ by only using a small fraction of the wavelet coefficients. Therefore the wavelet in a way has to resemble the data we want to represent, this leads to three core properties [37]:

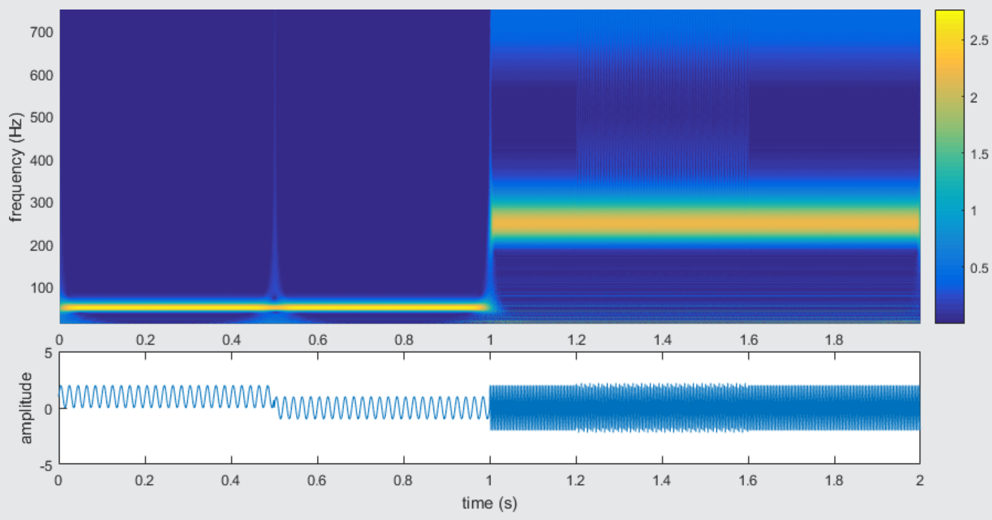
- A wavelet has to have compact support to ensure <u>localization</u> in time;



(a) The wavelet transform of the signal $x(t) = \sin(100\pi t) + \sin(500\pi t)$.

(b) The WT of the chirp signal, described in Example 4.7.

Figure 6.4: Scalogram examples



(c) The WT of the signal from Figure 4.5, with 10 times larger frequencies.

Figure 6.4: The continuous wavelet transform (scalogram) of signals from previous examples.

- a wavelet has to be smooth, which results in decay towards high frequencies, to also have localization in frequency. This localization in frequency is referred to as selectivity;
- and a wavelet has to have vanishing moments, which results in decay toward low frequencies.

The last characteristic is hidden in the word *quickly*. Sweldens [37] decribes that we want to switch between the original representation and the wavelet representation of some data in a time proportional to the size of the data. This characteristic of course is very important to ensure the applicability of wavelets. At last there is a choice between orthogonal and biorthogonal wavelets, this is further elaborated in Section 6.3.5.

6.2.1. Vanishing moments

In the last section three core properties have been discussed. The last one was the property of 'having vanishing moments'. This subsection will shine some light on these moments. The name moment comes from the probabilistic idea of a moment generating function. For a random variable, the n^{th} derivative, evaluated at 0 gives the n^{th} moment of this variable [31] . The best known moments are the first moment (expected value) and the second moment (expected value of the square of the random variable).

One way to differentiate various wavelet is by their vanishing moments. In wavelet theory the moments are a number to estimate the rate of decay of a wavelet $\psi(t)$ [20]. The rate of decay for a general function $f(t)$ can be estimated by the formal integral

$$\int_{-\infty}^{\infty} t^k f(t)\ \mathrm{d}t.$$

Here the parameter k indicates the rate of decay. If we for example consider the function $f(t) = \cos t/t^2$, then we know that the integral converges to 0 for $k = 0$ and $k = 1$. For $k = 2$ this integral converges to π. For a general wavelet $\psi(t)$ we say it has *p vanishing moments* if [20]

$$\int_{-\infty}^{\infty} t^k \psi(t)\ \mathrm{d}t = 0 \qquad \text{for } 0 \le k < p \in \mathbb{N}. \tag{6.8}$$

This leads to descriptions of moments for both discrete and continuous wavelets. The continuous moments μ_k and ν_k are defined by the integral (6.8) as

$$\mu_k = \int_{-\infty}^{\infty} t^k \phi(t)\ \mathrm{d}t, \qquad \nu_k = \int_{-\infty}^{\infty} t^k \psi(t)\ \mathrm{d}t.$$

In the Fourier domain, this can be checked by evaluating the transform of the derivatives of the functions. A function has p vanishing moments if the first $p-1$ derivatives of its Fourier transform are zero at $\omega = 0$ [20]. This leads to the analogous expression [18]:

$$\mu_k = 2\pi i^k \frac{\mathrm{d}^k\Phi}{\mathrm{d}\omega^k}(0), \qquad \nu_k = 2\pi i^k \frac{\mathrm{d}^k\Psi}{\mathrm{d}\omega^k}(0).$$

Note that the continuous moment μ_0 is not determined for the refinement equation [18], it depends on the scaling of ϕ. μ_0 can be picked arbitrarily for any given ϕ, but for a biorthogonal pair the following relationship has to hold: $\overline{\tilde{\mu}_0}\mu_0 = 1$. However, when μ_0 has been chosen, all other continuous moments are uniquely defined.

The definition of the moments are clear. A moment is called vanishing if it is equal to 0. The advantage of the vanishing moments is the possibility to write the wavelet as a low pass filter. This simplifies the wavelet design process [20], which will be addressed in Section 6.5. The higher the number of vanishing moments, the more complex a wavelet is and therefore it is more accurate in the representation of a complex signal. The disadvantage of a high number of vanishing moments is that it results in a longer support. As the number of vanishing moments increases, polynomials up to that order will not be identified by the wavelet.

6.2.2. Localization and selectivity

So a wavelet with one vanishing moment, $p = 1$, cannot identify constant signals, but it does identify linear, quadratic, etc. signals. In an analysis with a wavelet with two vanishing moments, linear signals cannot be identified anymore. In the Fourier transform, for each analyzing function one frequency is addressed. In the wavelet transform a range of frequencies is encompassed in one analyzing function $\psi(t)$. So to analyze a lower range of frequencies, you need to be more *selective*, earlier referred to as 'localization in frequency'. However if we link this to the uncertainty principle, we note that the more selective a wavelet is, the less compact support it has. This links the selectivity of the wavelet to its number of vanishing moments, explaining the behaviour in Example 6.1.

6.2.3. Regularity and decay

The last related aspect of wavelets is its *regularity*. Wavelet with low regularity create jagged representations of the signal which is analyzed, wavelets with high regularity result in smoother representation of the functions. The more vanishing moments a wavelet has, the higher the regularity of the wavelet. However, the regularity of a wavelet increases linearly with the support width [11]. The application of the wavelet analysis lets us assess wavelet for different properties. Compact supported orthonormal wavelets are suitable for sparse representations of large matrices. Therefore the number of vanishing moments is far more important than the regularity [11]. For compression, smoothness is important to observe as little of the compression as possible, placing high regularity over number of vanishing moments [11].

6.3. MRA - Multi Resolution Analysis

The *multi resolution analysis* or *multi resolution approximation* (MRA) is another way of defining wavelets. In the first part of this chapter the CWT was defined from the theory of the STFT. The *discrete wavelet transfrom* (DWT) theory can be approached from the definition of the CWT, with addition of the notion of frames in Hilbert spaces [4, 11, 21]. Here however, the MRA approach of the DWT will be followed, which has two main advantages. The first advantage is that the MRA theory is discrete to begin with, resulting in a more natural derivation of the DWT, which is easier to implement as a computer algorithm [4, 18, 27]. Secondly, the MRA structure allows for convenient, fast and exact calculation of wavelet coefficients by providing a recursion relation, for both the discrete and the continuous case. This recursion relation is a relation between scaling coefficients at a given scale 2^{-n-1} and the scaling and wavelet coefficients at the next coarser scale 2^{-n} [43]. This section starts with the definition of a refinable function and an MRA and it ends with the definition of the DWT. This is followed by the covering of biorthogonal MRAs, in addition to the orthogonal ones discussed here. Through out this section the example of the Haar wavelet will be used. First we start with a *refinable function*, which is a function $\phi : \mathbb{R} \to \mathbb{C}$ which satisfies a two-scale refinement equation, or *recursion relation* of the form

$$\phi(t) = \sqrt{2} \sum_{k=k_0}^{k_1} h_k \phi(2t-k). \tag{6.9}$$

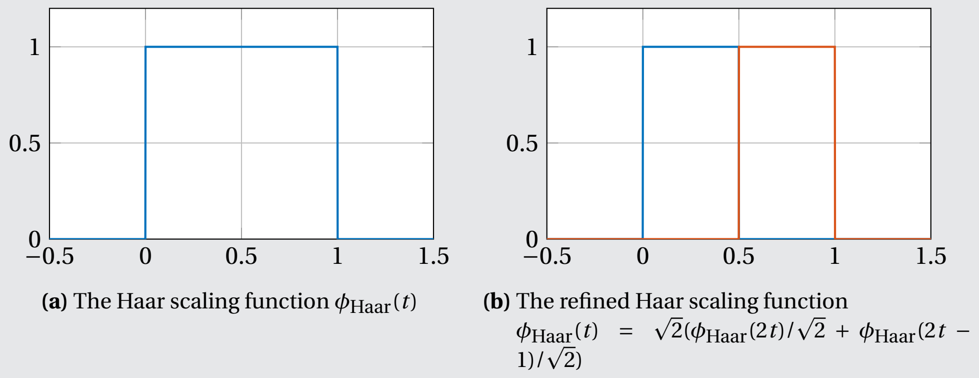
The $h_k \in \mathbb{C}$ are known as the *recursion coefficients*. A refinable function ϕ is called orthogonal if for $k \in \mathbb{Z}$ [4, 18]

$$\langle \phi(t), \phi(t-k) \rangle = \delta_{0k} \qquad \text{holds.} \tag{6.10}$$

Example 6.2 (Haar function) An example of such an orthogonal refinable function is the *Haar function*, defined as

$$\phi_{\text{Haar}}(t) := \mathbb{1}_{[0,1]} = \begin{cases} 1 & 0 \le t \le 1 \\ 0 & \text{elsewhere} \end{cases}. \tag{6.11}$$

The Haar function is orthogonal and refinable with $h_0 = h_1 = 1/\sqrt{2}$, as shown in Figure 6.5.



(a) The Haar scaling function $\phi_{\text{Haar}}(t)$ **(b)** The refined Haar scaling function $\phi_{\text{Haar}}(t) = \sqrt{2}(\phi_{\text{Haar}}(2t)/\sqrt{2} + \phi_{\text{Haar}}(2t-1)/\sqrt{2})$

Figure 6.5: Example: the Haar scaling function.

The set $\{V_j\}_{j\in\mathbb{Z}}$ is called a orthogonal *multi resolution analysis* (MRA) of L^2, where V_j, $j \in \mathbb{Z}$ is a sequence of subspaces of L^2, if it complies to six conditions [18]:

$$V_j \subset V_{j+1} \qquad \text{(nested subsets)} \tag{6.12a}$$

$$\cup_{j\in\mathbb{Z}} V_j = L^2 \qquad \text{(density axiom)} \tag{6.12b}$$

$$\cap_{j\in\mathbb{Z}} V_j = \{0\} \qquad \text{(separation axiom)} \tag{6.12c}$$

$$f(t) \in V_n \iff f(2x) \in f(2t) \in V_{n+1} \; \forall n \in \mathbb{Z} \qquad \text{(scaling property)} \tag{6.12d}$$

$$f(t) \in V_n \iff f(2x) \in f(t-2^{-n}k) \in V_n \; \forall n,k \in \mathbb{Z} \qquad \text{(scaling property)} \tag{6.12e}$$

$$\exists \phi(t) \in L^2 \text{such that } \{\phi(t-k) : k \in \mathbb{Z}\} \text{ forms an orthogonal basis of } V_0 \qquad \text{(scaling function)} \tag{6.12f}$$

There also exist MRAs built on non orthogonal scaling functions, these will be discussed later (Section 6.3.5). Orthonormalizing an existing scaling function is possible, but the resulting new ϕ often does not have compact support anymore, losing its function for practical applications [18]. The fourth condition (6.12d) expresses the main property of an MRA: each subspace V_n consists of the functions in V_0 compressed by a factor 2^n, therefore spanning V_0. From this can be concluded that a stable basis of V_n is given by $\{\phi_{nk}(t) : n \in \mathbb{Z}\}$, where

$$\phi_{nk}(t) = 2^{n/2}\phi(2^n t - k), \qquad k \in \mathbb{N}. \tag{6.13}$$

The last condition (6.12f) implies that any function $f \in V_0$ can be written uniquely as a sum of coefficients f_k multiplied with a scaling function

$$f(t) = \sum_{k\in\mathbb{Z}} \overline{f_k}\phi(t-k), \tag{6.14}$$

converging in L^2. The essential characteristic of the MRA is that $\phi(t) \in V_0$ can be written in the terms of the basis of V_1 as

$$\phi(t) = \sum_k h_k \phi_{1k}(t) = \sqrt{2}\sum_k h_k \phi(2t-k), \tag{6.15}$$

for some coefficients h_k. This is called the *refinement equation*. From this follows that ϕ is a refinable function, for ϕ complies to the recursion relation (6.9). This refinement equation can be an infinite sum, but for now we will continue assuming a finite sum. The orthogonality condition (6.10) in this form becomes

$$\sum_k h_k \overline{h}_{k-2\ell} = \delta_{0\ell}. \tag{6.16}$$

The orthogonal projection onto the subspace V_n, denoted with P_n, of an arbitrary function $f \in L^2$ is given by

$$P_n f = \sum_k \langle f, \phi_{nk}\rangle \phi_{nk}. \tag{6.17}$$

Note that the projection $P_n f$ cannot represent details smaller than 2^{-n}. Therefore we say that functions in V_n have *resolution* or *scale* 2^{-n}. An MRA provides a sequence of approximations $P_n f$ of increasing accuracy to a given function f.

Example 6.3 (Orthogonal projection) Lets continue with the Haar example. We are going to approximate the function $\cos(t)$ on the interval $[0,10]$. We start with the function ϕ_{Haar} (6.11), then

$$P_0 f(t) = \sum_k \langle f, \phi_{0k}\rangle \phi_{0k} = \sum_k \langle \cos t, \phi_{\text{Haar}}(t-k)\rangle \cdot \phi_{\text{Haar}}(t-k).$$

And so the same can be applied to $P_1 f(t)$:

$$P_1 f(t) = \sum_k \langle f, \phi_{1k}\rangle \phi_{1k} = \sum_k \langle \cos t, \sqrt{2}\phi_{\text{Haar}}(2t-k)\rangle \cdot \sqrt{2}\phi_{\text{Haar}}(2t-k).$$

The result of this approximation is given in Figure 6.6. Note that the Haar function produces an orthogonal MRA.

6.3.1. Fine details

Example 6.4 (Orthogonal components) In Example 6.3 we have seen that we can approximate a function using the scaling function ϕ. The representation of, for instance, a function $x(t) \in V_1$ can be done by the space V_1, but also by all spaces V_j, $j > 1$ [27]. This representation however is not very efficient, because more parameters than necessary are needed. How do you use less parameters? Note we used the functions in V_1 only to represent the part of $x(t)$ which could not be represented by V_0. If we only use the difference between the spaces V_1 and V_0 to describe $x(t)$, less parameters are needed. Therefore the space W_0 will be explicitly designed. This space W_0 is in V_1, but not in V_0 and therefore is called the *orthogonal component* of V_0 in V_1. A more mathematical description follows.

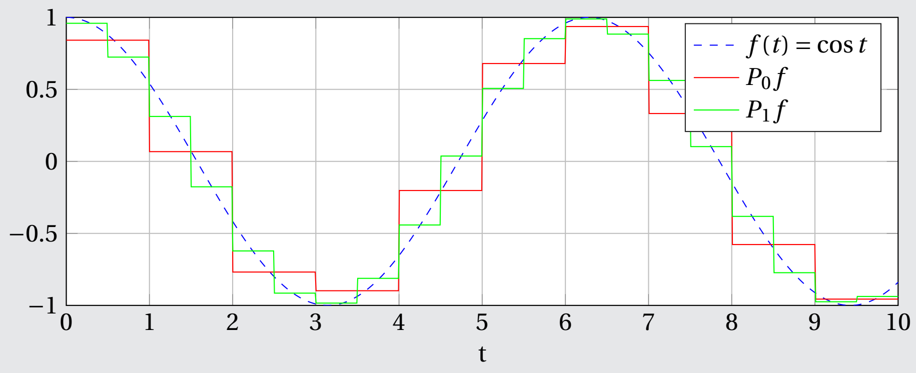


Figure 6.6: Orthogonal projection of the function $\cos t$ using the Haar function (6.11).

When this difference between two approximations at different levels is considered, the applicability of the MRA rises. The difference between two levels of resolution 2^{-n} and 2^{-n-1} is also called the *fine detail at resolution* 2^{-n}, denoted as Q_n:

$$Q_n f = P_{n+1} f - P_n f. \tag{6.18}$$

Note that Q_n is also an orthogonal projection [18] and that its range W_n is orthogonal to V_n, so the direct sum of the function space V_n and W_n is V_{n+1}:

$$V_n \oplus W_n = V_{n+1}.$$

This is the final step that brings us to another definition of the wavelet, approached discretely instead of continuous (6.4). Wavelets are an element of an orthogonal MRA. For any orthogonal MRA with scaling function ϕ [18]

$$\bigoplus_n W_n = L^2 \text{ dense} \tag{6.19a}$$

$$W_k \perp W_n \text{ if } k \neq n \tag{6.19b}$$

$$f(t) \in W_n \iff f(2t) \in W_{n+1} \forall n \in \mathbb{Z} \tag{6.19c}$$

$$f(t) \in W_n \iff f(t - 2^{-n}k) \in W_n \forall n, k \in \mathbb{Z} \tag{6.19d}$$

$$\exists \psi \in L^2, \text{ called a wavelet, such that } \{\phi(t-k) : k \in \mathbb{Z}\} \text{ forms an orthogonal basis of } W_0$$

$$\text{and } \{\psi_{nk} : n, k \in \mathbb{Z}\} \text{ forms a stable basis of } L^2. \tag{6.19e}$$

Since $\psi \in V_1$, it can be represented as

$$\psi(t) = \sum_k g_k \phi_{1k} = \sqrt{2} \sum_k g_k \phi(2x - k), \text{ with } g_k = (-1)^k h_{N-k}, \text{ with } N \text{ odd.} \tag{6.19f}$$

Here ψ is known as the *mother wavelet*. In terms of the wavelet function, the projection Q_n is given, in the same way as P_n (6.17):

$$Q_n f = \sum_k \langle f, \psi_{nk} \rangle \psi_{nk}. \tag{6.20}$$

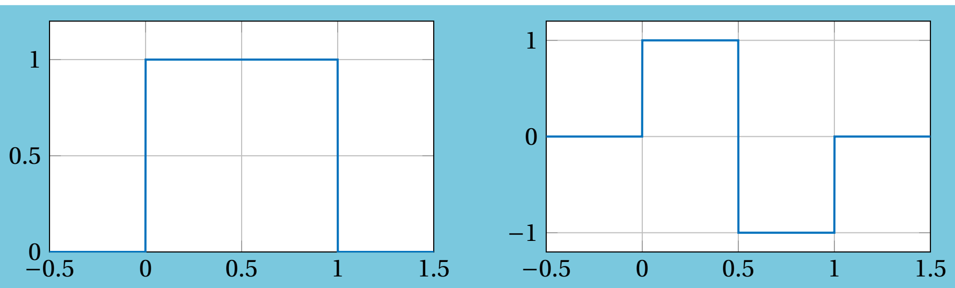
This projection is the final step to the *discrete wavelet transform* DWT.

The Haar wavelet Now we will apply this to the example of the Haar function from Example 6.2, to introduce our first wavelet build using MRA: the Haar wavelet (see Figure 6.7). The coefficients g_k from (6.19f) to create the Haar wavelet are $\{g_0, g_1\} = \{h_1, -h_0\} = \{1/\sqrt{2}, -1/\sqrt{2}\}$, resulting in the function $\psi_{\text{Haar}}(t) = \mathbb{1}_{[0;0.5)}(t) - \mathbb{1}_{[0.5;1)}(t)$.

6.3.2. DWT - Discrete Wavelet Transformation

We have seen that given a function $f \in L^2$ we can represent it as a complete decomposition in terms of detail at all levels:

$$f = \sum_{k=-\infty}^{\infty} Q_k f.$$



(a) The Haar scaling function $\phi_{\text{Haar}}(t)$ **(b)** The Haar wavelet $\psi_{\text{Haar}}(t)$

Figure 6.7: The Haar scaling function and wavelet.

As an alternative one can choose to start at a level ℓ and use the approximation at resolution $2^{-\ell}$ together with the detail at finer resolution, to decompose f as:

$$f = P_\ell f + \sum_{k=\ell}^{\infty} Q_k f.$$

An infinite sum is not practical applicable, so the sum is reduced to a finite sum: therefore we assume $f \in V_n$ for some $n > \ell$. Then the *discrete wavelet transform* (DWT) is described by

$$f = P_n f = P_\ell f + \sum_{k=\ell}^{n-1} Q_k f. \tag{6.21}$$

The DWT approach is similar to the CWT approach, except of using continuous scale a and translation b, these are chosen discretely as scale n and translation k. The mother wavelet $\psi(t)$ is chosen and the daughter wavelets are

$$\psi_{nk}(t) = 2^{n/2}\psi(2^n t - k). \tag{6.22}$$

The nk Heisenberg box has size $2^{-n}\sigma_t \times 2^n \sigma_\omega$, with different spacings for different frequencies, as for the continuous case. The inverse operation of the DWT will be discussed in Section 6.4.

Example 6.5 (DWT) This is the last time the Haar example will be discussed. For the Haar example $P_0 f$ and $P_1 f$ have been computed, following the DWT (6.21) we should find that $P_1 f = P_0 f + Q_0 f$. Therefore we use (6.20) to find

$$Q_0 f = \sum_k \langle f, \psi_{0k}\rangle \psi_{0k} = \sum_k \langle \cos t, \mathbb{1}_{[0;0.5)}(t-k) - \mathbb{1}_{[0.5;1)}(t-k)\rangle \cdot (\mathbb{1}_{[0;0.5)}(t-k) - \mathbb{1}_{[0.5;1)}(t-k)).$$

The result is shown in Figure 6.8.

6.3.3. Requirements for wavelet transform

As for the Fourier transform, the integral defining the coefficients should converge. So, just like for the Fourier transform, only functions in the L^2 space are theoretically suitable for wavelet transformation [21]. For discrete signals some requirements will be discussed in Section 6.4. However effects such as the Gibbs effect are not present in the wavelet transform. This makes the wavelet transform much more suitable to use for signals with discontinuities. The Fourier transform can also show unwanted effects in the reconstruction of non periodic functions, for instance a linear function. The reconstruction of the wavelet transform for both non periodic as periodic functions is close to perfect [47]. Because the wavelet transform (up to a certain scale) only catches local effect, the wavelet transform is also better applicable to non stationary signals.

The downside of the wavelet transform is the time-frequency representation in the form of the scalogram. The scalograms shown in Figure 6.4 are hard to read and do not necessarily contain better information than the spectrograms of these signals (Figure 5.1, 5.3c and 5.3b.) The discrete version of the scalogram, which will be presented in Figure 6.11b is even harder to read. There is not necessarily a one to one coupling of wavelet scales to frequency. When complex wavelets are used this is determined most easy. Because of these

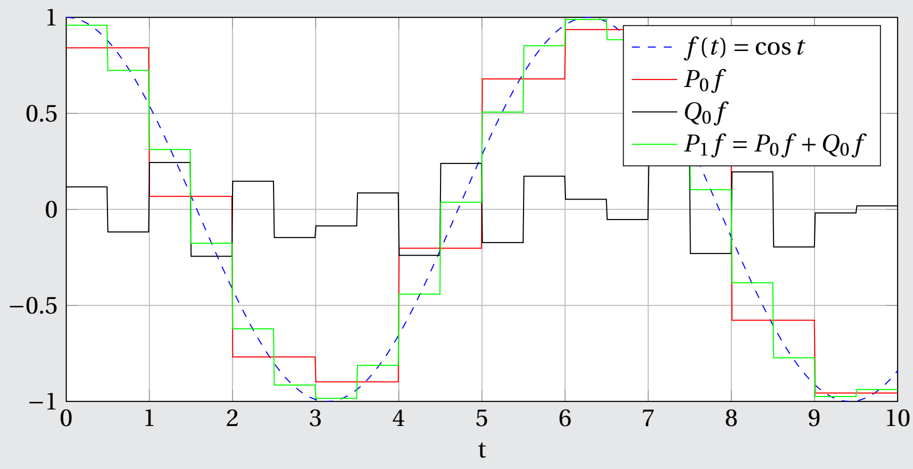


Figure 6.8: Discrete wavelet decomposition of $\cos t$ using the Haar function (6.11) and Haar wavelet.

downsides the scalogram is not much used in time-frequency domain analysis; especially the discrete version is almost never used.

From CWT to DWT

As mentioned before, there are two ways to derive the DWT. Following the MRA theory, we have found an expression for the (discrete) wavelet coefficient belonging to wavelet ψ_{nk} (from (6.20)): $\langle f, \psi_{nk}\rangle$. The definition of the inner product on L^2 in (1.1) shows the similarity with the continuous wavelet coefficient from (6.4). The CWT and DWT coefficients are defined through the same integral. Therefore a large set of functions is suitable for both continuous and discrete wavelet transform. However the most are better suited for one or the other; an overview will be given in Section 6.5. The DWT cannot be derived directly from the CWT expression (6.4), because the set of discrete scales and translations cannot be chosen arbitrarily. The justification of this choice can be made through the notion of frames from the Hilbert space theory [4, 11, 21]. This a very theoretical approach, where the MRA approach is less theoretical. They however result in the same expression for the DWT. A mayor advantage of the MRA approach is that an efficient algorithm of the DWT is easily derived, for both orthogonal and non orthogonal wavelets (see Section 6.3.5). This algorithm will be explained to the reader in Section 6.4.

6.3.4. The design equations

The Fourier transforms of the scaling function and the wavelet are indispensable in the design of wavelets. Here they are shortly reviewed. $H(\omega)$ and $G(\omega)$ are known as the *symbol* of respectively the refinable function and the wavelet[18]:

$$\Phi(\omega) = \mathfrak{F}\phi(\omega) = H(\omega/2)\Phi(\omega/2), \qquad \text{with } H(\omega) = \frac{1}{\sqrt{2}}\sum_k h_k e^{-ik\omega}, \tag{6.23}$$

$$\Psi(\omega) = \mathfrak{F}\psi(\omega) = G(\omega/2)\Psi(\omega/2), \qquad \text{with } G(\omega) = \frac{1}{\sqrt{2}}\sum_k g_k e^{-ik\omega}. \tag{6.24}$$

The orthogonality condition (6.16) in the Fourier domain becomes

$$|H(\omega)|^2 + |H(\omega+\pi)|^2 = 1. \tag{6.25}$$

Relationship (6.23) can be substituted recursively, to find the formal limit

$$\Phi(\omega) = \left[\prod_{k=1}^{\infty} H(2^{-k}\omega)\right]\Phi(0). \tag{6.26}$$

This approach is effective in the creation of wavelets. This relation can also be used to calculate the refinement equation corresponding to a certain low pass filter [20]. If convergence of this infinite product is assumed, this expression provides a way to compute $\phi(t)$ theoretically. $\Phi(0)$ can be chosen arbitrarily: solutions of the refinement equation (6.15) are only defined up to a constant factor. Choose $\Phi(0) \neq 0$ to find solutions other than $\phi(t) = 0$. Multiples of the solution of the equation are also solutions to the equation.

Cascade algorithm

The *cascade algorithm* is a more suitable way than (6.26) to approximate point values of $\phi(t)$ [18]. This algorithm applies a fixed point iteration applied to the refinement equation. Starting by choosing a suitable scaling function $\phi^{(0)}(t)$, and define

$$\phi^{(n)}(t) = \sqrt{2}\sum_k h_k \phi^{(n-1)}(2t-k), \tag{6.27}$$

which will converge in many cases.

6.3.5. Biorthogonal wavelets

In the derivation of the DWT we used the existence of orthogonal MRAs. These orthogonal MRAs however are not very common [18] and therefore the *biorthogonal* MRA will be discussed (sometimes referred to as *semi orthogonal*). We first start with an example from linear algebra, to explain the concept of biorthogonality.

Example 6.6 (Biorthogonal system) Consider two independent vectors $\boldsymbol{b}_1$ and $\boldsymbol{b}_2 \in \mathbb{R}^2$. By independence, $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$ are a basis for $\mathbb{R}$. If $\boldsymbol{b}_1 \perp \boldsymbol{b}_2$, this basis is called orthogonal, if also both vectors are unit vectors, the basis is called orthonormal. Any vector $\boldsymbol{x} \in \mathbb{R}^2$ can be written as $\boldsymbol{x} = \alpha\boldsymbol{b}_1 + \beta\boldsymbol{b}_2$. If we choose $B = [\boldsymbol{b}_1, \boldsymbol{b}_2]$, then we can solve for the coefficients:

$$\boldsymbol{x} = B\begin{bmatrix}\alpha\\ \beta\end{bmatrix} = B\boldsymbol{c} \qquad \Rightarrow \qquad \boldsymbol{c} = B^{-1}\boldsymbol{x}.$$

This however is not as easily solved for a nonorthogonal set as for an orthogonal set of basis vectors. Therefore the *dual* base $\{\tilde{\boldsymbol{b}}_1, \tilde{\boldsymbol{b}}_1\}$ is introduced. These vectors comply to [27]

$$\langle \boldsymbol{b}_1, \tilde{\boldsymbol{b}}_1\rangle = 1, \qquad \langle \boldsymbol{b}_2, \tilde{\boldsymbol{b}}_2\rangle = 1, \qquad \langle \boldsymbol{b}_2, \tilde{\boldsymbol{b}}_1\rangle = 0, \qquad \langle \boldsymbol{b}_1, \tilde{\boldsymbol{b}}_2\rangle = 0,$$

such that we can use the relation $\boldsymbol{c} = B^{-1}\boldsymbol{x} = \tilde{B}\boldsymbol{x}$, to determine the coefficients α and β. This matrix $\tilde{B}$ is chosen $\tilde{B} = \left[\tilde{\boldsymbol{b}}_1, \tilde{\boldsymbol{b}}_1\right]^{\mathsf{T}}$ such that

$$\boldsymbol{x} = \left(\tilde{\boldsymbol{b}}_1^{\mathsf{T}}\boldsymbol{x}\right)\boldsymbol{b}_1 + \left(\tilde{\boldsymbol{b}}_2^{\mathsf{T}}\boldsymbol{x}\right)\boldsymbol{b}_2. \tag{6.28}$$

E.g. we choose two non orthogonal vectors, spanning $\mathbb{R}^2$ and a vector $\boldsymbol{x}$:

$$\boldsymbol{b}_1 = \begin{bmatrix}1\\0\end{bmatrix}, \qquad \boldsymbol{b}_2 = \begin{bmatrix}1/2\\ \sqrt{3}/2\end{bmatrix} \qquad \boldsymbol{x} = \begin{bmatrix}1\\1\end{bmatrix}.$$

The dual base for the given $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$ is given by

$$\tilde{\boldsymbol{b}}_1 = \begin{bmatrix}1\\ -1/\sqrt{3}\end{bmatrix} \qquad \text{and } \tilde{\boldsymbol{b}}_2 = \begin{bmatrix}0\\ 2/\sqrt{3}\end{bmatrix}.$$

Then

$$\tilde{\boldsymbol{b}}_1^{\mathsf{T}}\boldsymbol{x} = 1 - 1/\sqrt{3}, \qquad \tilde{\boldsymbol{b}}_2^{\mathsf{T}}\boldsymbol{x} = 2/\sqrt{3}, \qquad \Rightarrow \quad (1-1/\sqrt{3})\begin{bmatrix}1\\0\end{bmatrix} + 2/\sqrt{3}\begin{bmatrix}1/2\\ \sqrt{3}/2\end{bmatrix} = \begin{bmatrix}1\\1\end{bmatrix} = \boldsymbol{x}.$$

The sets $\{\boldsymbol{b}_1, \boldsymbol{b}_2\}$ and $\{\tilde{\boldsymbol{b}}_1, \tilde{\boldsymbol{b}}_2\}$ are a biorthogonal system of $\mathbb{R}^2$.

As the word *bi* in biorthogonal implies, the biorthogonal MRA has not one but two refinable functions as a basis. Refinable functions in general are relatively easy to find, but a lot of them do not result in orthogonal MRAs, so the orthogonality conditions will be replaced by milder biorthogonality conditions [18]. Two refinable functions ϕ and $\tilde{\phi}$ are called *biorthogonal* if

$$\langle \phi(x), \tilde{\phi}(x-k)\rangle = \delta_{0k}.$$

$\tilde{\phi}$ then is referred to as the *dual* of ϕ. This dual however is not unique [18], dual lifting, a process discussed later, will produce numerous other duals of the same ϕ. These two scaling functions define two MRAs: $\{V_n\}_{n\in\mathbb{Z}}$ and $\{\tilde{V}_n\}_{n\in\mathbb{Z}}$. Then the construction of the projections follows the same line of reasoning as in Example 6.6: there are two projections, P_n (6.29a) and $\tilde{P}_n$ (6.29b) to project a function from L^2 to $\{V_n\}_{n\in\mathbb{Z}}$ and $\{\tilde{V}_n\}_{n\in\mathbb{Z}}$ respectively.

The projections Q_n (6.29c) and $\tilde{Q}_n$ (6.29d) are defined as before, spanning the spaces $\{W_n\}_{n\in\mathbb{Z}}$ and $\{\tilde{W}_n\}_{n\in\mathbb{Z}}$ respectively. The space W_n is now orthogonal to $\tilde{V}_n$, such that the fine detail relation (6.18) still holds. The same hold for $\tilde{W}_n$ and V_n. Now note that $V_n \oplus W_n = V_{n+1}$ still holds, but now as a nonorthogonal direct sum instead of an orthogonal one. Keinert [18] states that finding wavelet functions ψ and $\tilde{\psi}$ which span the spaces W_n and $\tilde{W}_n$ is not that hard, but stability is not guaranteed. How to find these functions is further elaborated in Section 6.5.

$$P_n f = \sum_k \langle f, \tilde{\phi}_{nk}\rangle \phi_{nk}, \tag{6.29a}$$

$$\tilde{P}_n f = \sum_k \langle f, \phi_{nk}\rangle \tilde{\phi}_{nk}, \tag{6.29b}$$

$$Q_n f = P_{n+1} f - P_n f, \tag{6.29c}$$

$$\tilde{Q}_n f = \tilde{P}_{n+1} f - \tilde{P}_n f. \tag{6.29d}$$

When an orthogonal wavelet is used for decomposition, the representation of the signal is the most compact: the number of convolutions at a scale is proportional to the size of a scale [40]. This will result in a relative sparse representation of the signal. This is characteristic is very desirable in signal compression [18]. An aperiodic shift in time series produces a different wavelet spectrum, this is often not beneficial in time series analysis. When a biorthogonal wavelet is used, the large scales are highly redundant, the wavelet coefficients at adjacent times are highly correlated. This makes the biorthogonal wavelets better applicable to time series analysis where smooth, continuous variations in wavelet coefficients are expected [40].

6.3.6. Discrete moments

In Section 6.2 the importance of the number of vanishing moments for the wavelet transform has been addressed. The discrete wavelet do have vanishing moments, just like the continuous case. They however can be defined by their coefficients: the kth discrete moment of the refinement function ϕ and the wavelet ψ are defined by their coefficients h_k and g_k. The m denotes the moment of the refinable function, the n the moment of the wavelet [18].

$$m_k = \frac{1}{\sqrt{2}}\sum_\ell \ell^k h_\ell, \qquad n_k = \frac{1}{\sqrt{2}}\sum_\ell \ell^k g_\ell,$$

$$m_k = i^k \frac{\mathrm{d}^k h}{\mathrm{d}\omega^k}(0), \qquad n_k = i^k \frac{\mathrm{d}^k g}{\mathrm{d}\omega^k}(0).$$

If in particular $m_0 = h(0) = 1$, the zeroth moment of a refinement function is 1. These discrete moments are uniquely defined and easy to calculate. They can be computed using the relation between the discrete and continuous moments:

$$\mu_k = 2^{-k}\sum_{p=0}^{k}\binom{k}{t} m_{k-p}\mu_p, \qquad \nu_k = 2^{-k}\sum_{p=0}^{k}\binom{k}{t} n_{k-p}\mu_p.$$

6.4. Discrete wavelet transform algorithmic

In this section of the chapter the two or three steps of the algorithm performing the DWT are explained. Some different formulations of the DWT algorithm are discussed, which are essential to the building of wavelets. The resulting algorithm needs $\mathcal{O}(N\log_2 N)$ operations, which as fast as the DFT, however it is asymptotically faster than the STFT algorithm, using $\mathcal{O}(N^2\log_2 N)$ operations. The two or three steps to do a complete DWT for a 1D signal of finite length are [18]:

1. Optional: preprocessing of the signal (Section 6.4.3);
2. Handling the boundary conditions (Section 6.4.4); and
3. Applying the algorithm (Section 6.4.1).

Finally there are different ways of formulation and implementation of the algorithm. These will be discussed at the end of this section. This section is started with the final step of the DWT: the algorithm. Note we assume the use of biorthogonal wavelets, if one uses a orthogonal wavelets the tildes in this explanation can be dropped. The DWT is based on the decomposition of the space V_n:

$$V_n = V_\ell \oplus W_\ell \oplus \ldots \oplus W_{n-1}.$$

6.4.1. The algorithm

We start with a function $s \in V_n$ (or signal), which by the theory can be represented by its coefficient vector $\boldsymbol{s}_n = \{s_{nk}\}_{k=1,\ldots,N}$:

$$s(t) = \sum_k \overline{s}_{nk}\phi_{nk}(t). \tag{6.30}$$

The function can also be expanded into two parts:

$$s(t) = \sum_k \overline{s}_{\ell k}\phi_{\ell k}(t) + \sum_{j=\ell}^{n-1}\sum_k \overline{d}_{jk}\psi_{jk}(t).$$

The notations s and d originate from the Haar wavelet, where s denotes the sum and d the difference. For regular wavelets, it is easier to remind s as the smooth part and d as the (fine) detail [18]. The complex conjugate notation of the coefficients comes from the multiwavelet theory. The DWT and inverse DWT (IDWT) convert the coefficients s_{nk} to $s_{\ell k}$ and d_{jk}, $j = \ell, \ldots, n-1$ and vice versa. Signals consisting of equally spaced samples of the signal s frequently are in the form $s(2^{-n}k)$. The conversion of $s(2^{-n}k)$ to s_{nk} is called *preprocessing*, the reverse processes *postprocessing*. Both are explained later. So the signal s is *decomposed* in its components in V_{n-1} and W_{n-1} by

$$\begin{aligned} s &= P_{n-1}s + Q_{n-1}s = \sum_k \langle s, \tilde{\phi}_{n-1,j}\rangle \phi_{n-1,j} + \sum_k \langle s, \tilde{\psi}_{n-1,j}\rangle \psi_{n-1,j} \\ \Rightarrow \boldsymbol{s}_n &= \sum_k \overline{s}_{n-1,j}\phi_{n-1,j} + \sum_k \overline{d}_{n-1,j}\psi_{n-1,j}. \end{aligned}$$

By this the (discrete) signal s, $\boldsymbol{s}_n$ in vector notation, is *decomposed* in two pieces: $\boldsymbol{s}_{n-1}$ (6.31a) and $\boldsymbol{d}_{n-1}$ (6.31b) [18]. From this the signal can be reconstructed following (6.31c).

$$s_{n-1,j} = \sum_k \tilde{h}_{k-2j}s_{nk}, \tag{6.31a}$$

$$d_{n-1,j} = \sum_k \tilde{g}_{k-2j}s_{nk}, \tag{6.31b}$$

$$s_n k = \sum_j (\overline{h}_{k-2j}s_{n-1,j} + \overline{g}_{k-2j}d_{n-1,j}) \tag{6.31c}$$

Where we define

$$\langle \phi_{n-1,j}, \tilde{\phi}_{nk}\rangle = h_{k-2j}, \qquad \langle \phi_{n-1,j}, \tilde{\psi}_{nk}\rangle = g_{k-2j}, \qquad \langle \tilde{\phi}_{n-1,j}, \phi_{nk}\rangle = \tilde{h}_{k-2j}, \qquad \langle \tilde{\psi}_{n-1,j}, \phi_{nk}\rangle = \tilde{g}_{k-2j}.$$

Convolution implementation

The decomposition step can be written as two discrete convolutions, of computation time $\mathcal{O}(N\log_2 N)$ using the same improvement as the FFT algorithm. These convolutions are[2]

$$((-)\tilde{\boldsymbol{h}} * \boldsymbol{s}_n)_j = \sum_k \tilde{h}_{-(j-k)}s_{nk}, \qquad ((-)\tilde{\boldsymbol{g}} * \boldsymbol{s}_n)_j = \sum_k \tilde{g}_{-(j-k)}s_{nk}, \tag{6.32a}$$

which are followed by downsampling to determine $\boldsymbol{s}_{n-1}$ and $\boldsymbol{d}_{n-1}$ (assume computation time of $\mathcal{O}(1)$):

$$\boldsymbol{s}_{n-1} = (\downarrow 2)((-)\tilde{\boldsymbol{h}} * \boldsymbol{s}_n), \qquad \boldsymbol{d}_{n-1} = (\downarrow 2)((-)\tilde{\boldsymbol{g}} * \boldsymbol{s}_n). \tag{6.32b}$$

[2]For notation, see Section 1.1.

This is only one step of the algorithm, in practice these steps often repeated several times:

$$\begin{aligned} \boldsymbol{s}_n &\to \boldsymbol{s}_{n-1}, \boldsymbol{d}_{n-1} \\ \boldsymbol{s}_{n-1} &\to \boldsymbol{s}_{n-2}, \boldsymbol{d}_{n-2} \\ &\vdots \\ \boldsymbol{s}_{\ell+1} &\to \boldsymbol{s}_{\ell}, \boldsymbol{d}_{\ell}. \end{aligned}$$

Because in every step, the length of the components $\boldsymbol{s}_{n-1}$, $\boldsymbol{d}_{n-1}$ is half the length of the components $\boldsymbol{s}_n$, $\boldsymbol{d}_n$, the algorithm has to compute $\mathcal{O}(N)+\mathcal{O}(N/2)+\mathcal{O}(N/4)+\ldots=\mathcal{O}(N)$ convolutions. This results in $\mathcal{O}(N^2 \log_2 N)$ arithmetic operations to determine the discrete wavelet transform of a signal. The reconstruction of the signal from the DWT is opposite to the decomposition: first upsampling, followed by two convolutions:

$$\boldsymbol{s}_n = \overline{\boldsymbol{h}} * (\uparrow 2)\boldsymbol{s}_{n-1} + \overline{\boldsymbol{g}} * (\uparrow 2)\boldsymbol{d}_{n-1}. \tag{6.32c}$$

Programming the routine

When we start with a signal $\boldsymbol{s}_n$ of length N, the first step produces $\boldsymbol{s}_{n-1}$ and $\boldsymbol{d}_{n-1}$, which are both of length $N/2$. These signals are most of the time stored in the same place as the original signal $\boldsymbol{s}_n$. So the output of the DWT routine after several steps becomes then

$$\begin{bmatrix} \boldsymbol{s}_\ell \\ \boldsymbol{d}_\ell \\ \boldsymbol{d}_{\ell+1} \\ \vdots \\ \boldsymbol{d}_{n-1} \end{bmatrix}.$$

This vector can be stored in the same space as $\boldsymbol{s}_n$ with which the routine started. This representation results in an ugly programmable routine. The *matrix formulation* of the DWT results in a more appealing matrix-vector product notation.

Both the decomposition as the reconstruction can be implemented as infinite matrix-vector products [18]. Here $(\boldsymbol{sd})_n = [\ldots, s_{n,-1}, d_{n,-1}, s_{n,0}, d_{n,0}, s_{n,1}, d_{n,1}, \ldots]^\mathsf{T}$, such that the decomposition step can be written as

$$(\boldsymbol{sd})_{n-1} = \tilde{L}\boldsymbol{s}_n, \qquad \text{with } \tilde{L} = \begin{bmatrix} \cdots & \cdots & \cdots & & \\ \cdots & \tilde{L}_0 & \tilde{L}_1 & \cdots & \\ & \cdots & \tilde{L}_0 & \tilde{L}_1 & \cdots \\ & & \cdots & \cdots & \cdots \end{bmatrix}, \qquad \text{for } \tilde{L}_k = \begin{bmatrix} \tilde{h}_{2k} & \tilde{h}_{2k+1} \\ \tilde{g}_{2k} & \tilde{g}_{2k+1} \end{bmatrix}. \tag{6.33}$$

The reconstruction step can thus be written as

$$\boldsymbol{s}_n = L^*(\boldsymbol{sd})_{n-1},$$

where the perfect reconstruction condition is expressed as $L^*\tilde{L} = I$. The finite, and therefore applicable versions will be derived when the boundaries are discussed.

6.4.2. Filter formulation

The filter formulation of this algorithm will be discussed shortly for it is widely used [26, 27, 38]. In the algorithm the decomposition is given by a convolution and a down sampling. The recomposition is given as a up sampling step follow by a convolution with the same signal. These steps are shown in Figure 6.9. From these figures it is clear that both filters $g[n]$ and $h[n]$ have different properties. The filter $g[n]$ is a high pass filter, whereas the filter $h[n]$ is a low pass filter. The down sampled low pass filter output is then treated as an input for the next stage, passing through the same analyzing filters. From this follows that admissible wavelets are either high pas or band pass filters [27]. This process can be repeated until the desired number of stages is achieved. An example for three stages is found in Figure 6.10. In the recomposition it is clear why only the most coarse smooth part and all fine details are saved. By omitting the finest details, a signal can for instance be compressed. The representation in Figure 6.10 is known as the dyadic implementation. The recomposition filters are known as *interpolating filters*. If wavelet packets are used, not only the smooth part is fed into the decomposition again, but also the detail parts are decomposed. If not all detail parts are decomposed, this can lead to interesting distributions of the time frequency plane (such as Figure 6.13). They are almost nowhere used to display data, but have different applications in mostly compression and noise reduction [27].

Wavelets can be designed from this filter perspective, where $h[n]$ can be seen as high pass filter and $g[n]$ as a low pass filter. This design perspective will not be addressed much in this literature thesis. However, note that the choice of filter $h[n]$ has impact on the vanishing moments, the regularity and the decay of the accompanied wavelet. The choice of the coefficients of $h[n]$ that lead to maximum regularity differs from the choice of a maximum number of vanishing moments [11]. The filter implementation of the DWT algorithm is also known as the Fast Wavelet Transform (FWT) and was first proposed by Mallat in 1988. It finds it strength in using the Fourier transform of the signals and the convolution property (4.5) [21]. This will be elaborated further in Section 6.4.5.

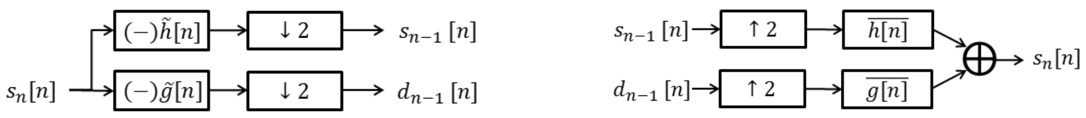
6.4.3. Pre- and postprocessing

As mentioned in the begin of Section 6.4, the data, mostly in the form of a signal $s(2^{-n}k)$, has to be converted to the coefficients s_{nk} from (6.30). Keinert [18] discusses a few options to find the coefficients s_{nk} (assume the signal $s(t)$ is real for simplification):

- Use for the coefficients their exact values: $s_{nk} = \int_{-\infty}^{\infty} s(t)\tilde{\phi}_{nk}(t)\ \mathrm{d}t$. This is only feasible for continuous signals $s(t)$.
- For discrete signals, for instance the trapezoidal rule can be used: $s_{nk} \approx 2^{-n/2}\sum_{\ell} s(2^{-n}\ell)\tilde{\phi}(\ell - k)\ \mathrm{d}t$. It is important to note that the point values of $\tilde{\phi}$ at the integer points are known. Higher order quadrature rules can be used too. The trapezoidal rule is one of the many quadrature rules, in general form written as [41]
$$\int_{t_1}^{t_2} f(t)\ \mathrm{d}t \approx \sum_{k=1}^{K} w_k f(v_k),$$
where K is the number of quadrature points, w_k the weights and v_k the quadrature points in $[t_1, t_2]$. The Gaussian rules are a special type of these quadrature rules, where the integration points and weights are chosen such that the highest order of accuracy is reached for a particular number of integration points v_k.
- Keinert [18] suggests to use $s_{nk} \approx s(2^{-n}k)$. Both Walnut [43] and Keinert [18] refer to the book of Strang and Nguyen [36], where this assumption is called a *wavelet crime*. However Keinert [18] shows that for smooth s (at least two times differentiable) the truncation error is smaller than the coefficients by a factor of order 2^{-n}.

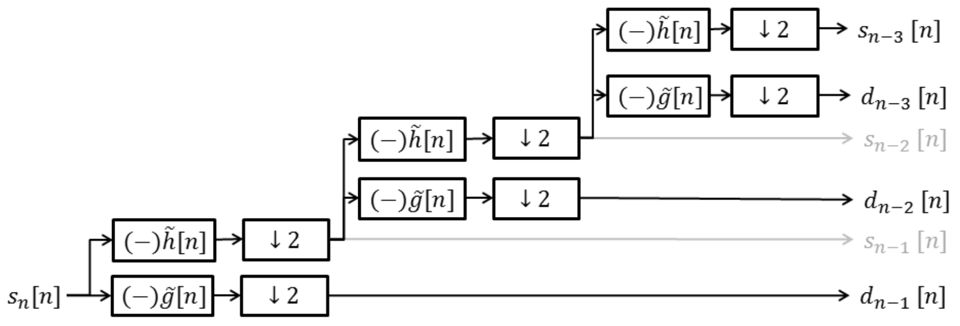
Postprocessing is the above described process in reverse: the conversion from s_{nk} to $s(t)$. There are two main approaches for postprocessing:

- Adding up scaling function expansions in between sampling points to retrieve the continuous signal $s(t)$. Because many scaling functions are not smooth, this might lead to a non continuous reconstruction [18].
- The alternative is finding intermediate points using interpolation.
- Other approaches from papers or books, a start point is given by Keinert [18].

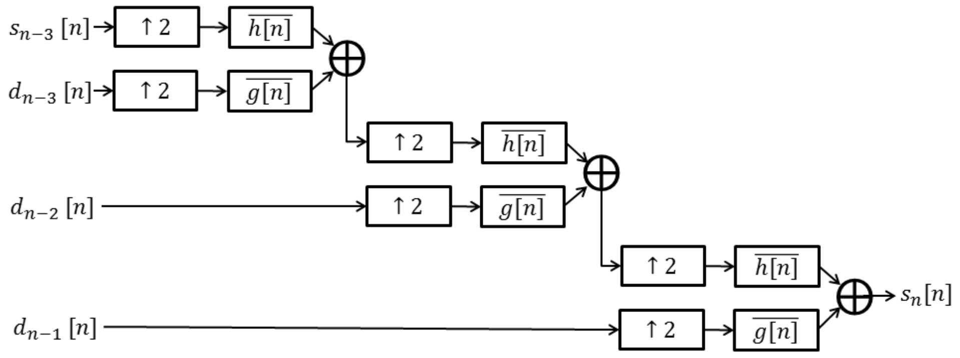


(a) One step decomposition **(b)** One step reconstruction

Figure 6.9: Filter formulation of the DWT algorithm.



(a) Three steps decomposition



(b) Three steps reconstruction

Figure 6.10: Filter formulation of the algorithm for a three step algorithm

6.4.4. Boundaries: different approaches

In Section 6.4.1 the infinite approach of the algorithm and the corresponding implementation (6.33) has been discussed. Of course this infinite approach is not implementable, therefore boundaries are introduced to abbreviate the infinite approach to a finite implementation. As for the infinite length DWT the finite length algorithm will be assumed linear, where the form of $\tilde{L}_n$ is changed to:

$$(\boldsymbol{sd})_{n-1} = \tilde{L}_n \boldsymbol{s}_n, \qquad \text{with } \tilde{L}_n = \begin{bmatrix} \tilde{L}_b & & \oslash \\ & \tilde{L}_i & \\ \oslash & & \tilde{L}_e \end{bmatrix}. \tag{6.34}$$

Here the subscript b, i and e stand for begin, interior and end. In the begin and end parts of the matrix, the boundaries will be handled. Generally the size of L_b and L_e are small with respect to the size of L_n and they both remain constant at all levels. The interior $\tilde{L}_i$ is a segment of the infinite matrix $\tilde{L}$ from (6.33). This part makes up the most of the matrix, approximately doubling in size when going from n to $n+1$. For the IDWT to exist, the matrix $\tilde{L}_n$ has to be invertible. In the orthogonal case $L_n^{-1} = L_n^*$, for the biorthogonal case the 'inverse' L_n^* has an analogous structure, being

$$L_n^* = \begin{bmatrix} \tilde{L}_b^* & & \oslash \\ & \tilde{L}_i^* & \\ \oslash & & \tilde{L}_e^* \end{bmatrix}.$$

There are three main ways to implement boundaries [18]. These boundary methods often do require some preprocessing. The choice for a specific method is data dependent. This will be discussed in the following enumeration. If one has non-periodic, non-symmetric data, and the boundary function approach is too hard, Keinert [18] advises to use linear extension for it is easy to implement and does not introduce artificial jumps in the data. For signals with enough zeros at the begin and end of the signal, the boundary handling is irrelevant.

- **The data extension approach**. This method extends the signal across the boundaries, such that each extended coefficient is a linear combination of known coefficients. The resulting $\tilde{L}_n$ can be singular, though most of the times it is not. The inverse L_n^* might not have the correct form such that reconstruction is possible. In this method several extension methods, known as modes in Matlab and Python:

- Periodic extension: the signal is assumed to be periodic, so the h- and g-coefficients disappearing on the left side of the matrix appear at the right and vice versa. This extension always works, preserving orthogonality and approximation $\mathcal{O}(h)$. Matlab and Python also know an extension method called periodization [26, 38]. Periodization is virtually the same as periodic extension, however it ensures the smallest length wavelet decomposition. When the data is not truly periodic, the jump at the boundary leads to large d-coefficients [18].
- Zero extension, also known as zero padding (see Chapter 7) is done by truncating the infinite matrix $\tilde{L}$. The non-existing infinite part of the signal is observed as 0's. This also introduces jumps, as the periodic extension, and this method does not preserve orthogonality or approximation orders [18].
- Symmetric extension mirrors the data in the endpoints, resulting in the coefficients in $\tilde{L}$ that disappear at the ends, get mirrored back. There are three ways of reflection: whole-sample symmetry, half-sample symmetry and antisymmetric reflection (only useful in the case $s_0 = 0$). If the data extension type matches the type of symmetry of the scaling function, the finite DWT will be equivalent to an infinite DWT [18].
- Extrapolation does not conserve the orthogonality condition. Constant extrapolation leads to approximation $\mathcal{O}(2^{-n})$, linear approximation (also known as smooth-padding [38]) to order $\mathcal{O}(2^{-2n})$ [18].

- **The matrix completion approach** guarantees $\tilde{L}_n L_n^* = I$, approaching the problem from a linear algebra view. The downside is that this approach generally does not conserve approximation order.
- **The boundary function approach** is the most time-consuming approach, preserving both orthogonality and approximation order. This approach introduces special functions at the boundaries of the interval which, unlike ϕ_0, do not extend over the border. The decomposition and reconstruction algorithm have to be worked out newly. The hardest part is to derive boundary wavelets which have the same number of vanishing moments as the original [21].
- Instead of assuming the periodicity of the signal, **periodic wavelets** can be used. Wavelets that cross one border of the domain are made periodic [21]. These wavelets create high amplitude wavelet coefficients in the neighborhood of the borders of the domain and do not have vanishing moments. The method is mathematically the same as extending the data periodically [21].

Example 6.7 (DWT decomposition) An example of a DWT decomposition: the signal at the bottom is decomposed four times. Note that the length of the detail coefficient vector is half the length of its predecessor and that the resulting smooth part and the fourth detail coefficients vector have the same length. The coefficients on the right have been translated to their contribution to the signal (left). The sum of the contributions results in the original signal. The error is negligible: in the order of 10^{-15} (see Figure 6.11c)

6.4.5. Formulations

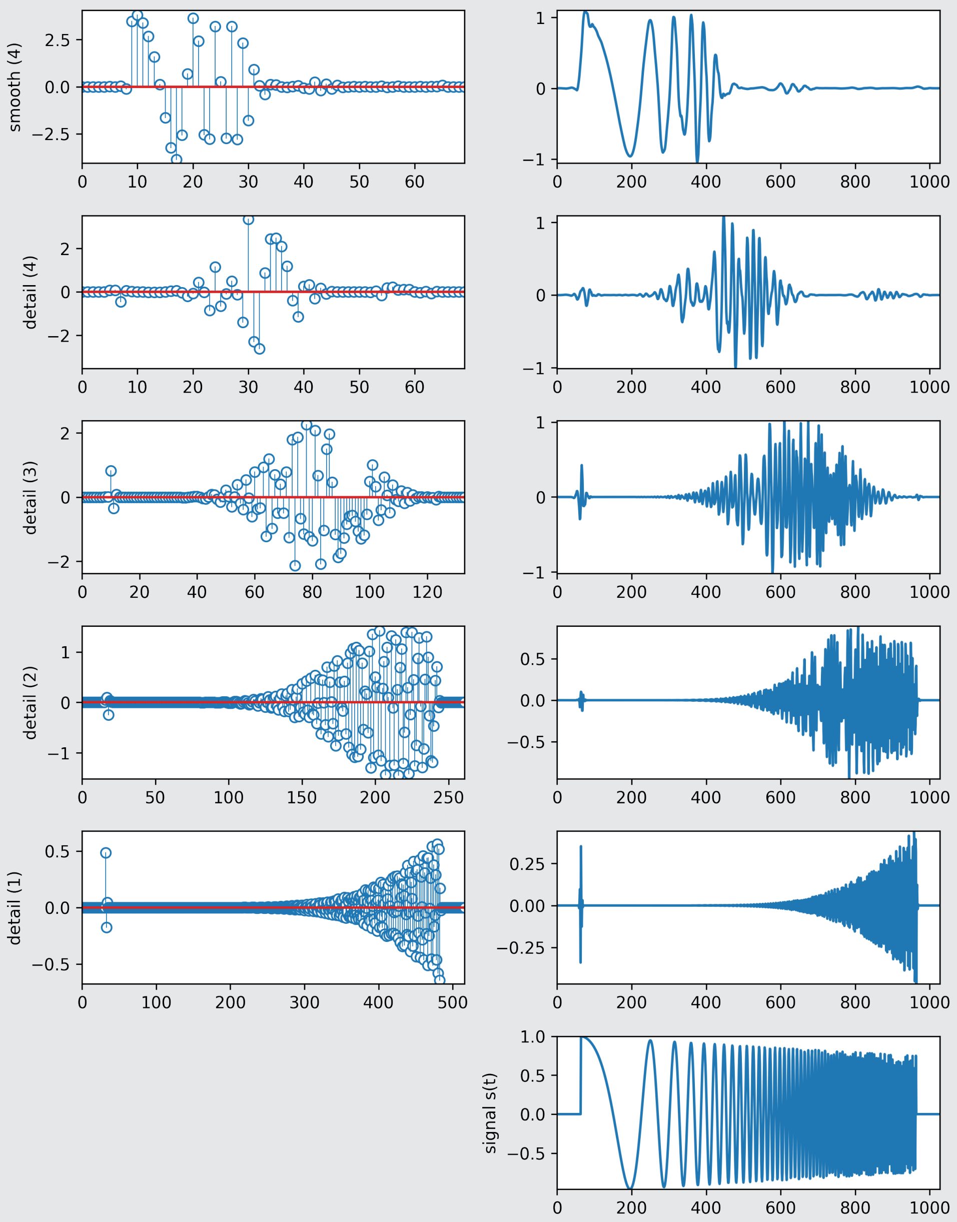
Till now we have seen the so called matrix notation of the DWT. This notation is very convenient for those used to linear algebra. There are two different formulations of the DWT. The first is the *modulation formulation*, which is a way of looking at the DWT from Fourier analysis. This way is not implementable, but this is useful in the creation of new wavelets bases, for instance by using the *lifting* method which will be discussed in Section 6.5.3. The *polyphase formulation* arranges the calculation of the DWT in such a way that convolutions are used without wasting computations. The direct implementation of the DWT in terms of the convolutions throws away the half of the computed values in the downscaling step. This is very regrettable, therefore the polyphase formulation uses convolutions without wasting computations [18].

Modulation formulation

Here we use the symbol formulation. For the discrete sequence $\boldsymbol{a} = \{a_k\}$ the symbol is defined as $a(\omega) = \sum_k a_k e^{-ik\omega}$. Remember that $c = a * b$, then $c(\omega) = a(\omega)b(\omega)$. Down- and upsampling are also defined for the Fourier domain as

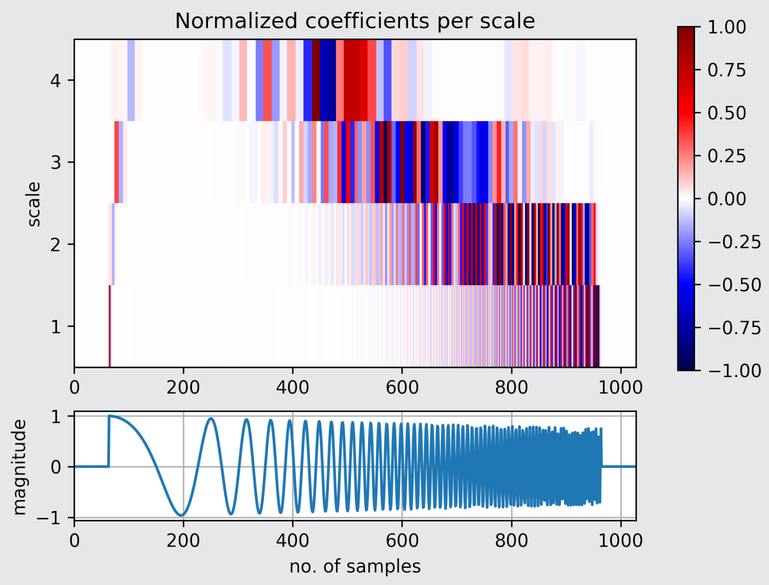
$$(\downarrow 2)a(\omega) = \frac{1}{2}\left[a(\omega/2) + a(\omega/2+\pi)\right],$$
$$(\uparrow 2)a(\omega) = a(2\omega).$$

The full DWT algorithm in terms of the symbols is the modulation formulation. The original signal is

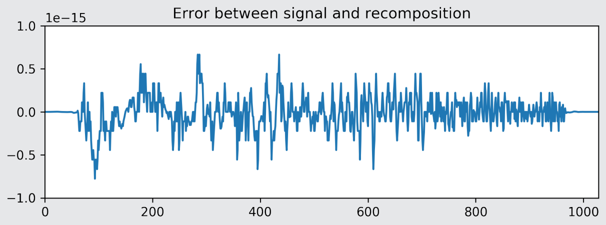


(a) An example of a DWT decomposition: the signal at the bottom is decomposed four times using the DWT algorithm. On the right the coefficients are translated to their contribution to the signal.

Figure 6.11: Figures with Example 6.7.



(b) The detail coefficients from (a) depicted differently.



(c) The error between the reconstruction and the original signal from Example 6.7, in numerical terms is the error 0.

Figure 6.11: Figures with Example 6.7.

$s_n(\omega)$, then the decomposition in s_{n-1} and d_{n-1} becomes

$$s_{n-1}(2\omega) = \frac{1}{\sqrt{2}}\left[\tilde{h}(\omega)s_n(\omega) + \tilde{h}(\omega+\pi)s_n(\omega+\pi)\right],$$
$$d_{n-1}(2\omega) = \frac{1}{\sqrt{2}}\left[\tilde{g}(\omega)s_n(\omega) + \tilde{g}(\omega+\pi)s_n(\omega+\pi)\right].$$

In this we recognize the convolution step from (6.32a) and the sampling step from (6.32b). The reconstruction is the same as (6.32c):

$$s_n(\omega) = \sqrt{2}[\overline{h(\omega)}s_{n-1}(2\omega) + \overline{g(\omega)}d_{n-1}(2\omega)].$$

This formulation can also be given in the matrix form. For the decomposition equation as

$$\begin{bmatrix} s_{n-1}(2\omega) \\ d_{n-1}(2\omega) \end{bmatrix} = \tilde{M}\cdot\frac{1}{2\sqrt{2}}\begin{bmatrix} s_n(\omega) \\ s_n(\omega+\pi) \end{bmatrix}, \qquad \text{with } M(\omega) = \begin{bmatrix} h(\omega) & h(\omega+\pi) \\ g(\omega) & g(\omega+\pi) \end{bmatrix}. \tag{6.35}$$

This matrix M is called the *modulation matrix*. For the reconstruction equation a redundant statement has to be added, which is the second row of the matrix formulation:

$$\begin{bmatrix} s_n(\omega) \\ s_n(\omega+\pi) \end{bmatrix} = M^*\cdot\sqrt{2}\begin{bmatrix} s_{n-1}(2\omega) \\ d_{n-1}(2\omega) \end{bmatrix}.$$

From these expressions another biorthogonality condition can be derived: $M(\omega)^*\tilde{M}(\omega) = I$. Note again that for the orthogonal MRA case, the tildes can be eliminated. The condition $M(\omega)^*M(\omega) = I$ is then called *paraunitary*.

Polyphase formulation

The polyphase formulation begins by splitting the signal and the recursion coefficients into odd and even phases. The notation used for the even and odd phases $\boldsymbol{a}_0$ and $\boldsymbol{a}_1$ of a sequence $\boldsymbol{a} = \{a_k\}$ is defined by $a_{0,k} = a_{2k}$ and $a_{1,k} = a_{2k+1}$. Using this notation, the convolution (6.32a) to find $\boldsymbol{s}_{n-1}$ can be written as [18]

$$\boldsymbol{s}_{n-1} = (-)\tilde{\boldsymbol{h}}_0 * \boldsymbol{s}_{n,0} + (-)\tilde{\boldsymbol{h}}_1 * \boldsymbol{s}_{n,1}.$$

The odd and even phases are computed separately and finally recombined. Note that the number of floating point operations is unchanged from the direct implementation.

The polyphase symbols of the sequence $\boldsymbol{a}$ are given by

$$a_0(\omega) = \sum_k a_{0,k} e^{-ik\omega} = \sum_k a_{2k} e^{-ik\omega}, \qquad a_1(\omega) = \sum_k a_{1,k} e^{-ik\omega} = \sum_k a_{2k+1} e^{-ik\omega}.$$

Then the decomposition of the signal $\boldsymbol{s}$ can be written as

$$\begin{bmatrix} s_{n-1}(\omega) \\ d_{n-1}(\omega) \end{bmatrix} = \tilde{P} \begin{bmatrix} s_{n,0}(\omega) \\ s_{n,1}(\omega) \end{bmatrix} \qquad \text{with } P(\omega) = \begin{bmatrix} h_0(\omega) & h_1(\omega) \\ g_0(\omega) & g_1(\omega) \end{bmatrix}.$$

Where $P(\omega)$ is called the *polyphase matrix*. As for the modulation approach, the tildes can be dropped when an orthogonal MRA is used. The reconstruction step then is

$$\begin{bmatrix} s_{n,0}(\omega) \\ s_{n,1}(\omega) \end{bmatrix} = P^* \begin{bmatrix} s_{n-1}(\omega) \\ d_{n-1}(\omega) \end{bmatrix}.$$

The biorthogonality condition in this formulation becomes $P(\omega)^* \tilde{P}(\omega) = I$. For the orthogonal MRA the polyphase matrix is paraunitary, as for the modulation formulation.

6.5. Wavelets

This sections starts with the derivation of orthogonal wavelets, with an example for the Daubechies 2 wavelet. This is followed by a short summary of theory on biorthogonal wavelets and te building and modification of wavelets. Then some common wavelets and their characteristics will be discussed. The section is concluded with a subsection about expansions of wavelet analysis: wavelet packets and multiwavelets.

6.5.1. Deriving orthogonal wavelets

A short overview of the important equations to design a wavelet will be given. Thereafter this will be applied to the derivation of the famous Daubechies wavelet [10]. At first we have the basic definitions of the scaling and wavelet functions from (6.9) and the same application to the wavelet function:

$$\phi(t) = \sqrt{2}\sum_k h_k \phi(2t-k), \tag{6.36a}$$

$$\psi(t) = \sqrt{2}\sum_k g_k \phi(2t-k). \tag{6.36b}$$

Their Fourier transforms, known as the symbols in (6.23) and (6.24). $H(\omega)$ and $G(\omega)$ are the discrete-time Fourier transforms of the discrete filters [20]. By substitution in the recursive equations, this leads to (6.26). From this we can switch from a low pass filter to a wavelet and vice versa. To recall:

$$\Phi(\omega) = H(\omega/2)\Phi(\omega/2), \qquad \text{with } H(\omega) = \frac{1}{\sqrt{2}}\sum_k h_k e^{-ik\omega}, \tag{6.36c}$$

$$\Psi(\omega) = G(\omega/2)\Psi(\omega/2), \qquad \text{with } G(\omega) = \frac{1}{\sqrt{2}}\sum_k g_k e^{-ik\omega}, \tag{6.36d}$$

$$\Phi(\omega) = \left[\prod_{k=1}^{\infty} H(2^{-k}\omega)\right]\Phi(0). \tag{6.36e}$$

The scaling equations are a connection between the scaling and the corresponding filters, on which the orthogonality condition (6.25) holds. Filters satisfying this relationship are called *conjugate mirror filters* [20].

$$|H(\omega)|^2 + |H(\omega+\pi)|^2 = 1. \tag{6.36f}$$

For the wavelet high pass filter a similar equation has to hold:

$$|G(\omega)|^2 + |G(\omega+\pi)|^2 = 1. \tag{6.36g}$$

In the next example, the assumption of a real filter will be used:

$$H(\omega) = \overline{H(-\omega)}, \qquad G(\omega) = \overline{G(-\omega)}.$$

These equations will lead to the design of a low pass filter. The design has to hold just for the frequency band $\omega \in [0, \pi/2]$, because the values in $[\pi/2, 3\pi/2]$ can be obtained using (6.36f), where the values in $[3\pi/2, 2\pi]$ are the result of the reflection due to the real filter property [20]. Finally the number of vanishing moments and the size of support of the wavelet has to be taken into account. It is preferable to compose an FIR filter to avoid stability and implementation issues. Liu [20] shows that if the support of $\phi(t)$ and $h[n]$ is the domain $[N_1, N_2]$, then the support of the wavelet $\psi(t)$ is $[(N_1 - N_2 + 1)/2, (N_2 - N_1 + 1)/2]$. For a higher number of vanishing moments, a longer filter size is indispensable. This forces a trade-off between the support and the number of vanishing moments [20].

The final part of the design process is the choice of number of vanishing moments p. Liu [20] gives five equivalent statements which can be used:

(i) $|\phi(t)| = \mathcal{O}((1+t^2)^{-p/2-1})$ and $|\psi(t)| = \mathcal{O}((1+t^2)^{-p/2-1})$;

(ii) The wavelet $\psi(t)$ has p vanishing moments;

(iii) $\Phi(\omega)$ and $\Phi(\omega)$ and its first $p-1$ derivatives are 0 at $\omega = 0$;

(iv) $H(e^{i\omega})$ and its first $p-1$ derivatives are zero at $\omega = \pi$;

(v) For any $0 \le k < p$, $q_k(t) = \sum_{n=-\infty}^{\infty} n^k \phi(t-n)$ is a polynomial of degree k.

Example 6.8 (Daubechies wavelet) In this example the Daubechies wavelets are derived. We start by combining (6.36f) and statement (iv) to write the low pass filter as

$$H(e^{i\omega}) = \sqrt{2}\left(\frac{1+e^{i\omega}}{2}\right)^p R(e^{i\omega}), \tag{6.37a}$$

where $R(x)$ is a polynomial. The wavelet will have vanishing moments p and it will be a minimum size discrete filter [20]. The absolute square of (6.37a) is

$$|H(e^{i\omega})| = H(e^{i\omega})\overline{H(e^{i\omega})} = 2\left(\cos\frac{\omega}{2}\right)^{2p} |R(e^{i\omega})|^2 = 2\left(\cos\frac{\omega}{2}\right)^{2p} P\left(\sin^2\frac{\omega}{2}\right). \tag{6.37b}$$

Now again apply (6.36f) and substitute $y = \sin^2(\omega/2)$, then we find (6.37c), which is uniquely solved using the Bezout Theorem [20], from which we find the solution (6.37d).

$$(1-y)^p P(y) + y^p P(1-y) = 1, \tag{6.37c}$$

$$\Rightarrow P(y) = \sum_{k=0}^{p-1} \binom{p-1+k}{k} y^k. \tag{6.37d}$$

From this minimum degree polynomial $P(y)$, the polynomial $R(e^{i\omega})$ can be derived. This is done by first decomposing $R(e^{i\omega})$ according to its roots (see Equation 6.37e) and then use the relation (6.37f) between P and R. Note the substitution of $e^{i\omega}$ by z to simplify the calculations.

$$R(z) = \sum_{k=0}^{m} r_k z^{-k} = r_0(1 - a_k z^{-1}), \tag{6.37e}$$

$$P\left(\frac{2 - z - z^{-1}}{4}\right) = r_0^2 \prod_{k=0}^{m} (1 - a_k z^{-1})(1 - a_k z). \tag{6.37f}$$

By solving the roots of the left-hand side, we find the roots of R, $\{a_k, 1/a_k\}_{k=\{0,\dots,m\}}$ and

$r_0 = 2^{p-1}$ Usually the a_k are chosen to lie within the unit circle to have a minimum phase filter [20].

This wavelet is named after its creator, Ingrid Daubechies, who published her paper in 1988 [9].

Daubechies 2 For the Daubechies 2 wavelet, the number of vanishing moments is 2: $p = 2$. This results in $P(y) = 1+2y$ (from (6.37d)) and the left hand size of (6.37f) becomes

$$P\left(\frac{2-z-z^{-1}}{4}\right) = 2 - \frac{1}{2}z - \frac{1}{2}z^{-1}. \tag{6.37g}$$

The roots of this equation are $2 \pm \sqrt{3}$, so the low pass filter of the Daubechies 2 filter is

$$H(e^{i\omega}) = \frac{\sqrt{2}+\sqrt{6}}{8} + \frac{3\sqrt{2}+\sqrt{6}}{8}e^{-i\omega} + \frac{3\sqrt{2}-\sqrt{6}}{8}e^{-i2\omega} + \frac{-\sqrt{2}-\sqrt{6}}{8}e^{-i3\omega}.$$

This results in the minimum filter with 2 vanishing moments and the corresponding size of 4. In general the minimum filter size is two times the number of vanishing moments [20]. The discrete time-domain representation of this filter is

$$h[n] = \frac{\sqrt{2}+\sqrt{6}}{8}\delta_{0n} + \frac{3\sqrt{2}+\sqrt{6}}{8}\delta_{1n} + \frac{3\sqrt{2}-\sqrt{6}}{8}\delta_{2n} + \frac{-\sqrt{2}-\sqrt{6}}{8}\delta_{3n}.$$

The Daubechies 2 scaling function, wavelet function, decomposition and reconstruction filters can be found in Figure 6.12.

The code to compile this figure is found in Listing B.7.

6.5.2. Deriving biorthogonal wavelets

The derivation of biorthogonal wavelets is more complicated than the orthogonal ones. Of this no example will be given. Keinert [18] derives the wavelets from their scaling functions. To find the wavelet functions belonging to a set of biorthogonal scaling functions, the definition of the modulation matrix from (6.35) is used. The biorthogonality conditions in terms of this matrix became $M(\omega)^*\tilde{M}(\omega) = M(\omega)\tilde{M}(\omega)^* = I$, from which we can derive

$$\tilde{M}(\omega)^* = M(\omega)^{-1} = \frac{1}{\det(M(\omega))}\begin{bmatrix} g(\omega+\pi) & -h(\omega+\pi) \\ -g(\omega) & h(\omega)\end{bmatrix}.$$

By comparing the entries of the two matrices, the following can be found [18]:

$$\tilde{g}_k = \frac{1}{\alpha}(-1)^k\overline{h_{2n+1-k}}, \qquad g_k = -\alpha(-1)^k\overline{\tilde{h}_{2n+1-k}}, \qquad \text{where } \det(M(\omega)) = \alpha e^{i(2n+1)\omega},\ \alpha \neq 0,\ n \in \mathbb{Z}.$$

Finding the dual scaling functions is a bit harder. It follows the same line of reasoning as in Section 6.5.1: the dual scaling function $\tilde{h}(\omega)$ is the solution to the equation

$$h(\omega)\overline{\tilde{h}(\omega)} + h(\omega+\pi)\overline{\tilde{h}(\omega+\pi)} = 1, \tag{6.38}$$

which can be converted into a Bezout equation (see Equation 6.37g). Constrains such as symmetry or vanishing moments can again be added as extra equations. For more information the reader is referred to Keinert [18].

6.5.3. Building and modifying wavelets

There are many ways to build wavelets from scratch. Some techniques are also used to modify wavelets into new ones. Keinert [18] gives an overview of the most common methods:

- **Biorthogonality relation**: by solving the relation (6.38) directly wavelets can be build. For short wavelets direct solving is possible, for longer wavelets it is very inconvenient [18]. The orthogonality relation (6.36f) can be solved better by using the Bezout equation, as seen in Example 6.8.

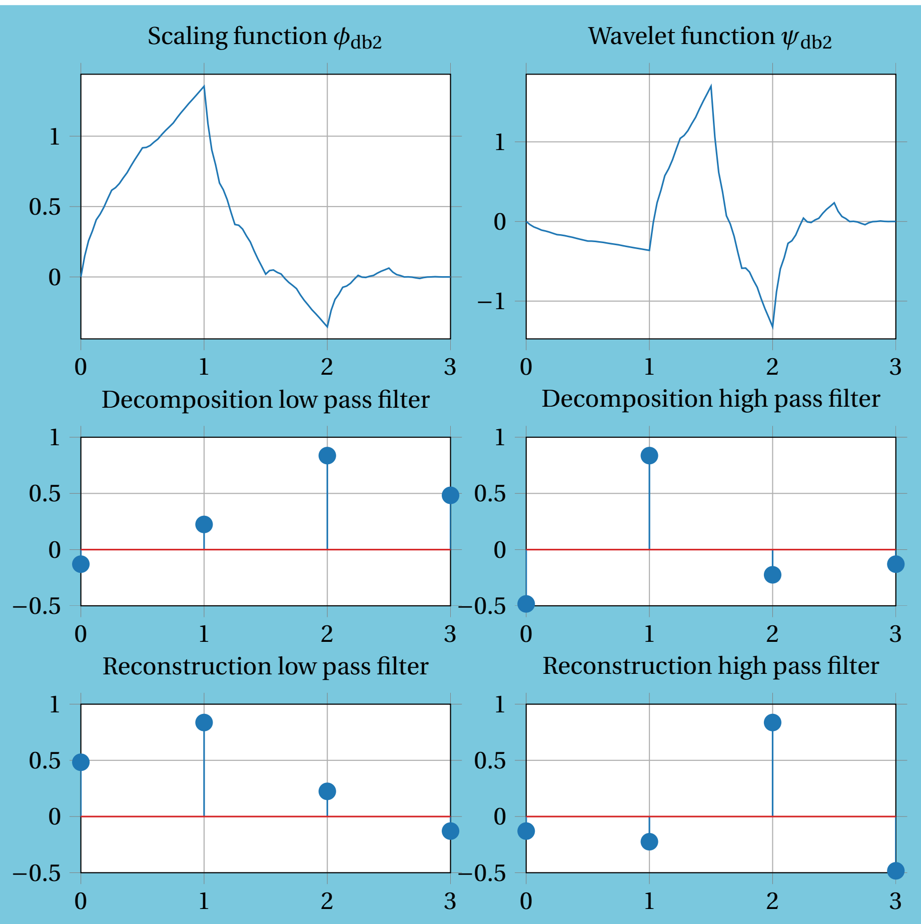


Figure 6.12: The Daubechies 2 scaling function, wavelet and discrete filters.

- **Lifting factors** are used to modify wavelets. These factors can be added to an existing wavelet (and its dual). Well chosen lifting factors preserve approximation order, but they always impair orthogonality [18]. To apply the lifting procedure, the reader is referred to Keinert [18].
- **Projecting factors** are both used in building and modifying. The projecting factors can be added to an existing wavelet, preserving orthogonality. The downside is the destruction of the approximation order. For longer wavelets this process has to be done numerically [18].
- **Shifting factors** are also used to both build and modify wavelets. The use of shifting factors results in biorthogonal wavelets of any size [18]. The modification process start with h and $\tilde{h}$ satisfying the relation (6.38). If h can be factorized as $h(\omega) = f(\omega)h_0(\omega)$ with $h_0(0) = 1$, then

$$h_{\text{new}}(\omega) = h_0(\omega), \qquad \tilde{h}_{\text{new}}(\omega) = \overline{f(\omega)}\tilde{h}(\omega).$$

  This results in a new biorthogonal wavelet. An example of this process is the Cohen wavelet, which is created by shifting factors in de Daubechies wavelet.

6.5.4. Common wavelets, an overview

Wavelet analysis has been applied already in a lot of fields. Therefore two commonly used numerical packages already have an extensive toolbox to do discrete wavelet transformations [26, 38]. These toolboxes, together with some books [4, 18, 27] are the basis of this short overview of most used wavelets and their characteristics. When the toolboxes are used, this is referred to by [TB]. The wavelets are split into discrete and continuous wavelets. The choice for a particular wavelet for a specific purpose is further addressed at the end of the next chapter.

Discrete wavelets

Discrete wavelets are derived from the MRA approach which are easily implemented on a computer for they possess easy implementable filters. These are all designed before 1992, most of them by Daubechies [11].

- **The Haar wavelet** has been discussed extensively in examples in this chapter. Named after its inventor Alfred Haar who laid the foundation of multi resolution analysis in 1910 [4].
- **Daubechies wavelets** are wavelets designed by Ingrid Daubechies [9]. They are called by their number of *vanishing moments* p, however sometimes they are referred to as $2p$, which is consistent with the filter length of the particular Daubechies wavelet. So the Daubechies 2 wavelet has two vanishing moments, and a filter length of four and therefore sometimes referred to as Daubechies 4. In this literature thesis the number of vanishing moments is leading. The derivation of this wavelet is treated in Example 6.8. Note that de Daubechies 1 wavelet is the same as the Haar wavelet. The Daubechies wavelets are *orthogonal*, have a *support* of $[0,2p-1]$ [18] and are asymmetric [TB].
- **Coiflets**, just like the Daubechies wavelets, are *orthogonal* wavelets. These are also designed by Ingrid Daubechies on request of Ronald Coifman [11]. They are designed such that not only the wavelet, but also the scaling function has a number of vanishing moments. The advantage of Coiflets is for smooth signals $s(t)$, the scaling function expansion coefficients s_{nk} are very close to $s(2^{-n}k)$ [18], such that preprocessing can be omitted. This results in *near symmetric* wavelets, which are all *orthogonal*. Narasimhan et al. [27] notes that some Coiflets have no zeros at $\omega = \pi$, which results in a *non smooth* wavelet, with possibly high frequency components. The error of the reconstructed signal therefore will be relatively large. There are again infinitely many Coiflets, numbered with n, with a number of vanishing moments $p = 2n$. The support of the wavelet is $[0,6n-1]$ and their filters are of length $6n$ [TB].

> *All above mentioned wavelets are* orthogonal *and* compactly supported. *These wavelet all have an orthogonal analysis, with an existing compactly supported $\psi(t)$ and $\phi(t)$. Their $\psi(t)$ has a known number of vanishing moments and their FIR filter description is very clear. These properties make them suitable for both continuous and discrete wavelet transformation, with applicability of the fast wavelet transform algorithm. The main difficulty using these wavelets is their poor regularity* [TB].

- **Symmlets** are an adaptation of the Debauchies wavelets. They are an answer of Daubechies to the request from engineers for *linear phase* filters. Such linear filters are symmetric around $b \in \mathbb{Z}$. Note that therefore the Haar wavelet is not symmetric, although its coefficients are symmetric. If a filter is not symmetric, its deviation from symmetry is judged by how much its phase deviates from a linear function. These symmetrized Daubechies wavelets are close to symmetric, but not completely symmetric [11]. This characteristic caused the wavelets of this family to be *biorthogonal*. Again these wavelets are numbered by their number of vanishing moments.
- **Cohen wavelets**, also known as Cohen-Daubechies-Feauveau wavelets are scalar *biorthogonal* wavelets derived from the Daubechies wavelet using the lifting factor technique. In both MATLAB and Python these wavelets are known as the **biorthogonal** wavelets [26, 38]. Again a lot of them can be made, the vanishing moments p and $\tilde{p}$ of the wavelet and dual wavelet function are restricted by $p+\tilde{p} = 2k$ for $k \in \mathbb{Z}$. They are symmetric around 0 if p is even and around 1/2 for p odd [18, 27]. The supports of these wavelets differs for the decomposition and reconstruction step (respectively ψ and $\tilde{\psi}$), they are $[0,2p+1]$ and $[0,2\tilde{p}+1]$. The filter lengths differ a lot by choice of p and $\tilde{p}$, but is limited by $\max\{p,\tilde{p}\}+2$ [TB]. Note that the Cohen(1,1) wavelet is the Haar wavelet.

Continuous wavelets

Continuous wavelets are defined by functions, not by their filters. However, most of them do have a filter implementation too. Often the filter has to be truncated to become finite. There is also a differentiation in *analytic* or *complex* and *real* wavelets [21, 40]. Analytic wavelets often are wavelets of which their Fourier transforms are zero for negative frequencies:

$$\Psi(\omega) = 0 \; \forall \omega < 0.$$

Therefore an analytic wavelet has to be complex, but it can be characterized by its real part only [21]. Their advantage is that they can measure time evolution of frequency transients by separating amplitude and phase components: it is better adapted to capturing oscillatory behavior in time series [40]. Real wavelets often are used to detect sharp signal transitions. Furthermore can complex wavelets be used to define a scalogram only for positive frequencies.

- The **Meyer** wavelet is a *symmetric* and *orthogonal* wavelet invented by Meyer in 1990 [11]. It is a *band-limited* wavelet, limited to $4\pi/2 \leq |\omega| \leq 8\pi/3$, which results in an *infinite support* in the time domain [27]. Its amplitude in time decays rapidly, making it suitable for wavelet analysis. The Meyer scaling function satisfies [27]

$$|\Phi_{\text{Meyer}}(\omega)|^2 = \begin{cases} 1, & 0 \leq |\omega| \leq 2\pi/3 \\ 1 - |\phi(2\pi - \omega)|^2 & 2\pi/3 \leq |\omega| \leq 4\pi/3 \\ 0, & \text{elsewhere} \end{cases}.$$

From this follows that the Fourier transform of the Meyer wavelet satisfies [4]

$$\Psi_{\text{Meyer}}(\omega) = e^{i\omega/2}(\Phi(\omega + 2\pi) + \Phi(\omega - 2\pi))\Phi(\omega/2).$$

The Meyer wavelet is a *symmetric, orthogonal* wavelet with non-compact support. The effective support of the Meyer wavelet is $[-8, 8]$ [TB]. This results in implementation of the Meyer wavelet transform in the Fourier domain, instead of the time domain. The Meyer wavelet has infinite vanishing moments [21].

> *The Meyer wavelet is known as an* infinitely regular wavelet [TB]. *It can be applied in both continuous and discrete wavelet analysis. Its filter implementation however is not a FIR filter. The analysis of the pair ψ_{Meyer} and ϕ_{Meyer} is orthogonal. Both functions are infinitely differentiable and have compact support. The symmetry and infinite regularity are its most important properties. The difficulty of this wavelet lies within the filter implementation: there is no fast algorithm available for the filter is of infinite impulse response.*

- The **Discrete Meyer** wavelet is the finite filter approximation of the Meyer wavelet [44]. It remains *orthogonal* and *symmetric*. By the truncation it is the FIR implementation of the Meyer wavelet. Technically this thus is a *discrete* wavelet.

- The **Mexican hat** wavelet is probably the best known continuous wavelet. It is defined by [27]:

$$\psi_{\text{mexh}}(t) = K(1 - t^2)e^{-(t^2/2)}, \qquad K = \frac{2}{\sqrt{3}\pi^{1/4}}. \tag{6.39}$$

The factor K normalizes the Mexican hat wavelet. If $K = 1$ the wavelet is known as the unnormalized Mexican hat [27]. The wavelet is named after its shape, however it is originally known as the **Ricker** wavelet. The Mexican hat wavelet is the second Gaussian wavelet, which characteristics will be discussed later. The Mexican hat specifically is well *localized* in time and has a *zero mean value* [27]. The support $[-5, 5]$ is used in the finite filter approximation.

- The **Complex Morlet** wavelet is defined by Narasimhan et al. [27] as

$$\psi_{\text{cm}}(t) = \pi^{-1/4}\left[e^{-i2\pi\omega_0 t}e^{-t^2/2}\right].$$

The general expression used in MATLAB is given on page 34. It is described as a complex wave $e^{i2\pi\omega_0 t}$ within the Gaussian envelope $e^{-t^2/2}$. The term in font of the expression ensures unit energy. For values of $\omega_0 \gg 0$ this wavelet has the property that it minimizes the error [27].

- The **Morlet** wavelet is the real part of the complex Morlet wavelet [27], which results in the function [TB]

$$\psi_{\text{Morlet}}(t) = e^{t^2/2}\cos(5t).$$

- The **Gaussian** wavelet family is a set of wavelets, derived from the derivatives of the Gaussian function

$$f(t) = e^{-t^2}.$$

This function can be derived infinitely many times, therefore there are a lot of Gaussian wavelets. They all do not have compact support, but do have an effective support of $[-5,5]$ [TB]. The best known is the Mexican Hat wavelet, which is the negative of the second derivative of the Gaussian function, multiplied by a scaling factor. This is an example of a nice wavelet, with compact support and mean value zero [27]. Not all derivatives lead to wavelet with these nice characteristics. For a n-times differentiated Gaussian, the number of vanishing moments is n too. Moreover, if n is even the wavelet is symmetric, for odd n it is anti symmetric [TB]. The **complex Gaussian** wavelet family has the same properties as the Gaussian wavelet family, the only difference is that this is a complex wavelet. The complex Gaussian wavelets are defined by derivatives of [TB]

$$f(t) = e^{it}e^{t^2}.$$

- The **Shannon** wavelet is defined by [27]

$$\psi_{\text{Shannon}}(t) = \frac{\sin(\pi t/2)}{\pi t/2}\cos(3\pi t/2).$$

From this definition we see it is *real* and *symmetric* but does not have a finite support [27], however is is infinitely differentiable and $\Phi(\omega)$ is zero in the neighborhood of $\omega = 0$, as are all its derivatives [21]. We may conclude that it therefore has infinitely many vanishing moments. It is therefore not a very good wavelet to apply wavelet analysis with [27]. The Shannon wavelet is a specific case of the spline wavelets [21].

The Morlet, Gaussian and Shannon wavelets are also known as crude *wavelets* [TB]. *These wavelets only have minimal properties. Their downsides are that $\phi(t)$ does not exist, the analysis is not orthogonal or biorthogonal and $\psi(t)$ is not of compact support. Therefore reconstruction is not insured and there are no fast algorithms to do calculations with. These wavelets are only useful for a (complex) continuous decomposition. Good properties are the symmetry and explicit declaration of the wavelet $\psi(t)$* [TB]. *Complex wavelet can have a spectrum for which $\Psi(\omega)|_{\omega<0} = 0$, such that a wavelet transform for only positive frequencies is possible.*

- **Spline** wavelets [TB], also known as **Battle-Lemarié** wavelets [21] are computed from spline MRA. The Fourier transform of the wavelet is chosen a block B-spline, leading to a band limiting FIR. The wavelet $\psi(t)$ has $p = m+1$ vanishing moments if its Fourier transform is made up of splines of order m. The resulting wavelet ψ has an exponential decay. They are again referred to with their number of vanishing moments p. In relation with the Meyer wavelet, they are less regular, but they decay faster [21]. For p even we have ψ symmetric about $1/2$, and for p odd it is antisymmetric around the same point. The spline wavelet of order 1 is again the Haar wavelet. The choice of such a spline MRA can lead to both *orthogonal* and *biorthogonal* wavelet bases [21, TB].

The Symmlets, Cohen and spline wavelets are part of the biorthogonal and compactly supported wavelet pairs [TB]. *They have in common that that $\psi(t)$, $\tilde{\psi}(t)$, $\phi(t)$ and $\tilde{\phi}(t)$ are compactly supported. They have a known number of vanishing moments and are regular. These wavelet are applicable in both continuous and discrete analysis, and again the FWT algorithm is applicable. The wavelets have symmetry with FIR filters, have desirable properties for decomposition and reconstruction and nice allocation is possible. The main downside is the loss of orthogonality* [TB].

6.5.5. Multiwavelets and wavelet packets

Wavelet analysis knows two extensions which are really common in modern research: multiwavelets and wavelet packets. These will be briefly be addressed. Remember the definition of the scaling function ϕ via the recursion relation (6.9), and its application in the refinement equation (6.15). Generalizations of this equation lead to all kind of other constructions [18, 27, 34]: wavelet packets, multivariate wavelets, ridgelets, curvelets, vaguelettes and much more. This theory is out of the scope of this literature thesis, but will be explained in a few sentences. The multiwavelet theory is more complicated than the wavelet theory. The general idea is the replacement of the scaling function $\phi(t)$ by a function vector $\boldsymbol{\phi}(t)$ (in **bold**), known as a *multiscale function*

$$\boldsymbol{\phi}(t) = \begin{bmatrix} \phi_1(t) \\ \vdots \\ \phi_r(t) \end{bmatrix}.$$

In the refinement equation (6.15) the recursion coefficients will become $r \times r$ matrices H_k:

$$\boldsymbol{\phi}(t) = \sqrt{m} \sum_k H_k \boldsymbol{\phi}(mt - k).$$

Also note the addition of the *dilation factor m*. In the MRA theory discussed so far, the factor $m = 2$ was used. The whole derivation can be done using a factor m, which adds some complexity to the notation and is therefore skipped. The factor $m = 2$ is also well suitable for computer implementation. The advantage of multiwavelets is their short support, coupled with high smoothness and high approximation orders. In addition they can be both symmetric and orthogonal, in contrary to the 'normal' wavelets. The disadvantages are the complexity of the theory and the requirement of pre- and postproccesing steps in the algorithm, which takes more computation time [18].

A wavelet packet choses different decomposition [34]. The spaces W_n are "split" again in two orthogonal subspaces. The basis functions for these subspaces, are constructed from both the refinable function as the wavelet. This approach leads to a redundant representation of the input data, and is best known by its use in the *FBI Fingerprint Compression Specification* [34]. The time frequency plane distribution changes from a 'regular' grid such as in Figure 3.1 but can be something like Figure 6.13, still the boxes have the same area by the Heisenbergs restriction. These techniques are not discussed further in this literature thesis.

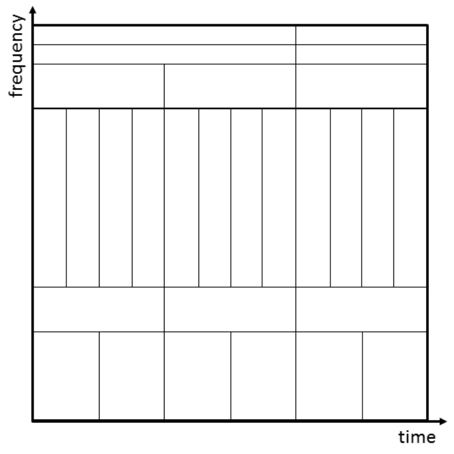


Figure 6.13: Example of the distribution of the time frequency plane by wavelet packets.

7
DSP - Digital Signal Processing

In the last three chapters we have seen three different signal analyzing techniques. These techniques have been made applicable to time signals in particular, but they can also be applied to for instance spacial information. The Fourier transform (FT, Chapter 4), short term Fourier transform (STFT, Chapter 5) and wavelet transform (WT, Chapter 6) do not only find their strength in just the analysis of a signal, but in the analysis signals can be processed too. A good example of a process is the time domain is the division of a signal into two parts of equal duration. In the frequency domain this process is close to impossible. On the other hand, when certain frequencies have to be canceled in a signal, one would definitely do this adjustment in the frequency domain, for this is very hard to do in the time domain. The possibilities using the time, Fourier and wavelet analysis will be discussed in this chapter about signal processing, especially *digital* signal processing (DSP): the origin and the result of the signals are considered digital. Sampling theory, described in Chapter 4, lies at the basis of digital signal processing. In this chapter mainly the power and differences between processing in the Fourier and wavelet domain will be discussed. The STFT has partially the same strengths and limitations as the Fourier transform and is therefore discussed less intensively. This chapter will conclude with the applicability to the problems discussed in Chapter 2.

7.1. Random signals

Till thus far we only have discussed deterministic signals in the examples. Remember that a deterministic signal can be modeled completely as a function of a variable, for instance time. A deterministic signal has a known and unambiguous value at every point in time [7, 15]. These signals are composed of additions, subtractions, delays, derivations and integrals of deterministic parts. This is not applicable to signals in real life applications. There, most signals are *non deterministic*, also known as *stochastic* or *random signals* [15] These non deterministic signals are modeled using probability theory. Here a short overview of the probabilistic theory together with some important definitions for signal processing purposes will be presented.

A *random process* is a process that generates random signals in general. Such a process can produce an entirety of random signals, this entirety is called an *ensemble*. One signal from such an ensemble is a *sample function* (in mathematics) or *realization* (in signal analysis) of the random process. A random signal cannot be described deterministically. To characterize them using deterministic descriptions, characteristics which are valid for the whole ensemble are used [15]. The whole ensemble is described as a set functions $\{x_i(t)\}$, consisting of the individual random signals $x_1(t)$, $x_2(t)$ etc. The *expected value* or *ensemble mean* of a random process at a certain time t_0 defines the mean value of the whole ensemble as [15]:

$$\mathrm{E}[x(t_0)] = \lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} x_i(t_0),. \tag{7.1}$$

For most signals the expected value is time dependent. Note that the *time average* of a signal is the average over the whole time domain, which can be computed for every ensemble member individually. The quadratic average of an ensemble, $\mathrm{E}[x^2(t)]$, is used to describe the average power of a random process. The square of the deviation from the linear average is called the variance:

$$\mathrm{Var}(x(t_0)) = \sigma^2(t) = \mathrm{E}\left[(x(t_0) - \mathrm{E}[(x(t_0))])^2\right] = \mathrm{E}\left[(x(t_0))^2\right] - \mathrm{E}[x(t_0)]^2. \tag{7.2}$$

The variance of a signal is a measure of the amount of fluctuation from its mean. The square root of the variance is known as the *standard deviation* $\sigma(t) = \sqrt{\sigma^2(t)}$. The average and variance are often sufficient to describe common random signals. The more general formulation of the expected value and the variance at a certain time t_0 is [7]

$$\mathrm{E}[f(x(t_0))], \tag{7.3}$$

where for the expected value: $f(x) = x$ and for the variance we choose $f(x) = (x - \mathrm{E}[x])^2$. A probability density function is a description of a characteristic (7.3) for the whole time t. Advanced models of stochastic process can become very complicated, using *higher-order statistics* [15], such as $f(x) = x^3$ in (7.3).

Stationary and ergodic random processes

So far, the general aspects of a random process have been discussed. Random processes have a lot of different classifications. Two important classes of random processes will be explained. A random process is called *stationary* if its statistical properties do not change with time. So in general a signal is *stationary to the order N* if [7]:

$$\mathrm{E}[f(x(t_1), x(t_2), \ldots, x(t_N))] = \mathrm{E}[f(x(t_1 + \Delta t), x(t_2 + \Delta t), \ldots, x(t_N + \Delta t))] \tag{7.4}$$

for all Δt. This expression is hard to use, therefore often a random process is *stationary* if its second-order expected values only depend on the difference of observed time points $t_1 - t_2$ [15], mostly checked using the *autocorrelation function* $\mathrm{E}[x(t_1)x(t_2)]$. This mathematical description excludes finite random signals and deterministic functions other than constant functions as stationary [15]. A random process is called *weak stationary* if only the autocorrelation and the expected value of a signal are stationary [5, 7], respectively if $\forall \Delta t$

$$\mathrm{E}[x(t_1)x(t_2)] = \mathrm{E}[x(t_1 + \Delta t)x(t_2 + \Delta t)], \qquad \mathrm{E}[x(t_1)] = \mathrm{E}[x(t_1 + \Delta t)].$$

Stationary signals are also described as signals whose Fourier analysis coefficients are time invariant [27]. This implies that both the power spectral density and the autocorrelation of the signal are time invariant too. For a stationary random process for which all time averages are the same as all ensemble averages, the process is an *ergodic* random process [7, 15]. This holds for all function $f(\boldsymbol{x})$ from (7.4). A signal is again called *weak ergodic* if the autocorrelation and expected value of the signal are the same as their time averages [15]:

$$\lim_{T\to\infty} \frac{1}{2T} \int_{-T}^{T} x_i(t) x_i(t - \Delta t)\, \mathrm{d}t, \qquad \lim_{T\to\infty} \frac{1}{2T} \int_{-T}^{T} x_i(t)\, \mathrm{d}t.$$

Noise

A received waveform $x(t) = s(t) + v(t)$ usually consists of two parts: a desired part $s(t)$, containing the information, and the undesired part $v(t)$ [7]. The desired part is referred to as the *signal*, the undesired part as *noise*. The sum of these parts is then referred to as the *noisy signal* or *received signal*. This is noise in the most broad sense of the word and it has all kinds of shapes and sizes: it can be 'added' to the desired signal by the sender, the measurement equipment or other processes. Often noise is assumed to be of constant power in all frequencies, *white noise*, but could also be contained in specific frequencies, which is known as *colored noise*. These names are in convention with light, where white light contains photons of all frequencies and colored light only those of specific frequencies. Noise does not always have to be of constant power, it can for instance fade out or build up. This is known as *non stationary noise*. The amount of noise on a signal is often described using the signal-to-noise ratio, SNR for short (sometimes denoted as S/N) [7]. It is the ratio between the power of the signal and the power of the noise:

$$\mathrm{SNR} = \frac{P_{\text{signal}}}{P_{\text{noise}}}, \tag{7.5}$$

often expressed in decibels. If the variance of the signal and noise are known and the signal is of zero-mean, the SNR is also expressed by

$$\mathrm{SNR} = \frac{\sigma^2_{\text{signal}}}{\sigma^2_{\text{noise}}}. \tag{7.6}$$

Amplification or delay

The noise theory discussed till now concerns noise as an addition to the signal: additional noise. Often a large share of the mutation of a signal consists of this additional noise [7]. However noise can also adjust or adapt the signal in other ways: noise can also be a distort the amplification and delay of (specific frequencies) the signal. Amplification (or damping) and the delay of parts of the signal may have different causes. If the whole signal has been amplified or delayed with the same factor, this does not affect the signal analysis that much. These two noise effects are often modeled as filters. Damping is often a result of the range of the measurement equipment. For instance the detection of sounds by the human ear rapidly decays under 50 Hz, which can be modeled as a damping. In a spectrum or scalogram this can be noted from the amplification or damping of specific frequencies or detail coefficients.

Delays can have a lot of different causes, also often modeled as a filter. As addressed before, delay is modeled as a phase change in the Fourier domain (A.2). Therefore it is also known as *phase noise*. Phase noise is generally hard to detect, because it is a non linear effect. E.g. $\sin(\omega_0 t + \psi(t))$ has a basis frequency of $2\omega_0\pi$, however this might be altered by the phase change $\psi(t)$. Because both the wavelet and Fourier transform are linear transforms, some phase changes $\psi(t)$ might be hard to detect: by the linearity they are modeled as different frequencies instead of one phase changing frequency. A good example is shown in the spectrum of the quadratic chirp signal in Figure 4.4b. The noise expression can thus be expanded like $x(t) = \epsilon(t)s(t + \psi(t)) + v(t)$.

7.2. Detection and filtering

In this section some basic concepts within digital signal processing will be discussed. These concept mainly concern detection and filtering. Detection is the term that describes discovery and classification of expected or unexpected behavior in a signal, whereas filtering concerns the deletion of unwanted or superfluous part of a signal. We start with a short overview of the most important processes, of which some will be elaborated.

- **Denoising** is the study of the removal of noise on a signal. Two different main types of additional noise will be discussed: white noise and colored noise. The filtering of noise in time, Fourier and wavelet domain will be addressed.

- **Transients** is a container concept for little amplitude disturbances of short duration [27] . These are best detected using wavelet analysis. Once detected, they can be reconstructed and possibly subtracted from the signal, resulting in a filtered signal. Note that the term transient sometimes also is used for momentary signal elements, which do not have less energy then the signal.

- **Discontinuities** can be very clear from a time signal, but smaller discontinuities can be harder to locate. Again wavelet analysis is a good way of detecting discontinuities in time. Often discontinuities are not taken out of the signal. Other filtering processes may have effects on discontinuities in a signal.

- Specific **frequencies** such as resonance frequencies can disturb data and therefore interesting to filter. This is briefly discussed under colored noise.

- Both Fourier and wavelet transforms are linear transformations. **Non linear** signal elements will therefore always be described as linear elements in the analysis. In the examples identifying the quadratic chirp (Example 4.7, 5.2 and 6.1) still a quadratic behaviour is recognized, but for different non linear signal or non linear signal elements one can imagine this does not always work.

- **Non stationary** signal elements are elements that have changing statistical properties over time. One can think of bandwidth changing noise, which may be caused by the warming of measurement equipment. These effects are considered hard to determine and filter.

- **Compression** is not a standard signal processing subject, but it is part of the domain where wavelet analysis has become most popular: image processing. A great and still used example is the JPG image format [18, 27]. In compression it is the challenge to discard the unimportant data (a.o. noise) and to retain the essential information. This will not be elaborated further.

7.2.1. Denoising

The process to remove noise from a signal is called denoising. Often noise is assumed to be an addition to the signal as $x(t) = s(n) + v(n)$, by the linearity of the FT, the spectrum then becomes $X(\omega) = S(\omega) + V(\omega)$. These noise processes are mostly concerned stationary. To denoise in the frequency domain often the spectral values of which the magnitude are above a certain threshold, $|X(\omega)| \geq \omega_0$, are set to zero [5, 27]. The inverse of the resulting signal $\tilde{X}(\omega)$ is supposed to have reduced noise. The threshold value can be determined with the help of a histogram of the spectral magnitude values [27]. This thresholding removes high frequency values, which are indispensable for the reconstruction of sharp edges or discontinuities in a signal. Furthermore a sharp cut-off will lead to Gibbs ripples in the time domain. This sharp cut-off is the same as multiplication with a rect-window. Other window types are chosen to have a less ripply result [7].

Other well known noise reduction methods in the Fourier domain (especially in speech processing) are spectral subtraction and the Wiener filter [33]. In the spectral subtraction method a 'signal free' period is chosen. The spectrum of this period is then subtracted from spectra of periods containing the signal. The Wiener Filter computes a linear estimation of the signal which is optimal in the minimum mean square error sense. Therefore it needs spectral information of the noiseless signal and the noise to derive a filter that mimics the behavior of the noiseless signal. The Wiener Filter shows better results than the spectral subtraction method [47]. Both methods are relatively weak in non stationary signal environments, moreover they suffer from so-called 'musical notes': the subtraction can lead to artificial zeros in the spectrum, leading to the noticeable absence of those frequencies [33].

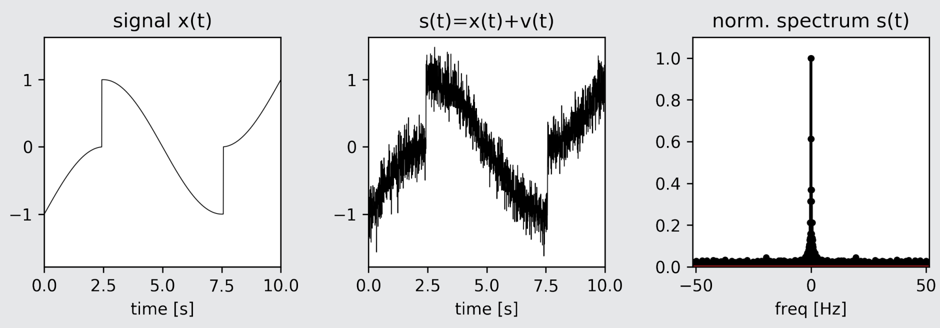


Figure 7.1: Signal $x(t)$, noisy signal $s(t) = x(t) + v(t)$ (SNR=10 dB) and its spectrum.

Example 7.1 (Filtering noise using Fourier) In this example the signal in Figure 7.1 will be filtered in the frequency domain. Therefore the spectrum is multiplied with a rectangular window and a Hanning window (see Figure 7.2). For the use of the rectangular window, we expected a lot of ripples, which are clearly present in the filtered signal. The Hanning window should let through some more noise, but is should also result in less ripples, especially near discontinuities. The resemblance of the original signal $x(t)$ is better using the Hanning window then the rectangular window.

The Python-code for this example: Listing B.8.

In the wavelet domain random noise will mostly show up in the d-coefficients [18]. Just as in the Fourier case, if the smaller coefficients are set to zero, much of the noise will disappear, inevitably together with some features of the signal. The Gibbs phenomenon on the other hand will not play a role in the wavelet denoising. The thresholding technique is most applied to white noise, distributed over all scales, which is assumed of smaller magnitude than the signal [27]. The most commonly used thresholding functions, $d_{nk} \to T_\epsilon(d_{nk})$, are *hard* and *soft* thresholding [1, 18, 27]. Hard thresholding sets values smaller then the threshold to zero:

$$T_\epsilon(d) = \begin{cases} 0 & \text{if } |d| < \epsilon \\ d & \text{otherwise} \end{cases}, \tag{7.7}$$

whereas soft thresholding also shortens the values above the threshold:

$$T_\epsilon(d) = \begin{cases} d - \epsilon & \text{if } d > \epsilon \\ 0 & \text{if } |d| \leq \epsilon \\ -d + \epsilon & \text{if } d < -\epsilon \end{cases}. \tag{7.8}$$

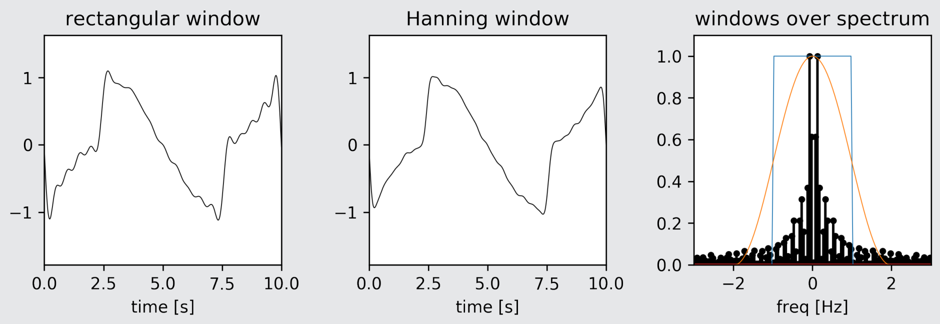


Figure 7.2: The signal $s(t)$ has been denoised using a rectangular window (blue, resulting in the left reconstruction of $x(t)$) and a Hanning window (orange, resulting in the right construction of $x(t)$) in the frequency domain.

The soft thresholding can be read as removing the noise component part from the coefficient d, therefore it is known as *wavelet shrinkage* [27]. The implementation of a such an algorithm is shown in Figure 7.4a Edge information will be in the lower scales (high resolutions) and have large magnitudes compared to the noise. So in comparison to the filtering in the Fourier domain, a better preservation of the edge information is predicted. The wavelet thresholding will therefore preserve more edge information than the described Fourier denoising [27]. No scale will be removed completely, unless it has a very small contribution to the signal. For every scale i a new choice for a threshold is made.

Often the variance or standard deviation of the coefficients in the i^{th} scale multiplied with a scaling factor is used. An other option used is a *universal threshold* given by [1, 27]

$$T_\epsilon = \sqrt{2\ln N_\nu}\sigma_\nu. \tag{7.9}$$

The factor in the square root represents the expected maximum value of a unit variance Gaussian white noise sequence of length N_ν. σ_ν is the standard deviation of the noise, which in practice is not known. The estimator for σ_ν, denoted as $\hat{\sigma}_\nu$, is based on the median of the absolute deviation (MAD) of the wavelet coefficients at the lowest scale, multiplied with a conversion factor to match with the Gaussian distribution: $\hat{\sigma}_\nu = \text{MAD}/0.6745$ [27]. For large samples the removal of noise is effective, however it will remove parts of the signal too. Because in soft thresholding more coefficients are shortened, the reconstruction is more adjusted. Therefore when the universal threshold is used in soft thresholding, the value half the value for the hard thresholding is used [27]. Other, more complicated thresholding techniques are described by [1]. This are non linear regularization methods, which are often superior to the soft and hard thresholding techniques.

Example 7.2 (Filtering noise using wavelets) Now we apply filtering using wavelets on the signal from Figure 7.1. Of the described methods, three have been applied: soft thresholding with the universal threshold, hard thresholding with the universal threshold and thresholding per scale with a threshold of 1.3 times the standard deviation of the scale. The results are shown in Figure 7.3. If we compare these three results, the choice of thresholding per scale clearly has more noise on it than the other two. Between the hard and the soft thresholding, we note that the peaks we see in the hard threshold are less high than the low threshold peaks. In comparison with the filtered Fourier transforms both universal thresholded wavelet transforms give a better approximation of the discontinuities.

The difference of the filtered $s(t)$, $s_f(t)$ and the the signal $x(t)$ has been measured by $\sum_{k=1}^{2056}|s_f(t)-x(t)|$ to show the wavelet filtering approaches the signal $x(t)$ better. The difference between the noisy signal $s(t)$ and its wavelet recomposition is $5.1\cdot10^{-13}$ and therefore negligible.

Error between $x(t)$ and ...	**Error**
$s(t)$	320
rectangular window filter	136
Hanning window filter	94
Hard universal thresholding	84
Soft universal thresholding	78
Scaled thresholding	161

In this example, the wavelet filtering performs better, for it has a smaller error and it is better in approximating the discontinuities than the Fourier transform. For larger SNR, this is not necessarily still valid. A combination of Fourier and wavelet noise canceling might be more suitable.

The Python-code for this example: Listing B.8.

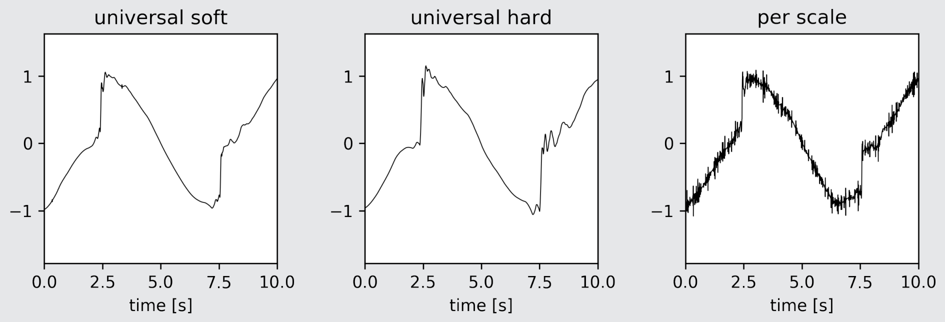


Figure 7.3: The signal $s(t)$ from Figure 7.1 has been denoised using three different techniques: soft and hard thresholding, both using the universal threshold (7.9); and a thresholding per scale of 1.3 times the standard deviation. For this example the Daubechies 4 wavelet has been used.

Shifting and cycle spinning

Denoising by thresholding works better if several shifted copies of the signal are denoised and averaged [1, 18, 27]. The wavelet denoising algorithm is not shift invariant because the DWT is not shift invariant. If the discrete signal is shifted by one, different coefficients at the next lower level will appear. If the signal is shifted by 2, the original coefficients rise again. After k levels, we need a shift by 2^k before the original coefficients at the lowest level reappear [18]. Denoising several shifted copies and averaging them, improves the result. Implementation of the shifted denoising algorithm is more efficient when decomposition without down sampling is done. In the regular DWT algorithm the signals are down sampled to match the filter. In the so called *shift invariant* discrete WT the filters are interpolated in order to match the signal [27]. The amount of work for a signal of length N over k levels is $\mathcal{O}(kN)$ instead of $\mathcal{O}(2^k N)$ for separately denoising all shifts 2^k shifts [18]. Note that more memory capacity will be needed.

Finally an adaptation of the shifted denoising algorithm is the *cycle spinning algorithm* [27]. This algorithm is a recursive algorithm, where the previous iterations output becomes the input to the new iteration [27]. The initialization is done by setting the first input equal to the original noisy signal $x[n]$: $\hat{y}_0[n] = x[n]$. The recursive update follows from

$$\hat{x}_{\ell+1}[n] = D_i(x_\ell[n]), \qquad i,\ell = 0,\dots,M-1,$$

where $D_i(x)$ is the denoising operator that uses shift i. This algorithm converges for sufficiently large M, however the threshold should be chosen carefully to not smoothen the discontinuities [27].

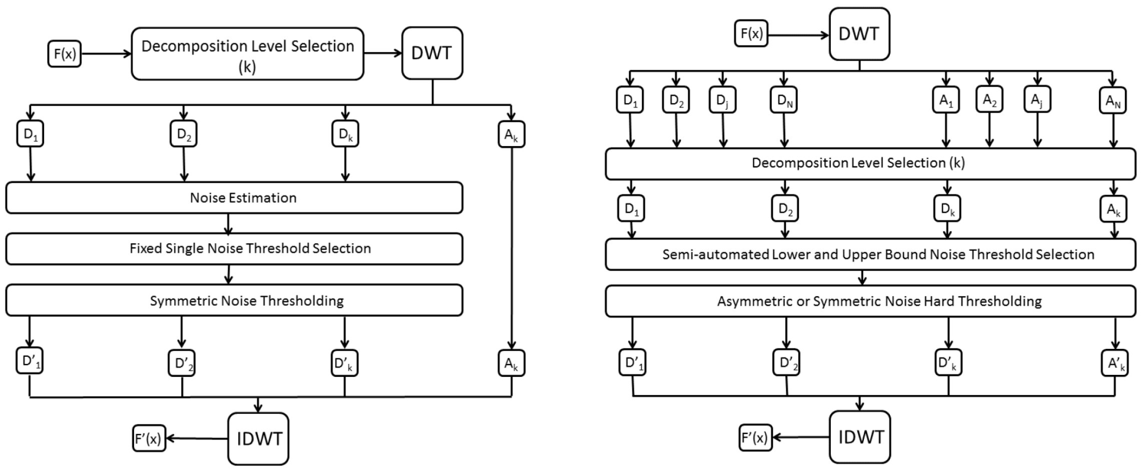
Colored noise

Till now we have assumed the noise is white noise. This is only valid for a small number of applications. Often noise is colored: it covers a specific bandwidth. If we for instance assume a speech signal, car noise would be colored noise on the speech signal. In Fourier analysis an 'inverse' bandpass filter can be applied over the noisy bandwidth to filter that specific bandwidth. This is known as a *notch* or *band-stop* filter. This however does not only address the specific frequencies, but also some of the surrounding frequencies. This is a known problem in audio processing, where they use a so-called *anti-hum filter*, for the 50 Hz distortion of the signal can be heard as a low frequency hum in the signal. Filter application in Matlab is proven effective for audio signals [24]: at least half the power of the frequency components in the short range is reduced.

For colored noise reduction in wavelet analysis, a scale dependent threshold can be used in stead of a universal threshold [6, 27]:

$$T_j = \sigma_j \sqrt{2 \ln N_j}, \qquad \sigma_j = (\mathrm{MAD})_j / 0.6745,$$

where now the number of samples used and the median of the absolute deviation is dependent on the scale j. This choice of threshold is still very rigid and will not always show the desired result. However, the choice of threshold still can be very hard. A block diagram of the standard wavelet noise suppression method is shown in Figure 7.4a. More state of the art level-dependent noise threshold selection methods such as the Stein's Unbiased Risk Estimate (SURE) and the minmax threshold show excellent performance [35]. The downside of these methods is that they only work for white Gaussian noise [6]. Srivastava et al. [35] excellently describe the drawbacks of wavelet shrinkage methods for data analysts and suggest a new denoising method. This method describes decomposition level choice and automated thresholding (however, user input is possible) applicable to measurement data analysis. The resulting algorithm is tested on spectroscopy time series and they are promising. A block diagram of this algorithm is shown in Figure 7.4b. An interesting feature is of this proposed algorithm is the choice for hard thresholding. An other threshold definition proposed by Srivastava et al. [35] is to select a 'signal free' period and base the thresholding on those. Often in non stationary noise filtering wavelet packets are used, with better result then standard wavelet decomposition [27]. There are also other noises such as brown noise, pink noise and red noise. These are noises that show a specific behavior, for instance red noise has a larger contribution in lower frequencies [40]. How to address these is not discussed here.


F(x)
Decomposition Level Selection (k)
DWT
D1
D2
Dk
Ak
Noise Estimation
Fixed Single Noise Threshold Selection
Symmetric Noise Thresholding
D'1
D'2
D'k
Ak
F'(x)
IDWT
F(x)
DWT
D1
D2
Dj
DN
A1
A2
Aj
AN
Decomposition Level Selection (k)
D1
D2
Dk
Ak
Semi-automated Lower and Upper Bound Noise Threshold Selection
Asymmetric or Symmetric Noise Hard Thresholding
D'1
D'2
D'k
A'k
F'(x)
IDWT

(a) Block diagram of standard wavelet denoising method (or shrinkage method).

(b) Block diagram of the wavelet denoising method, proposed by Srivastava et al. [35].

Figure 7.4: Two block diagrams for wavelet denoising algorithms from [35, fig. 1&2]. The discrete signal to denoise is $F(x)$, denoised signals and coefficients are denoted with an apostrophe. A_k stands for the k^{th} smooth level (or approximation level [35]).

Time domain

Next to the frequency and wavelet domain, noise can also be reduced 'in the time domain', or better stated: without doing a transformation. The advantage of this type of filtering is that in theory less calculations are needed, because the signal does not have to be transformed (twice). These filters (except the last one) are often referred to as signal smoothing filters, for they optically make the signal smoother. The most used filters are [23]:

- The **moving average filter** of length N takes the average of every N consecutive samples of the waveform [23]. The two sided moving average filter for N even is defined by [5]:

$$x_{\text{filtered}}[i] = (N+1)^{-1} \sum_{j=-N/2}^{N/2} x[i-j].$$

  Especially linear trends are reconstructed well. For $x[i]$ not linear, it is important not to choose N to large, for the filtered signal will be smooth, but not a good estimator of $x[i]$. N can also be chosen odd, and the center of the filter does not necessarily has to be the sample $x[i]$. This results however in a delay, which is easily corrected.

- The **weighted moving average filter** does not weight each sample equally, common weighings are the binomial expansion of $[1/2, 1/2]^n$, known as the Gaussian expansion filter [23], and the exponentially weighted moving average [5]:

$$x_{\text{filtered}}[i] = \alpha x[i] + (1-\alpha)x_{\text{filtered}}[i-1], \quad i = 1,\dots,N-1 \quad \text{with } x_{\text{filtered}}[0] = x[0].$$

- The **Savitzky-Golay filter** is a weighted moving average filter that fits a polynomial of a specified order over a specified number of samples in a least-squares sense. The fitting of a polynomial is better in preserving peak values [23].

- A signal with sharp edges will be under-corrected by a moving average filter and over-corrected by a Savitzky-Golay filter [23]. A better approximation is given by the **median filter**, which uses, instead of the average, the median of the values in the window and is therefore recommended for non linear data [2].

- The **Kalman filter** finds it origin in control theory, but also found its way to time series analysis [5]. The Kalman filter consists of two parts: a model prediction and a measurement. The model prediction gives an estimate of the unknown quantity at a certain moment, based on a computer model. This estimate has a certain uncertainty. The measurement is a recording of that same quantity with a certain amount of noise: it will have a different uncertainty. The Kalman filter combines the model prediction and the measurement to create a better estimate: the uncertainty of the prediction by the filter is limited by the minimum uncertainty used, but often it is even less uncertain. The computational load of the Kalman filter depends on the complexity of the used model. In general for a signal of length N, the filter algorithm is equivalent with $\mathcal{O}(2N+1)$ model simulations [46]. For a relative simple model this would result in $\mathcal{O}(N^2)$ operations.

  The most important assumption in the derivation of this filter is assuming that the probability density function of both the model prediction and the measurement are Gaussian [13]. This assumption ensures convergence in Gaussian noise situations. Moreover, the **extended Kalman filter** can be used for non linear problems, it performs a linearizion of the model around the mean of the most recent estimate. Another solution for non linear problems is the **ensemble Kalman filter**, a Monte Carlo type approach, which enables the treatment of non linear problems without the need to introduce an extra model [46]. Computational cost of this filter can be high, mainly driven by the computational costs of the model.

For samples disturbed by (relative high frequent) periodic components, resampling of the signal can be very helpful to filter these components [23]. Note that aliasing effects may disturb the spectrum of a down sampled signal. Most of the mentioned filters are applicable to non stationary problems, they however cannot focus on specific frequencies. The Kalman filter is an exception to this. The covariances assumed in the filter can be adapted such that the filter addresses specific frequencies. A field where Kalman filtering often is used is speech enhancement [14]. In this field both white and colored noise are successfully filtered using the Kalman filter. To apply a Kalman filter the variance of the noise has to be known: a time interval without an active set-up could be used to determine estimates for the variance, which can be used in the filter [47]. Xia and Wei [47] introduce a model prediction that is based on the same measurements used for the measurement update. By rewriting the problem this model prediction is 'whitened': the model is corrupted by white noise in stead of the measured colored noise. This allows the Kalman filter to make a better estimation of the underlying signal. This method of Xia and Wei [47] shows good results for speech enhancement. Speech signals are considered periodic, where the measurement signals as presented in Chapter 2 are not. This might influence the behaviour of the Kalman filter.

7.2.2. Transients and discontinuities

The detection of transients and discontinuities (or edges in signal analysis) is very important in most engineering fields. Any discontinuity measured by a sensor may characterize an event [27]. *Transients* are always of short duration and unpredictable nature, changing frequency over time and they often decay fast: the parameters of the transient and its arrival time are unknown. In the Fourier domain the STFT is used to determine the location of the transient. The trade-off between time and frequency chosen beforehand influences the detection of a certain transient. The time frequency representation of the WT on the other hand enables exact localization of any abrupt change, impossible for the STFT. This is not only applicable to transients, but also to discontinuities. Transients usually only appear in the lower scales of the WT, whereas the higher scales represent the low frequency basis of the signal. The choice of wavelet is very important in the detection of transients: remember the WT coefficients represent the correlation between the transient and the wavelet function used. The detection will improve when the the shape of the transient and wavelet are similar [27].

The scales that need to be considered for detecting transients are dependent on the size of these transients. The amplitude of a transient often is rather low in comparison to the signals amplitude. This brings down the computational load for transient detection; only a few scales have to be computed. The lowest scales contain information about the discontinuities and high frequency noise of the signal, transient detection most of the time is done from scale 4 and up [27]. But for shorter signals this might be in higher scales. To detect transients in the presence of noise can be difficult, it may be necessary to first apply some thresholding and then do transient detection on the reconstructed signal. Discontinuities are recognizable by their high detail coefficients over a lot of decomposition levels, as mentioned before [18]. Especially the Haar wavelet had good discontinuity recognizing properties, because it is a discontinuous wavelet.

Example 7.3 (Transient detection using wavelets) To show how wavelet transforms detect transients, a short example is given. In Figure 7.5a a very small transient is placed on a sine wave. In the spectrum of this signal, the transient cannot be detected. Also a spectrogram gives little insight in the place of the spectrum. If one level of wavelet decomposition is applied (see Figure 7.5b) the place of the transient is immediately clear from the coefficients in the detailed part. The reconstruction of the transient is very similar to the original transient.

The Python-code for this example: Listing B.9.

7.2.3. Non linear and non stationary signals

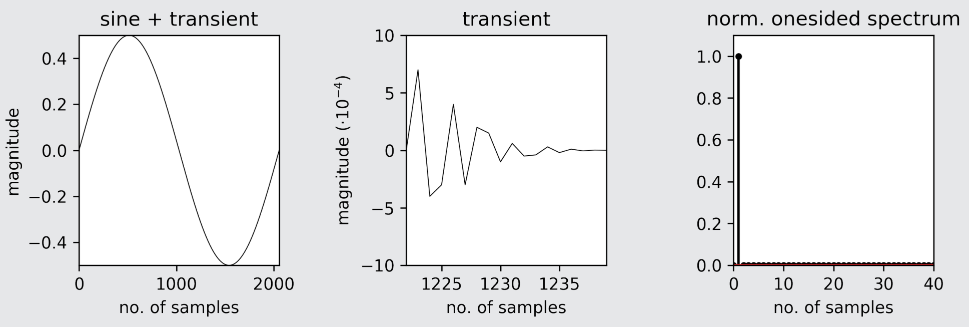
Finally the two last signal elements are discussed. The first one to discuss is are te non linear elements. As described before, linear functions or systems comply to (3.2):

$$f(A\boldsymbol{x} + B\boldsymbol{y}) = Af(\boldsymbol{x}) + Bf(\boldsymbol{y}).$$

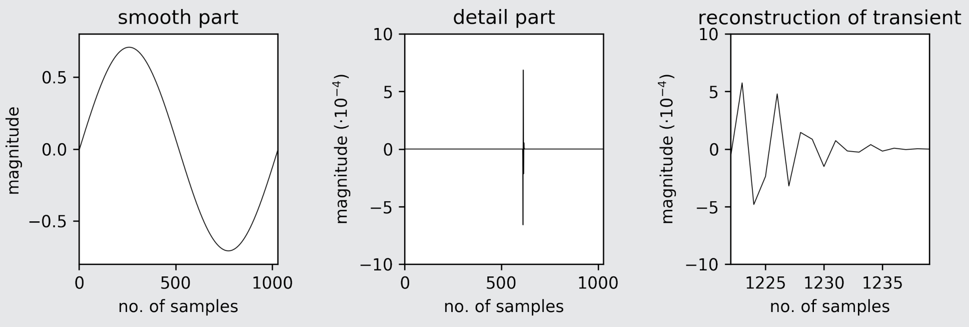
As mentioned before, both Fourier and wavelet transform are linear, which results in a linear representation of the transformed signals. This linearity comes from the basis of an analysis as discussed in Chapter 3: an analysis is seen as a superposition of deterministic functions with different weights. The functions are complex exponents in the Fourier case and short waves in the wavelet case, the different weights are known as coefficients. Non linear elements are very common in nature. They are often described by non linear differential equations, which do not always have a finite representable solution [42]. The mathematical description using harmonic components or wavelet functions does have a mathematical meaning, however physically this often has no meaning [16]. The second signal element to discuss are the non stationary signal elements. The concept of stationarity is already explained in Section 7.1.

To analyse non linear, non stationary data the Hilbert Huang transform has been developed by NASA in the late 90s. Applications of this transform emerged around 2005 [17, 28]. This method combines an empirical mode decomposition with Hilbert spectral analysis [16]. The signal is first decomposed in so-called empirical modes: simple harmonic components which can have variable amplitude and frequency in time. These modes are then analyzed using Hilbert spectral analysis, which computes instantaneous frequencies easily. The Hilbert transform of a signal $x(t)$, $\mathcal{H}\{x(t)\}(i\omega)$ is defined by[15]:

$$\mathcal{H}\{x(t)\}(\omega) := \frac{1}{\pi}\mathcal{F}\{x(t)\}(\omega) * \frac{1}{\omega} = \frac{1}{\pi}\int_{-\infty}^{\infty} \frac{X(\eta)}{\omega - \eta}\, \mathrm{d}\eta.$$



(a) On a sinusoidal signal a small transient is placed (at sample 1225). In the normalized onesided spectrum of this signal the transient is not visible. The spectrum on the right is therefore truncated; the '1 oscillation per frame' sine is visible.



(b) The signal from part (a) is decomposed using a Daubechies 4 wavelet. Only one decomposition step is computed. Note the difference in y-scale in the signal and transient plots. The smooth part contains the sine, where the detail part shows a peak at the place of the transient. The reconstruction of the transient in these ideal circumstances is close to perfect.

Figure 7.5: Transient detection and reconstruction using wavelets

In contrary to the Fourier and wavelet transform, the basis of the decomposition, the empirical modes, is adaptive. The instantaneous frequency is a highly controversial definition, and inceptive: when something is instantaneous, it is localized in time. However, time and frequency are inverse quantities, resulting in the ambiguity [12]. The resulting algorithm, the called the Hilbert-Huang transform has a lot of empirical support [16, 17, 28]. An overview of the comparison of the three methods is given in Table 7.1.

	Fourier	**Wavelet**	**Hilbert-Huang**
Basis	a priori:	a priori: $\psi(t)$	adaptive
Frequency	convolution: global uncertainty	convolution: regional uncertainty	differentiation: local, certainty
Presentation	energy frequency STFT: energy-time-frequency	energy-time-frequency	energy-time-frequency
Non linear	no	no	yes
Non stationary	no	yes	yes
Feature extraction	no	discrete: no, continuous: yes	yes
Theoretical base	theory complete	theory complete	empirical base
Computation time	FFT: $\mathcal{O}(N\log_2 N)$ STFT: $\mathcal{O}(N^2\log_2 N)$	CWT: no fast algorithm DWT (all scales): $\mathcal{O}(N^2\log_2 N)$	$\mathcal{O}(N\log_2 N)$

Table 7.1: Comparative summary of Fourier, wavelet and HHT analyses from [16].

This all sounds very promising, however the great drawback of this method is the lack of (mathematical) theoretical base. This is in large contrast with Fourier and wavelet analysis, which both have an elegant mathematical framework, very suitable for model building [12]. Huang and Shen [16] sums up a lot of challenges, among others: non linear system identification, optimization and approximation problems.

7.3. Application to the problem

In Chapter 2 different problems with respect to measurement results have been addressed. In this section the different applications and possibilities to process the presented measurements are briefly discussed. This section consists of the following components: noise reduction, linearity and stationarity, the multichannel approach and some wavelet analysis specific considerations.

Noise reduction

The plots of the discussed signals from Chapter 2 can be found in Appendix C. The displayed signals are distorted by noise. To give an insight in the different sources of noise, the spectra of the example signals was already shown in Figure C.2. Additive noise contributions can exist of stationary or non stationary and colored or white noise. Stationary white noise can be filtered using both Fourier and wavelet analysis. To filter non stationary noise, wavelet analysis is recommended. To test whether noise is Gaussian white or not, there are a lot of options. Gaussian white noise in statistics is known as a realization of an independent and identically distributed or *iid* process.

- The autocorrelation of the samples could be used to test a white noise assumption [5]. The autocorrelation of an iid variable sequence $Y_1, \dots, Y_n$ with finite variance are approximately iid with distribution $\mathcal{N}(0, 1/n)$. So in 95% of the cases the sample autocorrelation should fall within the bounds $\pm 1.96/\sqrt{n}$. The iid hypothesis is dropped if up to lag 40 more than three autocorrelation values fall outside these bounds, and even if only one lies far outside these bounds [5].

- The *portmanteau test* uses, instead of checking each sample the autocorrelation on the given borders, the single statistic based on the autocorrelation of the signal $\hat{\rho}(j)$ [5]

$$Q(h) = n \sum_{j=1}^{h} \hat{\rho}^2(j).$$

  The iid hypothesis is rejected at a level α if $Q(h) > \chi^2_{1-\alpha}(h)$, where $\chi^2_{1-\alpha}(h)$ is the $1-\alpha$ quantile of the chi-squared distribution with h degrees of freedom.

  In the *Ljung-Box test* $Q(h)$ is replaced by an expression whose distribution is better approximated by the chi-squared distribution. An other test is the *McLeod-Li test* which uses the same statistic as Ljung-Box, except for the use of the autocorrelation of the squared data [5].

- The *turning point test* counts the number of turning points in the sample. A too small or too large number of turning points indicated the rejection of the iid hypothesis [5].

- The *difference-sign test* counts, instead of the number of turning points, the number of times the differenced series $y_i - y_{i-1}$ is positive [5].

- Different visual techniques can be used as well, although these are less reliable. The QQ-plot is the best known example of displaying correlation.

The most used test is the Ljung-Box text [5], which also has user friendly implementation in both Matlab [22] and Python [29].

The data presented in Chapter 2 (Figure C.2) does however show very little white noise. At specific frequencies, distinct peaks are detected: overall it can be stated that colored noise has a larger contribution to the noisy signal than white noise. The 50 Hz peak is known to originate from the power supply, and will always be there. For this power supply noise contribution the implementation of a notch filter seems the most logical. Note that the frequency of the power supply can deviate a little bit, and therefore the specific tuning of the filter may differ. For the other frequency peaks a same type of filtering is applicable, however this will result in (a lot of) manual work. There are three obvious noise filtering algorithms to apply. The first one is to use a low pass filter to cut of all high frequency parts of the signal, which always will result in Gibbs ripples and smoothening of jumps in the signals. This is the computational least intensive method. The second one is to apply wavelet filtering. We have seen some promising results already, however a bit more advanced coefficient filter should be applied to ensure the filtering of the right frequencies.

The filter proposed by Srivastava et al. [35] (Figure 7.4b) shows promising results: the computational load of the filtering is of $\mathcal{O}(N)$, which results in an filtering algorithm from signal to filtered signal of $\mathcal{O}(N + N^2 \log_2 N)$. Moreover, the use of wavelet packets to apply colored noise filtering is a more promising technique than the standard wavelet filtering. The final option is application of one of the 'smoothing filters' or the Kalman filter from page 67. The filter as proposed by Xia and Wei [47] without updating the noise parameters should be much less computational intensive then the wavelet filtering algorithm. When the parameters need to be updates, there will be almost no difference in computational load. In the wavelet domain filters such as the Kalman filter can also be applied, this might be interesting for further research.

Linearity and stationarity

The signals discussed in Chapter 2 have both non linear and non stationary elements. This could be both noise and part of the desired signal. For instance in the water height measurement in Figure C.3 at around 15.08 minutes a large amount of water is added to the basin, this results in sloshing waves: a non stationary process. This does not take for ever; before the jump the water level is constant. In the other measurement the resonance frequency of the tested construction is expected to also show non stationary influences. A linear model for the behavior as shown in Figure C.1 probably does not exist, however the goal is not to build a model to mimic this behavior, but to denoise the signal and determine the influence of effects of the measurement setup. Non linear effects do not have to be taken into account to successfully retrieve the signal.

A wish of the data analysts is to get more insight in the (non stationary) frequencies present in the noise free signal. Especially the influence resonance frequencies on the measurement are interesting to them. For they work with scale models the resonance frequency in the measurement can differ a lot from the life size application. These effects will ultimately have to be filtered out. For these frequencies are non stationary they are most practically filtered in the wavelet domain.

A multi-channel approach

In the first sections of this thesis the focus of the noise reduction was on so called *single-channel* noise reduction methods. These methods all assume the availability of just one sensor. In *multi-channel* noise reduction methods, the input of different sensors is used to suppress noise [47]. A good example is a modern mobile phone, which often has two or more microphones to record your voice when calling, to be able to suppress noise better. This method is well known from speech enhancement, however multi-channel noise reduction can have some application in this field as well. In multi-channel approaches often filters such as the Kalman filters are used. But there are also possibilities in the wavelet domain.

The applicability of wavelets

In the introduction was stated that the application of wavelets is Deltares' primal interest. In this subsection some advantages, disadvantages, applications and considerations of wavelet analysis will be highlighted. The largest advantage is that wavelet analysis has high applicability in non stationary signal analysis and noise reduction for non periodic signals which frequently occur in the field of coastal engineering. The wavelet theory is a bit more complicated than the general Fourier technique, which makes it harder to apply. There are however possibilities in automatizing wavelet filter application [35], more than in the Fourier analysis. Actually it is best to design a wavelet to decompose a specific signal [35]. Both the similarity of the wavelet and the signal in time [37] as the similarity of the spectra of the wavelet and the signal [35] are good indicators for the effectiveness of the decomposition, and thus the algorithms. Cortés et al. [6] notes the Symmlet 8 as a traditional mother wavelet for denoising natural signals. However, for specific applications, different wavelets may be recommended. Orthogonal wavelets, especially the Haar wavelet, are best applicable for discontinuity detection [40]. Their filtering properties differ from the biorthogonal wavelets, by using less decomposition levels. For fast frequency information insight the scalogram is not recommended. The spectrogram is easier to apply and to read. However, some proposals by Torrence and Compo [40] should reduce some reading problems. Moreover wavelet analysis can be extended further to multiwavelets and wavelet packets. These extensions in wavelet theory both show even better results in signal processing. Over all we may conclude that there are enough possibilities to apply wavelets in data analysis in the coastal engineering field and extend the analysis if results are non satisfactory.

8 Summary and conclusion

This chapter concludes this literature study with a research proposal for the master thesis. First we start with a short recap of all chapters in this thesis. In this proposal research questions are stated that will be answered in the masters thesis. Thereafter some details concerning the process of the master thesis will be addressed.

This thesis started with a short overview of measurements conducted at Deltares. These measurements are just a few of a vast number of time series in coastal engineering research, which are distorted by different effects. The most important effects are non stationary noise contributions and behaviors of the measurement setup. In the three chapters thereafter, three different signal analysis methods are elucidated: the Fourier transform (Chapter 4), the wavelet transform (Chapter 6) and an adaptation of the Fourier transform, the short term Fourier transform (Chapter 5). Finally in Chapter 7 the applications of these different analyzing techniques were discussed.

Fourier analysis

Real word data most often contains slowly changing trends (or oscillations) and abrupt changes, and almost always contains a certain amount of noise. Fourier analysis is a perfect method to find these slowly changing trends and is a mathematically stable and proven method. The Fourier Transform (FT) represents a signal as an infinite sum of complex exponentials (which can be rewritten to sines and cosines). These exponentials have one exact frequency, but therefore they lack localization in the time domain. If a finite sum of sines and cosines is used to describe a discrete signal, the transformation is called the discrete Fourier transform. When using the short term Fourier transform (STFT) technique the Fourier transform is applied to shorter parts of the signal. By choosing those parts wisely an overview of the presence of different frequencies at different times can be given. However a trade-off has to be made. By the Heisenberg uncertainty principle (Equation 5.5) localization in time is inversely proportional to frequency localization. The result: if one wants a better frequency resolution, the consequence is a decrease in time resolution (and vice versa).

Wavelet analysis

The wavelet transform (WT) represents a signal in terms of a short wave like signals, wavelets, instead of the infinite complex exponential waves. This results in an analysis with localization in both time and frequency, similar to the STFT. A short wave like function $\psi(t)$ is suitable as wavelet if it complies to at least these two requirements (see Section 6.1):

- Compact support: the signal is of finite duration;
- $\int\limits_{-\infty}^{\infty} \psi(t)\,\mathrm{d}t = 0$ and $\int\limits_{-\infty}^{\infty} |\psi(t)|^2\,\mathrm{d}t = 1$: to ensure an equal energy distribution.

The wavelet transform coefficient of a signal is dependent of two variables: the scale a and translation b. The scale (Equation 6.2) adjusts the length in time of the wavelet. A high scale 'stretches' the wavelet, increasing its duration and lowering its frequencies. High scaled wavelets have a smaller frequency spread which lie in the lower frequency regions, they also have less localization in time. The low scaled wavelets are shorter in time, which increases their accuracy in time. Moreover they support a larger bandwidth in higher frequency regions, in agreement with the Heisenberg uncertainty principle . To choose the right wavelet to analyze your signal, three characteristics have to be taken into account (see Section 6.2): the number of vanishing moments, the regularity and the selectivity of the wavelet. Often the resemblance of the wavelet and the signal to analyze are a good reference of the applicability of that wavelet. However, there are many possibilities to design a wavelet which is suitable for a specific analysis.

The information above is general information for both the continuous as the discrete wavelet transform. For the discussed application: signal analysis, the discrete wavelet transform (DWT) is more suitable. Instead of a continuous scale and translation, in this literature thesis the discrete scales of 2^k for $k \in \mathbb{Z}$ are chosen. The discrete translations depend on the chosen wavelet. The discrete wavelet transform is based on multi resolution analysis (Section 6.3) which provides a solid base for the algorithmic implementation of the DWT.

Digital signal processing

In Chapter 7 a set of common applications of these transforms are discussed: noise reduction; detection of transients, discontinuities and specific frequencies; and the applicability to non linear and non stationary signals. There we have seen that the simplicity of the Fourier transform, giving only insight in the frequency domain, has its advantages and disadvantages. The largest advantage are the comprehensibility (by engineers), the easy filter design and the applicability by modern computers. The main disadvantage of signal processing in the frequency domain is that any change to the spectrum of a signal will effect the whole signal, whereas signal processing in the STFT or the wavelet domain allows for the effects reduces to certain intervals. Non stationary effects therefore cannot be filtered in the Fourier domain, furthermore discontinuities for instance will be smoothened when filtering in the frequency domain is applied. By switching to the STFT some disadvantages are removed. However signal processing using the STFT might become a tedious job, because often the filters in the frequency domain have to be hand picked. Therefore this transform is not used much in situations alike our application. The (discrete) wavelet transform does not have the same disadvantages as the Fourier transform; it is recommended to use for non stationary and discontinuous signals.

A disadvantage of the use of wavelets is that one has less influence on the time or frequency spread of a wavelet. This can be solved partially by using wavelet packets instead of regular wavelets: wavelet packets are used to address small frequency bands (this however results in less localization in time). Furthermore a lot of choices have to be made to apply filtering using wavelet analysis. These however can (partially) be automated. A even newer technique is reviewed in Section 7.2.3: the Hilbert-Huang transform. The published results concerning this transform look very promising, however it only has an empirical base, which is in great contrast with the mathematical base of both Fourier and wavelet analysis. Therefore this transformation will not be part of the research.

The last important characteristic of these transforms is their algorithmic speeds. For implementation on the computer of quite long signals, this is important, especially if transforms have to be adjusted or cascaded to apply optimal filtering. Let us assume a discrete signal of length N. The fast Fourier transfrom (FFT) algorithm is the fastest discrete implementation of the discrete Fourier transform, which takes $\mathcal{O}(N \log_2 N)$ computations. To compute the STFT the number of computations is of $\mathcal{O}(N^2 \log_2 N)$. To compute one scale of the discrete wavelet decomposition, a computation of $\mathcal{O}(N \log_2 N)$ is performed. To do a full scale decomposition $\mathcal{O}(N \log_2 N)$ computations are needed.

So, despite the irregular shapes of wavelets, perfect reconstruction is possible for linear and higher order polynomial shapes. In Fourier analysis always some Gibbs effects will appear. Furthermore automatic thresholding in wavelet analysis is possible. For filtering of specific frequencies however, wavelet analysis definitely is not the best choice. The main difference of the wavelet approach with respect to the Fourier approach in signal processing is that the wavelet approach is more of a ’decomposition’ rather than a filter. The reason wavelets are being used more and more is because they are capable of deconstructing complex signals into basis signals of finite bandwidth, and then reconstructing them again with very little loss of information. Wavelet decomposition suffer less to no signal leakage or phase-shifting of the original signal when decomposed. Conventional filters in the Fourier domain generally have problems with signal leakage or phase-shifting that have to be dealt with.The best known practical application of this characteristic is compression. Here there is more interest in the noise filtering qualities.

8.1. Thesis proposal: research questions

The summary has given a problem sketch and an overview of the different time series analysis techniques which can be used to address the problem. There is a large number of possible improvements, however the project has to be specified to the project duration of six months. Therefore the main research question of my masters thesis will be:

How can wavelet analysis improve time series analysis

in the field of coastal engineering?

To answer this question, the following subquestions shall guide the research:

1. How to recognize and identify different types of noise in coastal engineering time series?
2. How to remove or reduce different types of noise efficiently in coastal engineering time series?
3. What is the added value of wavelet analysis over current time series analysis methods in coastal engineering?

Subquestions

The first subquestion focuses on the signal analysis through spectra, spectrograms and scalograms and how different types of noise can be recognized. In noise identification the challenge is to separate contributions to the noise by the measurement equipment, such as the power supply, and contributions which are considered component of the measured signal, such as resonance effects. This field connects seamlessly to the noise filtering field, which is addressed in the second question. The noise filtering algorithms to implement are limited to a number of algorithms. These algorithms will be implemented on a set of test cases, discussed in Section 8.1.1. The performance of these algorithms will be rated based on their noise reducing ability, computational speed, and discontinuity preserving characteristics for different signals. These algorithms are:

- *Frequency domain filtering, a combination of the Hanning window and notch filters*, will be used as a reference because it is the most used method in the field nowadays.
- *Wavelet filtering, both hard and soft thresholding* [1] are early wavelet thresholding techniques, which will also serve as a reference.
- *The semi-automated wavelet filtering algorithm as proposed by Srivastava et al. [35]* is a new algorithm which has been empirically approved and can be automatized.
- *Wavelet filtering based on a signal free period* as proposed by Srivastava et al. [35].
- *The Kalman filtering algorithm presented by Xia and Wei [47]* is a different approach, however its first results look promising and computation times are expected to be less than the other algorithms, because no transformation is involved. It is good applicable to signals disturbed by correlated noise, such as the example signals from Chapter 2.

As mentioned before noise filtering is a very extensive field. Based on the results of the algorithm possible further improvements could answers to these questions (without further elaboration):

- How does the choice of wavelet influence the result of the wavelet filtering algorithms?
- How can results from different sensors in the same measurement, i.e. a multi channel approach, be used to improve noise identification and filtering?
- Is there need to implement a more sophisticated wavelet analysis, i.e. wavelet packets or multi wavelets?
- Is there need to extend the wavelet filtering algorithms with a shifting or cycle spinning algorithm?
- Does cascading algorithms from different domains improve the performance?
- Can wavelet analysis be used to separate incoming and reflected waves?

The answer to the last subquestion should endorse the added value of wavelet analysis in the field of coastal engineering research. Based on conversations with people from the coastal engineering research field the answer should make clear why it is better to use wavelet analysis, if so. This subquestion is important to answer the main research question: a clear added value will prevent users to fall back to their known, regular data analysis methods. An important step in answering this question will be the accessibility of wavelet analysis, especially for engineers who do not have any knowledge about wavelet analysis. Also the applicability of different wavelets for specific purposes will be elaborated, possibly combined with some testing from the second subquestion. Further the possibility of fully automatized choice of wavelet and noise threshold could be investigated. This might lead to an extra algorithm to test.

8.1.1. Test cases

The test cases are used to test the algorithms and their performances. Therefore the algorithms will first be tested on artificial problems to quantify the performance of these different algorithms. In this simulation of a measurement signal, the SNRs can be adjusted to describe different measurement circumstances. The simulation signal should consist of a basis signal:

- A stationary signal, such as wave height measurements.
- A non stationary signal, for instance a signal with large peaks such as measurements by force meters.

This basis signal can have contributions of colored, non stationary signal elements which for instance resemble resonance effects. To this signal some noise will be added. The noise can consist of 4 elements.

- A Gaussian white noise contribution;
- A contribution of colored noise (within a bandwidth), or of a specific 'color' such as red noise;
- A contribution of colored noise at a few specific frequencies (e.g. 50 Hz and mirrors);
- A contribution of colored non stationary noise.

The colored noise can be chosen such that it interferes with the signal or not. The challenge lies in the ability to filter the noise, but not filter the non stationary or colored effects in the signal. For these noise choices four different cases are defined: noise absent, SNR= 0 dB, 5 dB and 10 dB. This leads to a set of 16 different settings per basis signal which can be tested. During the research some settings can be discarded based on earlier findings. The test signal will span two minutes, sampled at 3 kHz and all start with a idle period of 10 seconds.

Signals from measurements

The known behavior to the test case will then be applied to measurements conducted at Deltares. The expert eyes of the data analysts at Deltares will have to judge the effectiveness of the used method. Therefore these sets over measurements will be leading:

- A pressure signal with peaks
- A force signal with peaks
- A periodical water height measurement

8.1.2. Project details

This graduation project will be fulfilled at Deltares. The main programming language to implement wavelet analysis will be Python, because the Wavelet Toolbox of MathWorks is not available for Deltares employees. The other programming tasks will be conducted in Matlab. My supervisor from the Delft University of Technology will be prof. dr. ir. Kees Vuik, at Deltares ir. Jan Kramer will be my daily supervisor.

Further research options

For further research and a broader application of wavelets in coastal engineering research, the following topics can be addressed. These are not further elaborated.

- Can sources of the noise contributions be explained? Can the measurements or the equipment be optimized to suppress the noise?
- How can the spectrum, spectrogram or the scalogram be used effectively for signals in coastal engineering research?
- How can the computation times of the algorithms be improved?
- Does wavelet analysis have the same added value for two dimensional results in coastal engineering research?
- How does Hilbert Huang Transform filter perform in comparison with the other tested filters?

Bibliography

[1] Anestis Antoniadis. Wavelet methods in statistics: some recent developments and their applications. Statistics Surveys, 1:16–55, 2007. ISSN 1935-7516. doi: 10.1214/07-SS014. URL http://arxiv.org/abs/0712.0283{%}5Cnhttp://dx.doi.org/10.1214/07-SS014{%}5Cnhttp://projecteuclid.org/euclid.ssu/1196693422.

[2] Gonzalo R. Arce. Nonlinear Signal Processing: A Statistical Approach. John Wiley & Sons, inc., 2005. ISBN 0471691844. doi: 10.1002/0471691852.ch4. URL http://books.google.com/books?hl=en{&}lr={&}id=4DKZH0aPXMMC{&}pgis=1.

[3] Alex Bellos. Abel Prize 2017: Yves Meyer wins 'maths Nobel' for work on wavelets | Science | The Guardian. https://www.theguardian.com/science/alexs-adventures-in-numberland/2017/mar/21/abel-prize-2017-yves-meyer-wins-maths-nobel-for-work-on-wavelets, 2017. Date accessed: 24-3-2017.

[4] Christian Blatter. Wavelets: A Primer. A K Peters, Ltd, 1998. ISBN 9781568811956.

[5] Peter Brockwell and Richard Davis. Introduction to Time Series and Forecasting. Springer, 2002. ISBN 0387953515. doi: 10.2307/1271510. URL http://scholar.google.com/scholar?hl=en{&}btnG=Search{&}q=intitle:Introduction+to+Time+Series+and+Forecasting{#}0{%}5Cnhttp://www.jstor.org/stable/1271510?origin=crossref.

[6] Camilo Cortés, Francisco Santamaría, Francisco Roman, Farhad Rachidi, and Chandima Gomes. Analysis of wavelet based denoising methods applied to measured lightning electric fields. 30th International conference on lightning protection - ICLP 2010, 2010:1–6, 2010. doi: 10.1109/ICLP.2010.7845950.

[7] Leon W. Couch. Digital and Analog Communication Systems. Pearson Prentice Hall, 7th edition, 2007. ISBN 0-13-203794-7.

[8] creativecommons.org. Time-Frequency Dictionaries. http://archive.cnx.org/contents/4a47d3df-bfe5-428c-8bb6-e5b738077caa@2/time-frequency-dictionaries, 2017. Date accessed: 20-2-2017.

[9] Ingrid Daubechies. Orthonormal bases of compactly supported wavelets. Communications on Pure and Applied Mathematics, 41(7):909–996, oct 1988. ISSN 10970312. doi: 10.1002/cpa.3160410705. URL http://doi.wiley.com/10.1002/cpa.3160410705.

[10] Ingrid Daubechies. The Wavelet Transform, Time-Frequency Localization and Signal Analysis. IEEE Transactions on Information Theory, 36(5):961–1005, 1990. ISSN 15579654. doi: 10.1109/18.57199.

[11] Ingrid Daubechies. Ten Lectures on Wavelets. Society for Industrial and Applied Mathematics, Pennsylvania, 1992. ISBN 0-89871-274-2.

[12] Jaidev Deshpande. PyHHT Tutorials – pyhht 0.0.1. documentation. http://pyhht.readthedocs.io/en/latest/tutorials.html, 2017. Date accessed: 29-3-2017.

[13] Ramsey Faragher. Understanding the basis of the kalman filter via a simple and intuitive derivation [lecture notes]. IEEE Signal Processing Magazine, 29(5):128–132, 2012. ISSN 10535888. doi: 10.1109/MSP.2012.2203621.

[14] Jerry D. Gibson, Boneung Koo, and Steven D. Gray. Filtering of Colored Noise for Speech Enhancement and Coding. IEEE Transactions on Signal Processing, 39(8):1732–1742, 1991. ISSN 19410476. doi: 10.1109/78.91144.

[15] Bernd Girod, Rudolf Rabenstein, and Alexander Stenger. Signals and systems. John Wiley & Sons, inc., 2001. ISBN 0-471-98800-6.

[16] Norden E. Huang and Sauuel S. P. Shen. Hilbert Huang Transform and Its Applications, volume 16. World Scientific Publishing Co. Pte. Ltd., second edition, 2014. ISBN 978-9814508230. doi: 10.1007/s13398-014-0173-7.2. URL `http://www.worldscientific.com/worldscibooks/10.1142/8804`.

[17] Norden E Huang and Zhaohua Wu. A Review on Hilbert-Huang Transform: Method and Its Applications to Geophysical Studies. Reviews of Geophysics, 46(2007):1–23, 2008. ISSN 87551209. doi: 10.1029/2007RG000228.1.INTRODUCTION. URL `http://rcada.ncu.edu.tw/reference010.pdf`.

[18] Fritz Keinert. Wavelets and Multiwavelets. Chapman & Hall/CRC, 2004. ISBN 1-58488-304-9.

[19] Gihyoun Lee, Sung Dae Na, KiWoong Seong, Jin-Ho Cho, and Myoung Nam Kim. Speech Enhancement Algorithm Using Recursive Wavelet Shrinkage. Ieice Transactions on Information and Systems, E99D(7): 1945–1948, 2016. ISSN 1745-1361. doi: 10.1587/transinf.2015EDL8251.

[20] Chun-Liu Liu. A Tutorial of the Wavelet Transform. National Taiwan University, Department of Electrical Engineering (NTUEE), pages 1–72, 2010. ISSN 0166526X. doi: 10.1111/j.1600-0404.1995.tb01711.x. URL `http://disp.ee.ntu.edu.tw/tutorial/WaveletTutorial.pdf`.

[21] Stéphane Mallat. A Wavelet Tour of Signal Processing. Academic Press, 2nd edition, 1999. ISBN 0-12-466606-X.

[22] MathWorks Benelux. Ljung-Box Q-test for residual autocorrelation - MATLAB lbqtest - MathWorks Benelux. `http://nl.mathworks.com/help/econ/lbqtest.html`, 2017. Date accessed: 5-4-2017.

[23] MathWorks Benelux. Signal Smoothing - MATLAB & Simulink Example - MathWorks Benelux. `https://nl.mathworks.com/help/signal/examples/signal-smoothing.html`, 2017. Date accessed: 7-4-2017.

[24] MathWorks Benelux. Remove the 60 Hz Hum from a Signal - MATLAB & Simulink - MathWorks Benelux. `https://nl.mathworks.com/help/signal/ug/remove-the-60-hz-hum-from-a-signal.html`, 2017. Date accessed: 10-4-2017.

[25] MathWorks Benelux. Complex Morlet wavelet - MATLAB cmorwavf - MathWorks Benelux. `https://nl.mathworks.com/help/wavelet/ref/cmorwavf.html`, 2017. Date accessed: 6-3-2017.

[26] MathWorks Benelux. Wavelet Toolbox - MATLAB. `https://nl.mathworks.com/products/wavelet.html`, 2017. Date accessed: 27-2-2017.

[27] SV Narasimhan, N Basumallick, and S Veena. Introduction to wavelet transform: a signal processing approach. Alpha Science International Ltd., 2011. ISBN 978-1-84265-629-7.

[28] Z. K. Peng, Peter W. Tse, and F. L. Chu. A comparison study of improved Hilbert-Huang transform and wavelet transform: Application to fault diagnosis for rolling bearing. Mechanical Systems and Signal Processing, 19(5):974–988, 2005. ISSN 08883270. doi: 10.1016/j.ymssp.2004.01.006.

[29] Taylor, Jonathan Perktold, Josep, Seabold, Skipper. statsmodels.stats.diagnostic.acorr_ljungbox – statsmodels 0.6.0 documentation. `http://statsmodels.sourceforge.net/0.6.0/generated/statsmodels.stats.diagnostic.acorr_ljungbox.html`, 2017. Date accessed: 6-4-2017.

[30] Roger Peyret. Spectral Methods for Incompressible Viscous Flow. Springer Science+ Business Media, New York, 2002. ISBN 978-1-4419-2913-6. doi: 10.1007/978-1-4757-6557-1.

[31] John A. Rice. Mathematical Statistics and Data Analysis. Thomson Brooks/Cole, third edition, 2007. ISBN 0-534-39942-8.

[32] Allen L Ronald and Duncan W Mills. Signal Analysis Time, Frequency, Scale and Structure. John Wiley & Sons, inc., 2004. ISBN 9786468600. doi: 10.1002/047166037X.

[33] Sujan Kumar Roy, Wei Ping Zhu, and Benoit Champagne. Single channel speech enhancement using subband iterative Kalman filter. Proceedings - IEEE International Symposium on Circuits and Systems, 2016-July:762–765, 2016. ISSN 02714310. doi: 10.1109/ISCAS.2016.7527352.

[34] David K. Ruch and Patrick J. Van Fleet. Wavelet Theory: An Elementary Approach with Application. John Wiley & Sons, inc., 2011. ISBN 9780470388402. doi: 10.1002/9781118165652.

[35] Madhur Srivastava, C. Lindsay Anderson, and Jack H. Freed. A New Wavelet Denoising Method for Selecting Decomposition Levels and Noise Thresholds. IEEE Access, 4:3862–3877, 2016. ISSN 21693536. doi: 10.1109/ACCESS.2016.2587581.

[36] G. Strang and T. Nguyen. Wavelets and Filter Banks. Wellesley-Cambridge Press, 2003. ISBN 0961408871.

[37] W Sweldens. The Lifting Scheme: A New Philosophy in Biorthogonal Wavelet Constructions. 2007. URL papers://b6c7d293-c492-48a4-91d5-8fae456be1fa/Paper/p7558.

[38] The PyWavelet Developers. Wavelets – PyWavelets Documentation. http://pywavelets.readthedocs.io/en/latest/ref/wavelets.html, 2017. Date accessed: 17-3-2017.

[39] Albert C. To, Jeffrey R. Moore, and Steven D. Glaser. Wavelet denoising techniques with applications to experimental geophysical data. Signal Processing, 89(2):144–160, 2009. ISSN 01651684. doi: 10.1016/j.sigpro.2008.07.023.

[40] Christopher Torrence and Gilbert P. Compo. A Practical Guide to Wavelet Analysis. Bulletin of the American Meteorological Society, 79(1):61–78, 1998. ISSN 00030007. doi: 10.1175/1520-0477(1998)079<0061:APGTWA>2.0.CO;2. URL http://atoc.colorado.edu/research/wavelets/.

[41] J van Kan, A Segal, and F Vermolen. Numerical methods in scientific computing. Delft Academic Press / VSSD, second edition, 2014. ISBN 97890-6562-3638. URL https://scholar.google.nl/scholar?hl=nl{&}q=vermolen+numerical+methods+in+scientific+computing{&}btnG={&}lr=.

[42] Ferdinand Verhulst. Nonlinear differential equations and dynamical systems. Springer, second edition, 1993. ISBN 978-3-540-60934-6. URL https://books.google.nl/books?hl=nl{&}lr={&}id=oyaFXgcFJ2cC{&}oi=fnd{&}pg=PA1{&}dq=nonlinear+differantial+equations+systems+verhulst{&}ots=5O8iWikygE{&}sig=fgwM{_}ZCFByHXvdhfb2s87EzZWN8.

[43] David F. Walnut. An Introduction to Wavelet Analysis. Birkhauser, 2002. ISBN 0-8176-3962-4.

[44] Filip Wasilewski. Discrete Meyer (FIR Approximation) wavelet (dmey) properties, filters and functions - Wavelet Properties Browser. http://wavelets.pybytes.com/wavelet/dmey/, 2017. Date accessed: 17-3-2017.

[45] Wikipedia. Comparison convolution correlation - Convolution - Wikipedia. https://en.wikipedia.org/wiki/Convolution#/media/File:Comparison_convolution_correlation.svg, 2017. Date accessed: 3-3-2017.

[46] Peter Wilders and W.A. Heemink. Transport Modelling and Data Assimilation - WI4054. Delft University of Technology, 2016.

[47] Y Xia and Q Wei. An effective Kalman filtering method for enhancing speech in the presence of colored noise. Audio, Language and Image Processing (ICALIP), 2016. URL http://ieeexplore.ieee.org/abstract/document/7846530/.

List of Figures

A Fourier transforms

A.1. CFT - Continuous Fourier transforms

Property	**Time signal** $f(t)$	**Fourier transform** $\mathfrak{F}f(\omega) = F(\omega)$	
Linearity	$c_1 f_1(t) + c_2 f_2(t)$	$c_1 F_1(\omega) + c_2 F_2(\omega)$	(A.1)
Translation	$f(t-h)$	$e^{-i\omega h} F(\omega)$	(A.2)
Modulation	$e^{i\xi t} f(t)$	$F(\omega - \xi)$	(A.3)
Scaling	$f\left(\frac{t}{a}\right)$	$\lvert a \rvert F(a\omega)$	(A.4)
Convolution	$f_1 * f_2(t)$	$F_1(\omega) F_2(\omega)$	(A.5)
Multiplication	$f_1 f_2(t)$	$\frac{1}{2\pi} F_1 * F_2(\omega)$	(A.6)
Inversion	$F(t)$	$f(-\omega)$	(A.7)
Time differentiation	$\frac{\mathrm{d}f(t)}{\mathrm{d}t}$	$i\omega F(\omega)$	(A.8)

more needed? [21, p.25]

A.1.1. Important functions

Time signal $f(t)$	**Fourier transform** $\mathfrak{F}f(\omega) = F(\omega)$	
$\delta(t-\tau)$	$e^{-i\tau\omega}$	(A.9)

B Scripts

B.1. FT examples

Script B.1: Code for Example 4.5

```
close all;

%% DFT transforms
Fs = 1000;           % sampling frequency
t = 0:1/Fs:2-1/Fs;  % time distance

% choose signal
x = sin(100*pi*t)+sin(500*pi*t);

% use FFT algorithm to find the DFT
X=fftshift(fft(x));
X_norm=abs(X)/max(abs(X));
freq=linspace(-Fs/2,Fs/2,length(X));

% twosided spectrum
plot(freq, X_norm);
xlabel('frequency (Hz)');
ylabel('normalized energy density')

% onesided spectrum
plot(freq(end/2:end),X_norm(end/2:end));
xlabel('frequency (Hz)');
ylabel('normalized energy density')
```

Script B.2: Code for Example 4.6

```
%% FT transforms
Fs = 1000;           % sampling frequency
t = 0:1/Fs:2-1/Fs;  % time distance

% self composed signal
y1 = ones(size(t));
y2 = sin(10*pi*t);
y3 = 2*sin(50*pi*t);
y4 = 0.2*sin(80*pi*t);
nul = zeros(size(t));

x1=[y1(1:end/4) nul(end/4+1:end)];
x2=[y2(1:end/2) nul(end/2+1:end)];
x3=[nul(1:end/2) y3(end/2+1:end)];
x4=[nul(1:3*end/5) y4(3*end/5+1:4*end/5) nul(4*end/5+1:end)];

x=x1+x2+x3+x4;

X=fftshift(fft(x));
```

```
X_norm=abs(X)/max(abs(X));
freq=linspace(-Fs/2,Fs/2,length(X));

plot(freq(end/2:end),X_norm(end/2:end));
xlabel('frequency (Hz)');
ylabel('normalized energy density')
axis([0 100 0 1])
```

B.2. STFT examples

Script B.3: Code for Example 5.1

```
close all;

%% STFT transforms
Fs = 1000;           % sampling frequency
t = 0:1/Fs:2-1/Fs;   % time distance

% signal
x=sin(100*pi*t)+sin(500*pi*t);

% course
Ny = length(x);          % length of signal
nsc = floor(Ny/10);      % (number of sections) divides signal into 10 sections
nov = 0;                 % (number of overlap) no overlap between contiguous sections
nff = 30;                % max number of points for FFT

spectrogram(x,hann(nsc),nov,nff,Fs,'yaxis')
shading interp

% fine
Ny = length(x);          % length of signal
nsc = floor(Ny/20);      % (number of sections) divides signal into 20 sections
nov = floor(nsc/2);      % (number of overlap) 20% overlap between contiguous sections
nff = 200;                  % max number of points for FFT

spectrogram(x,hann(nsc),nov,nff,Fs,'yaxis')
shading interp
```

Script B.4: Code for Example 5.2

```
%% STFT transforms
Fs = 1000;           % sampling frequency
t = 0:1/Fs:2-1/Fs;   % time distance
close all;

% self composed signal of higher frequency
    y1 = ones(size(t));
    y2 = sin(100*pi*t);
    y3 = 2*sin(500*pi*t);
    y4 = 0.2*sin(800*pi*t);
    nul = zeros(size(t));

    x1=[y1(1:end/4) nul(end/4+1:end)];
    x2=[y2(1:end/2) nul(end/2+1:end)];
    x3=[nul(1:end/2) y3(end/2+1:end)];
    x4=[nul(1:3*end/5) y4(3*end/5+1:4*end/5) nul(4*end/5+1:end)];

    x=x1+x2+x3+x4;
% end self composed signal

% choose signal
y = chirp(t,100,1,200,'quadratic'); %or
y = x;

Ny = length(y);          % length of signal
nsc = floor(Ny/20);      % (number of sections) divides signal into sections of length
    ...
nov = floor(nsc/2);      % (number of overlap) 50% overlap between contiguous sections
nff=2000;                % max number of points for FFT
```

```
[spectro,freq,time]=spectrogram(y,hann(nsc),nov,nff,Fs,'yaxis');
subplot(2,1,1);
spectrogram(y,hann(nsc),nov,nff,Fs,'yaxis')
shading interp

subplot(2,1,2);
plot(t,y);
xlabel('Time (secs)')
ylabel('Amplitude')
```

B.3. Wavelets

https://github.com/scipy/scipy/blob/v0.15.1/scipy/signal/wavelets.py#L309 komt de code vandaan op 1/3/2017

Script B.5: Python code for CWT function

```
def cwt(data, wavelet, widths):
    """
    Continuous wavelet transform.

    Performs a continuous wavelet transform on `data`,
    using the `wavelet` function. A CWT performs a convolution
    with `data` using the `wavelet` function, which is characterized
    by a width parameter and length parameter.

    Parameters
    ----------
    data : (N,) ndarray
        data on which to perform the transform.
    wavelet : function
        Wavelet function, which should take 2 arguments.
        The first argument is the number of points that the returned vector
        will have (len(wavelet(width,length)) == length).
        The second is a width parameter, defining the size of the wavelet
        (e.g. standard deviation of a gaussian). See `ricker`, which
        satisfies these requirements.
    widths : (M,) sequence
        Widths to use for transform.

    Returns
    -------
    cwt: (M, N) ndarray
        Will have shape of (len(data), len(widths)).

    Notes
    -----
    >>> length = min(10 * width[ii], len(data))
    >>> cwt[ii,:] = scipy.signal.convolve(data, wavelet(length,
    ...                                     width[ii]), mode='same')

    Examples
    --------
    >>> from scipy import signal
    >>> import matplotlib.pyplot as plt
    >>> t = np.linspace(-1, 1, 200, endpoint=False)
    >>> sig  = np.cos(2 * np.pi * 7 * t) + signal.gausspulse(t - 0.4, fc=2)
    >>> widths = np.arange(1, 31)
    >>> cwtmatr = signal.cwt(sig, signal.ricker, widths)
    >>> plt.imshow(cwtmatr, extent=[-1, 1, 1, 31], cmap='PRGn', aspect='auto',
    ...            vmax=abs(cwtmatr).max(), vmin=-abs(cwtmatr).max())
    >>> plt.show()

    """
    output = np.zeros([len(widths), len(data)])
    for ind, width in enumerate(widths):
        wavelet_data = wavelet(min(10 * width, len(data)), width)
        output[ind, :] = convolve(data, wavelet_data,
                                  mode='same')
    return output
```

Script B.6: MATLAB code for Example 6.1.

```
Fs = 1000;            % sampling frequency
dt= 1/Fs;
t = 0:1/Fs:2-1/Fs;   % time distance

% choose signal
y = chirp(t,100,1,200,'quadratic');
y = sin(100*pi*t)+sin(500*pi*t);

% own signal
    y1 = ones(size(t));
    y2 = sin(100*pi*t);
    y3 = 2*sin(500*pi*t);
    y4 = 0.2*sin(800*pi*t);
    nul = zeros(size(t));

    x1 = [y1(1:end/4) nul(end/4+1:end)];
    x2 = [y2(1:end/2) nul(end/2+1:end)];
    x3 = [nul(1:end/2) y3(end/2+1:end)];
    x4 = [nul(1:3*end/5) y4(3*end/5+1:4*end/5) nul(4*end/5+1:end)];

    y = x1+x2+x3+x4;
% end own signal

a0 = 2^(1/64);
scales = 2*(a0.^(0:6*64));

[cfs,frequencies]=cwt(y, scales,'cmor1-1.5',dt);
surf(t,frequencies,abs(cfs));
```

Script B.7: Python code for Figure 6.12.

```
import pywt
from matplotlib import pyplot as plt

w= pywt.Wavelet('db2')
(phi,psi,x)=w.wavefun(level=5)
plt.subplot(3,2,1)
plt.plot(x,phi)
plt.title("Scaling function")

plt.subplot(3,2,2)
plt.plot(x,psi)
plt.title("Wavelet function")

plt.subplot(3,2,3)
plt.stem(w.dec_lo)
plt.axis([0, 3, -0.5, 1])
plt.title("Decomposition low pass filter")

plt.subplot(3,2,4)
plt.stem(w.dec_hi)
plt.axis([0, 3, -0.5, 1])
plt.title("Decomposition high pass filter")

plt.subplot(3,2,5)
plt.stem(w.rec_lo)
plt.axis([0, 3, -0.5, 1])
plt.title("Reconstruction low pass filter")

plt.subplot(3,2,6)
plt.stem(w.rec_hi)
plt.axis([0, 3, -0.5, 1])
plt.title("Reconstruction high pass filter")
```

B.4. Digital signal processing

Script B.8: Python code for Example 7.1 and 7.2.

```python
from pylab import *
import numpy as np
from math import *
import pywt
import Timfunctions as Tf
from statsmodels import robust
from matplotlib2tikz import save as tikz_save
import copy as copy

t = linspace(0, 10, 2056)
x = np.sin(2 * pi / 10 * t)
move = np.zeros_like(t)
move[:500] = -1 * np.ones([1, 500])
move[len(x)-500:] = np.ones([1, 500])

x += move
var_noise = var(x) / Tf.db2mag(10)
noise = np.random.normal(0, sqrt(var_noise), len(x))
xpn = x + noise

# FOURIER
XpN = fftshift(fft(xpn))
XpN_abs = abs(XpN) / max(abs(XpN))

filter_size = 40

rect = np.ones([1, filter_size])
hamm = np.hanning(2 * filter_size)

filter_rect = np.zeros_like(t)
filter_hann = np.array(filter_rect)
filter_rect[len(t)/2 - filter_size / 2:len(t)/2 + filter_size / 2] = rect
filter_hann[len(t)/2 - filter_size:len(t)/2 + filter_size] = hamm

XmR = XpN * filter_rect
XmH = XpN * filter_hann

xmr = ifft(ifftshift(XmR))
xmh = ifft(ifftshift(XmH))

# WAVELET
wavelet = "db4"
decomp = pywt.wavedec(xpn, wavelet)

# Universal threshold
T_U = sqrt(2 * np.log(len(xpn))) * robust.mad(decomp[-1]) / 0.6745

Thres_hard = copy.deepcopy(decomp)
Thres_soft = copy.deepcopy(decomp)
Thres_scale = copy.deepcopy(decomp)

for i in range(len(decomp))[1:] :
    Thres_hard[i][:] = pywt.threshold(Thres_hard[i][:], T_U, 'hard')
    Thres_soft[i][:] = pywt.threshold(Thres_soft[i][:], T_U / 2, 'soft')
    Thres_scale[i][:] = pywt.threshold(Thres_scale[i][:], 1.3 * std(Thres_scale[i][:]),
     'soft')

x_filt_hard = pywt.waverec(Thres_hard, wavelet)
x_filt_soft = pywt.waverec(Thres_soft, wavelet)
x_filt_scale = pywt.waverec(Thres_scale, wavelet)
```

Script B.9: Python code for Example 7.3.

```python
from math import *

import numpy as np
import pywt
from pylab import *
import copy as copy
import scipy.signal as signal
```

```
t = linspace(0, 10, 2056)
x = 0.5 * np.sin(2 * pi / 10 * t)

transient = 10**-4 * np.array([0, 7, -4, -3, 4, -3, 2, 1.5,-1, 0.6, -0.5, -0.4, 0.3,
    -0.2, 0.1, -0.05, 0.02, 0])

x[1222:1222+len(transient)] += transient

# FOURIER
X = fftshift(fft(x))
X_abs = abs(X) / max(abs(X))

# WAVELET
wavelet = "db4"
decomp = pywt.wavedec(x, wavelet, level=1, mode = 'smooth')

recomp = copy.deepcopy(decomp)
recomp[0] = zeros_like(decomp[0])
recomposition = pywt.waverec(recomp, wavelet, mode='smooth')
```

C
Signal examples

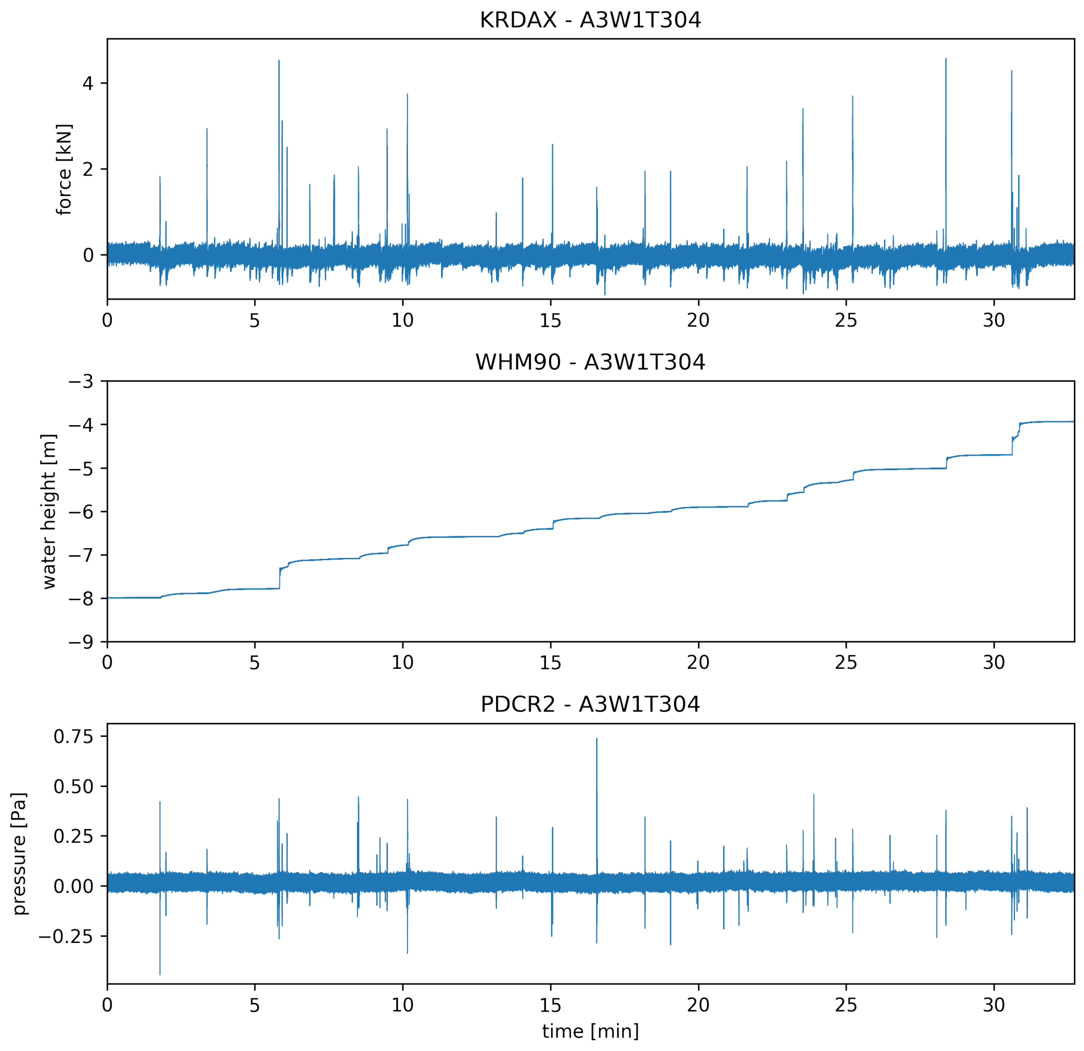


Figure C.1: Example time signals from Deltares: KRDAX is a force measurement, WHM90 is a water height measurement and PDCR2 is a pressure measurement. A more elaborate description of the signals can be found in Section 2.1.

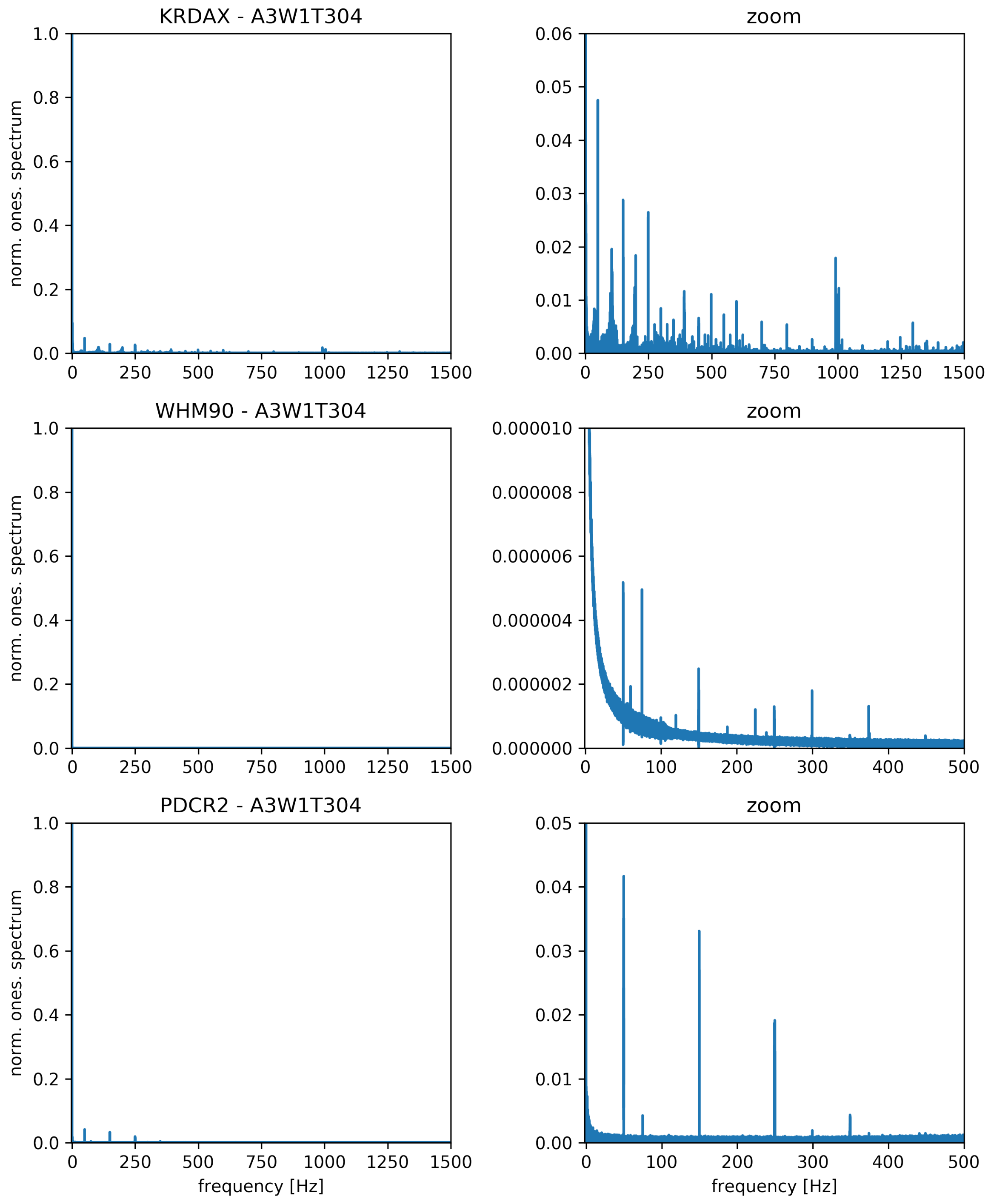


Figure C.2: Spectra of time signal examples from Figure C.1. On the right hand side one or both of the axes are zoomed to have more insight in the spectrum

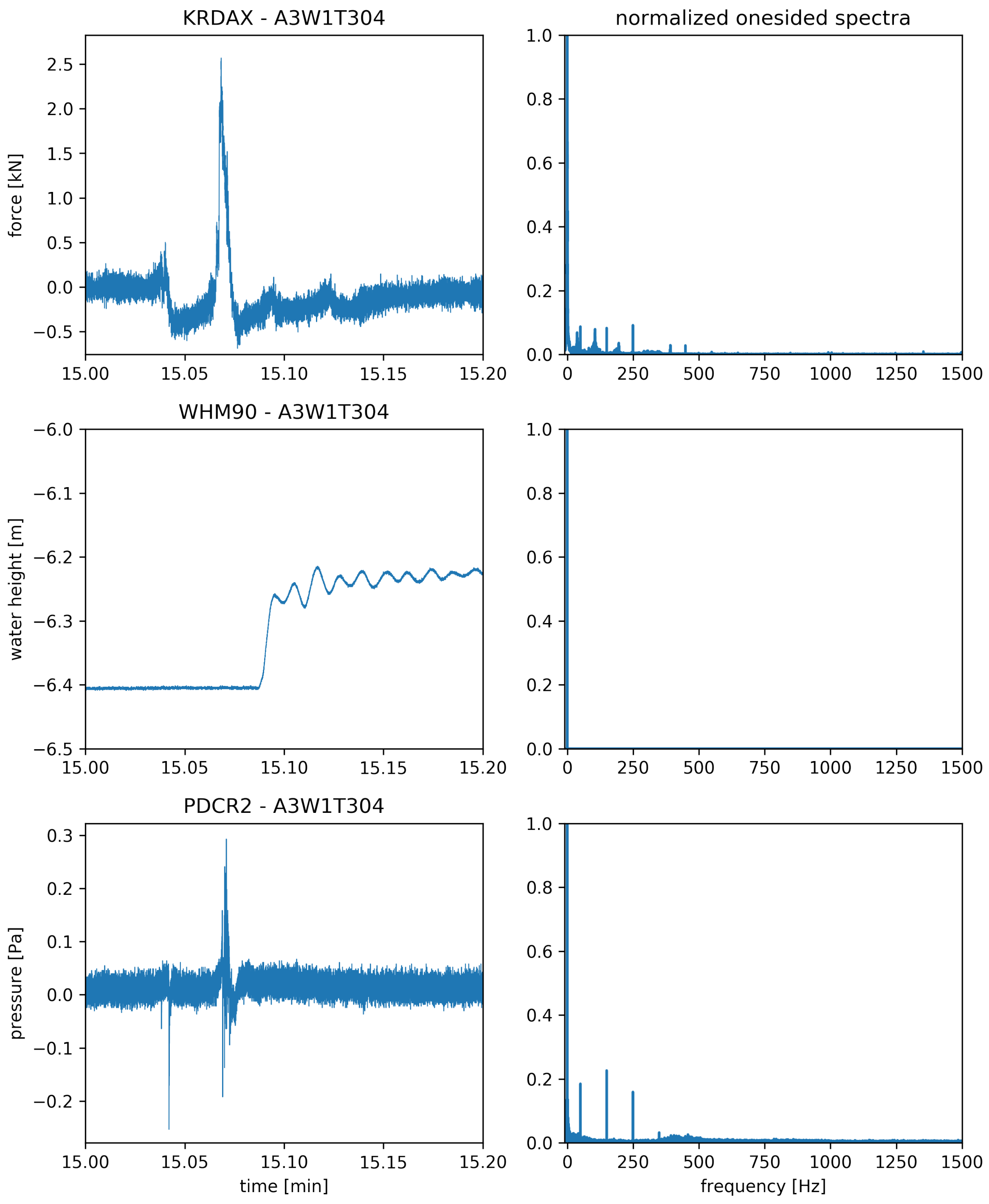


Figure C.3: The time signals from Figure C.1 between 15 min and 15 min and 12 sec (5.2 minute): the spectra show a relatively larger contribution of the frequencies >30 Hz than in Figure C.2.

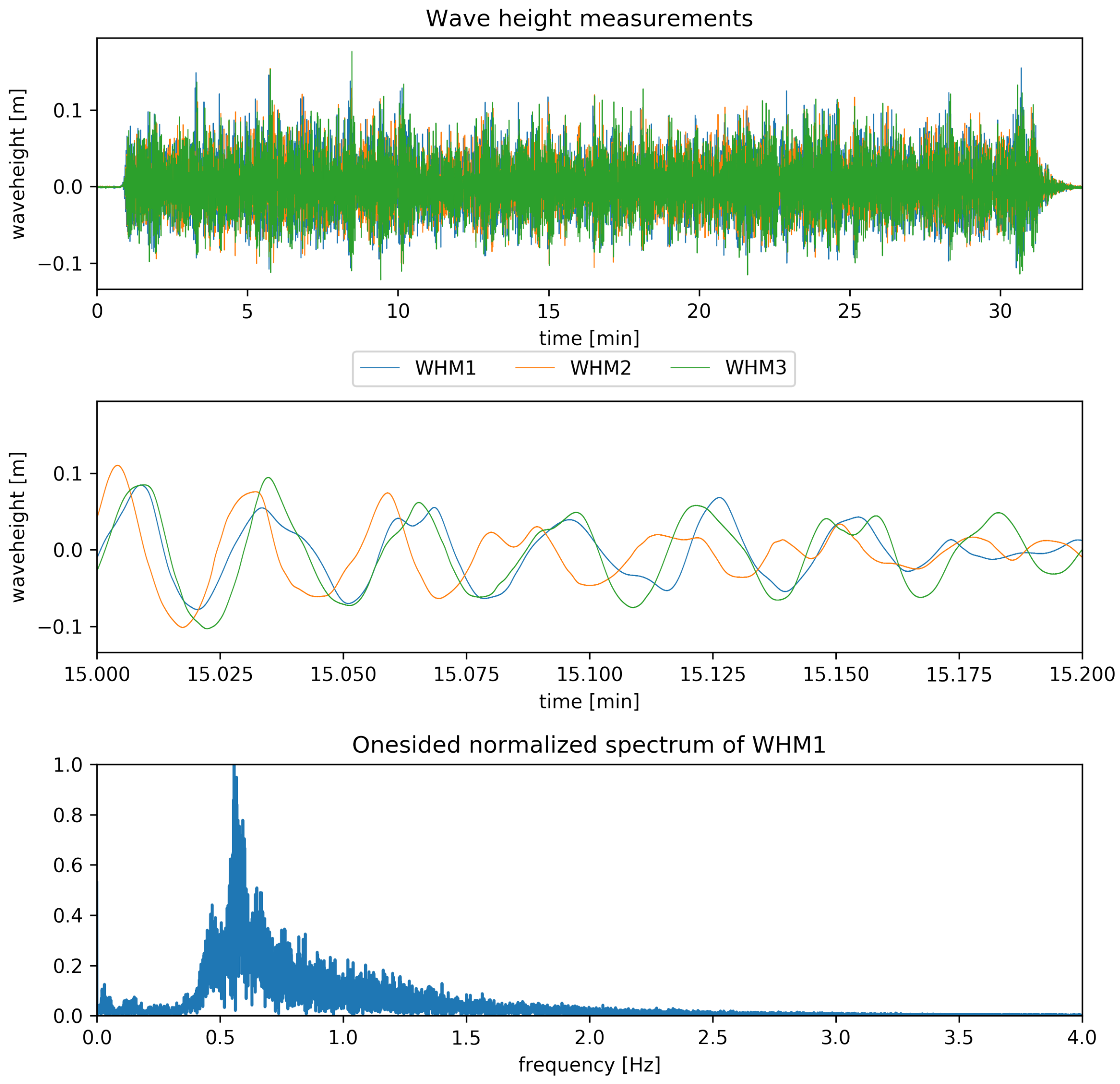


Figure C.4: The wave heights in the flume.

D MSc assignment

MSc-thesis assignment

Wavelets

Figure 1 Example of obtaining ware field information near trunk of a breakwater

The FBI uses wavelet techniques for image compression of digital fingerprints to save storage space. In geophysics wavelets are being used to analyse seismic signals for detecting e.g. earthquakes and oil layers. In finance the wavelet is used to analyse stock markets due to their dynamic and non-linear nature. These are just a few examples to highlight the applicability of the wavelet techniques.

The same techniques are potentially very interesting for Deltares as coastal engineers have to deal with complex time dependent physical processes as for example illustrated in the figures included.

To improve the understanding of these physical processes associated with waves, wave structure interaction, stability of structures or the influence on morphology various measurements techniques like time sampling, lasers scanning, photography are employed to capture instant information on wave conditions, forces, currents, erosion and accretion for further detailed analysis.

Figure 2 Example of measuring wave field

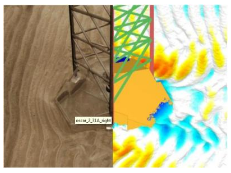


Figure 3 Example Stereophotography

This assignment is related to the part of detailed analysis of time series containing the time evolution of wave heights, forces, etc. Currently, the analysis is performed through Fourier analysis combined with filtering techniques to remove e.g. noise. However, the Fourier analysis has its limitations.

The purpose of this assignment is to look into the added value of applying existing wavelets and the related techniques compared to Fourier type of analysis.

The question is: can we improve our analysis by employing wavelet instead of Fourier technique? To answer this question, the following tasks have been defined for this assignment.

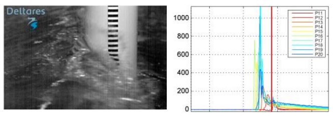
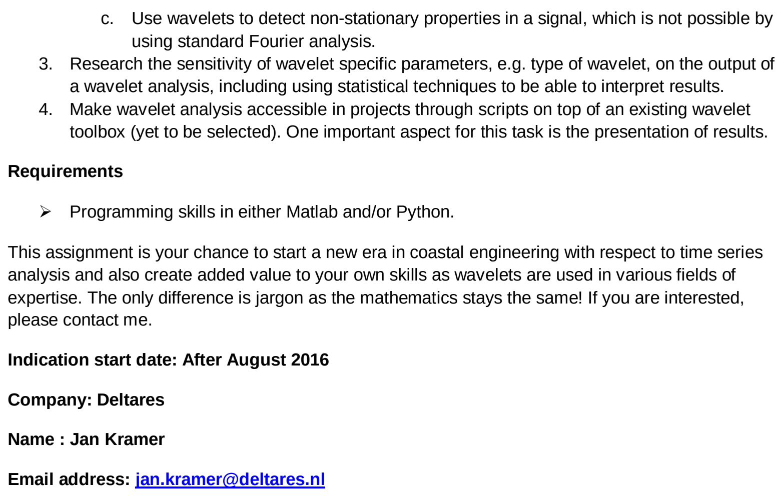


Figure 4 Example wave impact on a pile and measured pressure

Tasks

1. Provide a summary/overview on how wavelets are being used in other fields of expertise, including background information on the mathematical aspects of wavelets.
2. Verify the added value of wavelets by comparing results obtained through wavelet techniques with Fourier analysis by using different type of measured of time series (e.g. pressure due to wave impact, wave height, etc.) which representative for Deltares. Important aspects related to this task are:
    a. Use wavelets to detect and filter different components (e.g. noise) from the signal by using for example thresholding methods.
    b. Use wavelets to detect, if possible, the influence of wave basin characteristics on measured signals.



   c. Use wavelets to detect non-stationary properties in a signal, which is not possible by using standard Fourier analysis.
3. Research the sensitivity of wavelet specific parameters, e.g. type of wavelet, on the output of a wavelet analysis, including using statistical techniques to be able to interpret results.
4. Make wavelet analysis accessible in projects through scripts on top of an existing wavelet toolbox (yet to be selected). One important aspect for this task is the presentation of results.

Requirements

- Programming skills in either Matlab and/or Python.

This assignment is your chance to start a new era in coastal engineering with respect to time series analysis and also create added value to your own skills as wavelets are used in various fields of expertise. The only difference is jargon as the mathematics stays the same! If you are interested, please contact me.

Indication start date: After August 2016

Company: Deltares

Name : Jan Kramer

Email address: jan.kramer@deltares.nl

The MSc proposal as subjected by Deltares via `http://ta.twi.tudelft.nl/nw/users/vuik/numanal/rooij.html`.