# Wavelets and the Lifting Scheme

la Cour-Harbo, Anders; Jensen, Arne

*Publication date:*
2007

*Document Version*
Publisher's PDF, also known as Version of record

# AALBORG UNIVERSITY

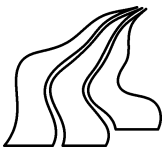
## Wavelets and the Lifting Scheme

by

Anders la Cour-Harbo and Arne Jensen

R-2007-27 October 2007

Department of Mathematical Sciences
Aalborg University
Fredrik Bajers Vej 7 G ▪ DK - 9220 Aalborg Øst ▪ Denmark
Phone: +45 96 35 80 80 ▪ Telefax: +45 98 15 81 29
URL: http://www.math.aau.dk



ISSN 1399–2503 ▪ On-line version ISSN 1601–7811

# Wavelets and the Lifting Scheme

Anders la Cour-Harbo[a], Arne Jensen[b]

[a] Aalborg University, Department of Electronics Systems, Denmark.
[b] Aalborg University, Department of Mathematical Sciences, Denmark

# Article Outline

Glossary
I. Definition
II. Introduction
III. Lifting
IV. Prediction and Update
V. Interpretation
VI. Lifting and Filter Banks
VII. Making Lifting Steps From Filters

# Glossary

**Lifting Step**

An equation describing the transformation of an odd (even) sample of a signal by means of a linear combination of even (odd) samples, respectively. Changing an odd sample is sometimes called a prediction step, while changing an even sample is called an update step.

**Discrete Wavelet Transform (DWT)**

Transformation of a discrete (sampled) signal into another discrete signal by means of a wavelet basis. The transform can be accomplished in a number of ways, typically either by a two channel filter bank or by lifting steps.

**Filter Bank**

A series of bandpass filters, which separates the input signal into a number of components, each with a distinct range of frequencies of the original signal.

**z-transform**

A mapping of a vector $\mathbf{x} = \{x[k]\}$ to a function in the complex plane by

$$X(z) = \sum_k x[k] z^{-k} \, .$$

For a vector with a finite number of non-zero entries $X(z)$ is a Laurent polynomial.

**Filter**

A linear map, which maps a signal $\mathbf{x}$ with finite energy to another signal with finite energy. In the time domain it is given by convolution with a vector $\mathbf{h}$, which in the frequency domain is equal to $H(z)X(z)$. The vector $\mathbf{h}$ is called the impulse response (IR) of the filter (or sometimes the filter taps), and $H(e^{j\omega})$ the transfer function (or sometimes the frequency response). If $\mathbf{h}$ is a finite sequence, then $\mathbf{h}$ is called a FIR (finite impulse response) filter. An infinite $\mathbf{h}$ is then called an IIR filter. We only consider FIR filters in this article. We restrict ourselves to filters with real coefficients.

**Filter taps**

The entries in the vector that by convolution with a signal gives a filtering of the signal. The filter taps are also called the impulse response of the filter.

**Laurent Polynomial**

A polynomial in the variables $z$ and $z^{-1}$. Some examples: $3z^{-2} + 2z$, $z^{-3} - 2z^{-1}$, and $1 + z + z^3$. The $z$-transform of a FIR filter $\mathbf{h}$ is a Laurent polynomial of the form

$$H(z) = \sum_{k=k_\mathrm{b}}^{k_\mathrm{e}} h[k] z^{-k}, \qquad k_\mathrm{b} \leq k_\mathrm{e} \; .$$

This is in contrast to ordinary polynomials, where we only have non-negative powers of $z$. Assuming that $h_{k_\mathrm{e}} \neq 0$ and $h_{k_\mathrm{b}} \neq 0$, the degree of a Laurent polynomial is defined as $|h| = k_\mathrm{e} - k_\mathrm{b}$.

**Monomial**

A Laurent polynomial of degree $0$, for example $3z^7$, $z^{-8}$, or $12$.

# I Definition

The objective of this article is to give a concise introduction to the discrete wavelet transform (DWT) based on a technique called *lifting*. The lifting technique allows one to give an elementary, but rigorous, definition of the DWT, with modest requirements on the reader. A basic knowledge of linear algebra and signal processing will suffice. The lifting based definition is equivalent to the usual filer bank based definition of the DWT. The article does not discuss applications in any detail. The reader is referred to other articles in this collection.

The DWT was introduced in the second half of the eighties, through the work of Y. Meyer, I. Daubechies, S. Mallat, and many other researchers. The lifting technique was introduced by

W. Sweldens in 1996, and the equivalence with the filter bank approach was established by I. Daubechies and W. Sweldens in an article published in 1998 [2], but available in preprint form from 1996.

This article is based on the book Ripples in Mathematics – The Discrete Wavelet Transform [3] with kind permission of Springer Science and Business Media. For more information on the background and history of wavelets, we again refer to other articles in this collection.

# II Introduction

We begin this exposition of the lifting technique with a simple, yet very descriptive example. It will shortly become apparent how this is related to lifting and to the discrete wavelet transform. We take a digital signal consisting of eight samples (this idea is due to Mulcahy [5], and we use his example, with a minor modification)

$$56, \quad 40, \quad 8, \quad 24, \quad 48, \quad 48, \quad 40, \quad 16\,.$$

We choose to believe that these numbers are not random, but have some structure that we want to extract. Perhaps there is some correlation between a number and its immediate successor. To reveal such a correlation we will take the numbers in pairs and compute the mean, and the difference between the first member of the pair and the computed mean. The result of this computation is shown as the second row in Table 1. This row contains the four means followed by the four differences, the latter being typeset in boldface. We then leave the four differences unchanged and apply the mean and difference computations to the first four entries of the second row. We repeat this procedure once more on the third row. The fourth row then contains a first entry, which is the mean of the original eight numbers, and the seven calculated differences. The boldface entries in the table are called the details of the signal.

Table 1: Mean and difference computation. Differences are in boldface type

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 56 | 40 | 8 | 24 | 48 | 48 | 40 | 16 |
| 48 | 16 | 48 | 28 | **8** | **−8** | **0** | **12** |
| 32 | 38 | **16** | **10** | **8** | **−8** | **0** | **12** |
| 35 | **−3** | **16** | **10** | **8** | **−8** | **0** | **12** |

It is important to observe that no information has been lost in this transformation of the first row into the fourth row. We can see this by reversing the calculation. Beginning with the last row, we compute the first two entries in the third row as $32 = 35 + (-3)$ and $38 = 35 - (-3)$. In the same way the first four entries in the second row are calculated as $48 = 32 + (16)$, $16 = 32 - (16)$, $48 = 38 + (10)$, and finally $28 = 38 - (10)$. Repeating this procedure we get the first row in the table.

So, what is the purpose of these computations? The four signals contain the same information, just in different ways, so how do we gain anything from this change of representation of the signal? If our assumption (pairwise equality of samples) is correct, the four means in the second row will equal the original numbers, and the four differences will be zero. If further, the original numbers are equal in groups of four, the two means in the third row will equal the original quadruples, and the differences will be zero. Finally, if all eight original samples are equal, the mean in row four will be equal to all eight samples, and the seven differences will be zero.

Apparently, our assumption is not correct, as the differences are not zero. However, the numbers in the fourth row are generally smaller than the original numbers. This indicates that while the numbers are not equal, they are 'similar', leaving us with small differences. We can say that we have achieved some kind of loss-free compression by reducing the dynamic range of the signal. By loss-free we mean that we can transform back to the original signal without any information being lost in the process. As a simple measure of compression we can count the number of digits used to represent the signal. The first row contains 15 digits. The last row contains 12 digits and two negative signs. So in this example the compression is not very large. But it is easy to give other examples, where the compression can be substantial.

We see in this example that the pair 48, 48 do fit our assumption of equality, and therefore the difference is zero. Suppose that after transformation we find that many 'difference entries' are zero. Then we can store the transformed signal more efficiently by only storing the non-zero entries (and their locations).

## II.1 Processing the Transformed Signal

Let us now suppose that we are willing to accept a certain loss of quality in the signal, if we can get a higher rate of compression. One technique for lossy compression is called thresholding. We choose a threshold and decide to set equal to zero all entries with an absolute value less than this threshold value. Let us in our example choose $4$ as the threshold. This means that we in Table 1 replace the entry $\mathbf{-3}$ by $\mathbf{0}$ and then perform the reconstruction. The result is in Table 2, left hand part.

The original and transformed signals are both shown on the left in Figure 1. We have chosen to join the sample points by straight line segments to get a good visualization of the signals. Clearly the two graphs differ very little. Now let us perform a more drastic compression. This time we

Table 2: Reconstruction with threshold $4$ (left) and threshold $9$ (right).

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 59 | 43 | 11 | 27 | 45 | 45 | 37 | 13 |
| 51 | 19 | 45 | 25 | **8** | **−8** | **0** | **12** |
| 35 | 35 | **16** | **10** | **8** | **−8** | **0** | **12** |
| 35 | **0** | **16** | **10** | **8** | **−8** | **0** | **12** |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 51 | 51 | 19 | 19 | 45 | 45 | 37 | 13 |
| 51 | 19 | 45 | 25 | **0** | **0** | **0** | **12** |
| 35 | 35 | **16** | **10** | **0** | **0** | **0** | **12** |
| 35 | **0** | **16** | **10** | **0** | **0** | **0** | **12** |

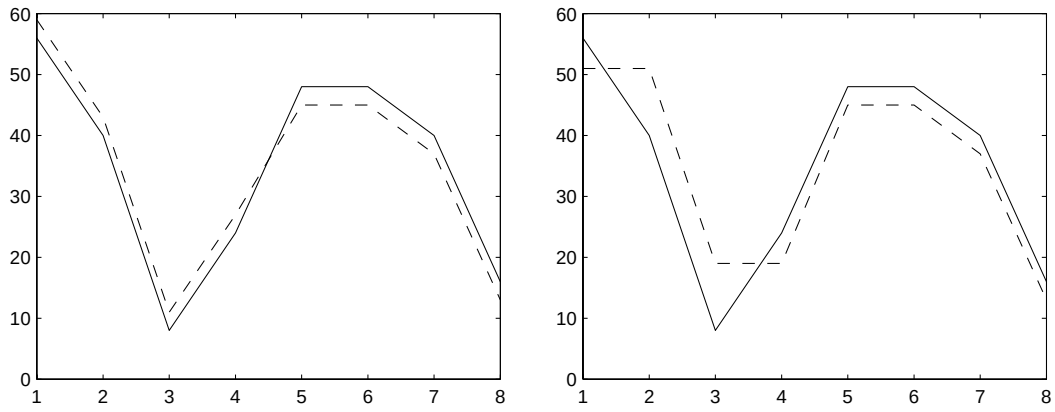



Figure 1: Original signal (solid) and modified signal (dashed) with threshold $4$ (left) and $9$ (right).

choose the threshold equal to $9$. The computations are given on the right in Table 2, and the graphs are plotted on the right in Figure 1. Notice that the peaks in the original signal have been flattened. We also note that the signal now is represented by only four non-zero entries.

There are several variations of the transformation. We could have stored differences instead of 'half-differences', or we could have used the difference between the second element of the pair and the computed average. The first choice will lead to boldface entries in the tables that can be obtained from the computed ones by multiplication by a factor $-2$. The second variant is obtained by multiplication by $-1$.

The 'mean-difference' transformation can be performed on any signal of even length, since all we need is the possibility of taking pairs. If we want to be able to keep transforming the signal until there is one signal mean left, we need a signal of length $2^N$, and it will lead to a table with $N + 1$ rows, where the first row is the original signal. If the given signal has a length different from a power of $2$, then we will have to do some additional operations on the signal to compensate for that. One possibility is to add samples with value zero to one or both ends of the signal until a length of $2^N$ is achieved. This is called zero padding.

# III Lifting

The transformation performed by successively calculating means and differences of a signal is an example of the discrete wavelet transform. Specifically, this transform introduced in the example is called the Haar transform, and occasionally also the first of the Daubechies transforms. It can be undone by simply reversing the steps performed, and it provides a number of different representations of the same signal. Actually, all discrete wavelet transforms can be realized by a similar procedure, where the only modification from the Haar transform is the way, in which we compute the means part and differences part of the transformed signal. The 'procedure' is known as

'lifting' in the literature. This procedure was introduced by Wim Sweldens in 1996 in a series of papers [8, 9].

To fully appreciate the concept of lifting we need to elaborate on the example in the previous section. We have a discrete signal of real (or complex) numbers

$$x[0], x[1], x[2], x[3], \ldots, x[N-1] \; .$$

Our convention is to start indexing at zero. In implementations one may need to adapt this to the programming language used. Here we assume the signal to have finite length $N$, but this is not a restriction. The theory also applies to infinite signals, provided they have finite energy. In the mathematical (and signal processing) literature finite energy means

$$\sum_{n=-\infty}^{\infty} \big|x[n]\big|^2 < \infty \; ,$$

and the set of such signals is usually denoted by $\ell^2(\mathbf{Z})$. Sometimes we use the mathematical term sequence instead of signal, and we also use the term vector, in particular in connection with use of results from linear algebra.

## III.1 Introducing Prediction and Update

Let us now return to the example in the Introduction. We took a pair of numbers $a, b$ and computed the mean, and the difference between the first entry and the mean

$$s = \frac{a+b}{2} \; , \tag{1}$$

$$d = a - s \; . \tag{2}$$

Alternatively, almost the same result can be achieved in the following way

$$d = a - b \; , \tag{3}$$

$$s = \frac{a+b}{2} = \frac{d}{2} + b \; , \tag{4}$$

except we now have the difference rather than half the difference. This mathematical formulation of the Haar transform, albeit quite simple, brings us to the definition of lifting. The two operations, mean and difference, can be viewed as special cases of more general operations. Remember that we assumed that there is some correlation between two successive samples, and we therefore computed the difference. If two samples are almost equal, the difference is, of course, small. Thus one can use the first sample to predict the second sample, the prediction being that they are equal. If is is a good prediction, the difference between them is small. Thus we can call (3) a *prediction step*. We can use other and more sophisticated prediction steps than one based on just the previous sample. We will give a few examples of this later.

We also calculated the mean of the two samples. This step can be viewed in two ways. Either as an operation, which preserves some properties of the original signal (later we shall see how the mean value or the energy of a signal is preserved during transformation), or as an extraction of essential features of the signal. The latter viewpoint is based on the fact that the pair-wise mean values contain the overall structure of the signal, but with only half the number of samples. We use the term *update step* for the this operation, and (4) is the update step in the Haar transform. Just as the prediction operation, the update operation can be more sophisticated than just calculating the mean.

The prediction and update operations are shown in Figure 2. The notation and the setup is a little different from the Introduction; we start with a finite sequence $\mathbf{s}_j$ of length $2^j$ and end with two sequences $\mathbf{s}_{j-1}$ and $\mathbf{d}_{j-1}$, each of length $2^{j-1}$. Let us explain the steps in Figure 2.

**split** The entries are sorted into the even and the odd entries. It is important to note that we do this only to explain the algorithm. In (effective) implementations the entries are not moved or separated.

**prediction** If the signal contains some structure, then we can expect correlation between a sample and its nearest neighbors. In our first example the prediction is that the signal is constant. More elaborately, given the value at the sample number $2n$, we predict that the value at sample $2n+1$ is the same. We then compute the difference and in the figure let it replace the value at $2n+1$, since this value is not needed anymore. In our notation this is

$$d_{j-1}[n] = s_j[2n+1] - s_j[2n] \ .$$

In general, the idea is to have a prediction procedure $P$ and then compute the difference signal as

$$\mathbf{d}_{j-1} = \text{odd}_{j-1} - P(\text{even}_{j-1}) \ . \tag{5}$$

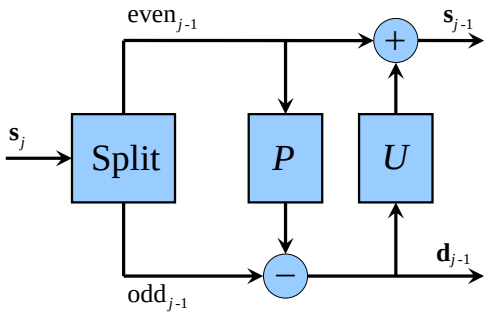



Figure 2: The three steps in a lifting building block. Note that the minus sign means 'the signal from the left minus the signal from the top'

Thus in the $\mathbf{d}$ signal each entry is one odd sample minus some prediction based on a number of even samples.

**update** Given an even entry we predicted that the next odd entry has the same value, and stored the difference. We then update our even entry to reflect our knowledge of the signal. In the example above we replaced the even entry by the average. In our notation

$$s_{j-1}[n] = s_j[2n] + d_{j-1}[n]/2 \; .$$

In general we decide on an updating procedure, and then compute

$$\mathbf{s}_{j-1} = \text{even}_{j-1} + U(\mathbf{d}_{j-1}) \; . \tag{6}$$

The algorithm described here is called one step lifting. It requires the choice of a prediction procedure $P$, and an update procedure $U$.

### III.2 Multiple Transforms

The discrete wavelet transform is obtained by combining a number of lifting steps. As in the example in Table 1 we keep the computed differences $\mathbf{d}_{j-1}$ and use the average sequence $\mathbf{s}_{j-1}$ as input for one more lifting step. This two step procedure is illustrated in Figure 3. Obviously, we can apply more lifting steps if we want to transform the signal further. In general, we start with a signal $\mathbf{s}_j$ of length $2^j$ and we can repeat the transformation $j$ times until we have a single number $s_0[0]$ and $j$ difference sequences $\mathbf{d}_{j-1}$ to $\mathbf{d}_0$. If we let $j = 3$ and use the Haar transform $s_0[0]$ will be be the mean value of the eight entries in the original sequence, and $\mathbf{d}_0$, $\mathbf{d}_1$, and $\mathbf{d}_2$ will be the numbers in boldface in Table 1. In lifting step notation this table becomes as Table 3.

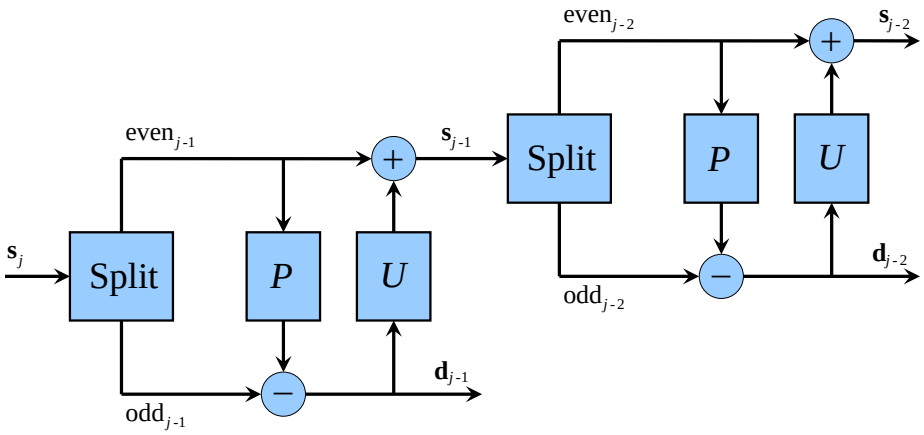



Figure 3: Two step discrete wavelet transform

Table 3: This is Table 1 in the lifting step notation.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $s_3[0]$ | $s_3[1]$ | $s_3[2]$ | $s_3[3]$ | $s_3[4]$ | $s_3[5]$ | $s_3[6]$ | $s_3[7]$ |
| $s_2[0]$ | $s_2[1]$ | $s_2[2]$ | $s_2[3]$ | $d_2[0]$ | $d_2[1]$ | $d_2[2]$ | $d_2[3]$ |
| $s_1[0]$ | $s_1[1]$ | $d_1[0]$ | $d_1[1]$ | $d_2[0]$ | $d_2[1]$ | $d_2[2]$ | $d_2[3]$ |
| $s_0[0]$ | $d_0[0]$ | $d_1[0]$ | $d_1[1]$ | $d_2[0]$ | $d_2[1]$ | $d_2[2]$ | $d_2[3]$ |

We have previously motivated the prediction operation with the reduction in dynamic range of the signal obtained in using differences rather than the original values, potentially leading to good compression of a signal. The update procedure has not yet been clearly motivated. The update performed in the first example was

$$s_{j-1}[n] = s_j[2n] + \tfrac{1}{2} d_{j-1}[n] = \tfrac{1}{2}(s_j[2n] + s_j[2n+1]) \; .$$

It turns out that this update operation preserves the mean value. The consequence is that all the $\mathbf{s}$ sequences have the same mean value. It is easy to verify in the case of the example in Table 1, since

$$\frac{56+40+8+24+48+48+40+16}{8} = \frac{48+16+48+28}{4} = \frac{32+38}{2} = 35 \; .$$

It is not difficult to see that this hold for any sequence $\mathbf{s}$ of length $2^j$. In particular, $s_0[0]$ equals the mean value of the original samples $s_j[0], \ldots, s_j[2^j - 1]$.

### III.3 Inverse Transform

A lifting step is easy to undo. In fact, looking at Figure 2 it is clear that we can go backwards from $\mathbf{s}_{j-1}$ and $\mathbf{d}_{j-1}$ to $\mathbf{s}_j$ simply by first 'undoing' the update step and then 'undoing' the prediction step. This is accomplished by using the same update step as in the direct lifting step, but subtracting instead of adding, followed by the same prediction step, but using addition instead of subtraction. Finally, we need to merge the odd and even samples. The direct lifting step and the corresponding inverse lifting step are shown in Figure 4. This merging step, where the sequences $\text{even}_{j-1}$ and $\text{odd}_{j-1}$ are merged to form the sequence $\mathbf{s}_j$, is shown to explain the algorithm. It is not performed in implementations, since the entries are not reordered in the first place. Mathematically, the inverse lifting step is accomplished by using the formulas in opposite order, and isolating the odd and even samples. In our example the inverse lifting step becomes

$$b = s - d/2 \; , \tag{7}$$

$$a = d + b \; , \tag{8}$$

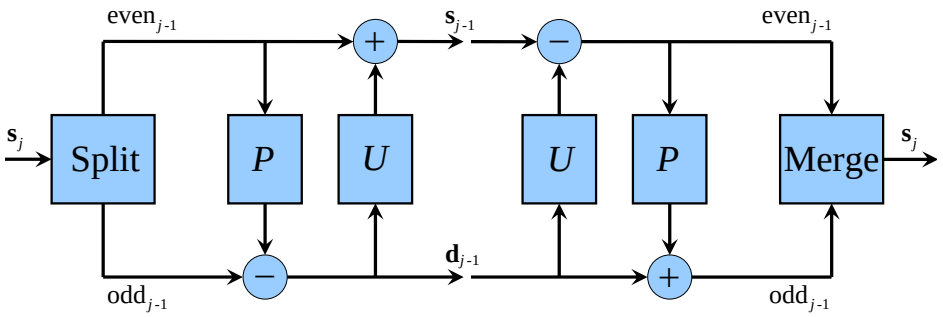



Figure 4: Direct and inverse lifting step.

where (7) is actually (2) with $b$ isolated, and (8) is (1) with $a$ isolated. In the lifting step notation this is first (6) rewritten as

$$s_j[2n] = s_{j-1}[n] - d_{j-1}[n]/2$$

to undo the update step, and then (5) rewritten as

$$s_j[2n+1] = d_{j-1}[n] + s_j[2n]$$

to undo the prediction step. For the general lifting step

$$\begin{aligned} \mathbf{d}_{j-1} &= \text{odd}_{j-1} - P(\text{even}_{j-1}) \\ \mathbf{s}_{j-1} &= \text{even}_{j-1} + U(\mathbf{d}_{j-1}) \end{aligned}$$

the inversion is accomplished by the steps

$$\begin{aligned} \text{even}_{j-1} &= \mathbf{s}_{j-1} - U(\mathbf{d}_{j-1}) \\ \text{odd}_{j-1} &= \mathbf{d}_{j-1} + P(\text{even}_{j-1}) \ . \end{aligned}$$

That is all there is to the inverse transform.

# IV Prediction and Update

Assuming that a signal is constant was useful in the first example to demonstrate the concept of lifting. However, this assumption is hardly the best for most signals. Fortunately, there are many other possible prediction and update procedures to accommodate various signal assumptions. We will present two examples of other lifting steps. The first example is the next logical step from the Haar transform when we assume the signal to be stepwise linear rather than just constant. The second example is not based on a specific assumption about the signal, but is taken from the classical wavelet theory and adapted to the lifting step method. We will show the lifting steps for the Daubechies 4 wavelet.

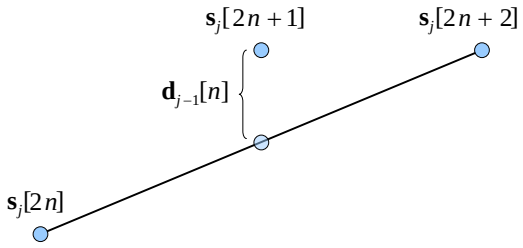



Figure 5: The linear prediction uses two even samples to predict one odd sample.

## IV.1 Linear Prediction

Instead of basing the prediction on the assumption that the signal is constant, we will now base it on the assumption that the signal is linear. We really do mean an affine signal, but we stick to the commonly used term 'linear.' Thus a linear signal is in this context a signal of the form $s_j[n] = \alpha n + \beta$, that is, all the samples of the signal lie on a straight line. For a given odd entry $s_j[2n+1]$ we now need the two nearest even neighbors to predict the value (remember that for the constant signal we needed only one neighbor). The predicted value is $\frac{1}{2}(s_j[2n] + s_j[2n+2])$, the mean of the two even samples. This value is the circle on the middle of the line in Figure 5. The correction is then the difference between what we predict the middle sample to be and what it actually is

$$d_{j-1}[n] = s_j[2n+1] - \tfrac{1}{2}(s_j[2n] + s_j[2n+2]) \; .$$

We decide to base the update procedure on the two most recently computed differences. We take it of the form

$$s_{j-1}[n] = s_j[2n] + A(d_{j-1}[n-1] + d_{j-1}[n]) \; ,$$

where $A$ is a constant to be determined. In the first example we had the property

$$\sum_n s_{j-1}[n] = \tfrac{1}{2} \sum_n s_j[n] \; , \tag{9}$$

that is, the mean was preserved. (Recall that $\mathbf{s}_j$ has length $2^j$ and $\mathbf{s}_{j-1}$ has length $2^{j-1}$. This explains the factor $\frac{1}{2}$.) We would like to have the same property here. Let us first rewrite the expression for $s_{j-1}[n]$ above,

$$\begin{aligned} s_{j-1}[n] &= s_j[2n] + Ad_{j-1}[n-1] + Ad_{j-1}[n] \\ &= s_j[2n] + A(s_j[2n-1] - \tfrac{1}{2}s_j[2n-2] - \tfrac{1}{2}s_j[2n]) \\ &\qquad + A(s_j[2n+1] - \tfrac{1}{2}s_j[2n] - \tfrac{1}{2}s_j[2n+2]) \; . \end{aligned}$$

Using this expression, and gathering even and odd terms, we get

$$\sum_n s_{j-1}[n] = (1-2A) \sum_n s_j[2n] + 2A \sum_n s_j[2n+1] \; .$$

To satisfy (9) we must choose $A = \frac{1}{4}$. Summarizing, we have the following two steps

$$d_{j-1}[n] = s_j[2n+1] - \tfrac{1}{2}(s_j[2n] + s_j[2n+2]) \ , \tag{10}$$

$$s_{j-1}[n] = s_j[2n] + \tfrac{1}{4}(d_{j-1}[n-1] + d_{j-1}[n]) \ . \tag{11}$$

The transform in this example also has the property.

$$\sum_n n s_{j-1}[n] = \tfrac{1}{4} \sum_n n s_j[n] \tag{12}$$

Note that there is a misprint in this formula in [3, formula (3.15)]. We say that the transform preserves the *first moment* of the sequence. The mean is also called the *zeroth moment* of the sequence. Finally, we want to note that this transform is known in the literature as CDF(2,2). The origin of this name is explained later.

In the above presentation we have simplified the notation by not specifying where the finite sequences start and end, thereby for the moment avoiding keeping track of the ranges of the variables. In other words, we have considered our finite sequences as infinite, adding zeroes before and after the given entries. In implementations one has to keep track of these things, but doing so now would obscure the simplicity of the lifting procedure. For more details on transformation of finite signals, see Chapter 10 in Jensen and la Cour-Harbo [3].

## IV.2 Daubechies 4 Wavelet

We now turn to the second example, the Daubechies 4 wavelet. This transform is a bit different from what we have seen so far. So to adapt this transform to the lifting step method we need to first take a closer look at the construction of the lifting step. Looking at Figure 4 once more we see that we can add another prediction step after the update step without changing the lifting concept; we still have even and odd samples, we still have $\mathbf{s}_{j-1}$ and $\mathbf{d}_{j-1}$ as output, and we can still invert the lifting step by reversing the order of the prediction and update steps. In fact, we can add as many prediction and update steps as we want, and still have the same lifting concept. As an illustration of this Figure 6 shows a direct transform consisting of three pairs of prediction and update operations. If we have two prediction and only one update, but insist on having them in pairs (this is occasionally useful in the theory), we can always add an operation of either type, which does nothing.

It turns out that this generalization is crucial in applications. There are many important transforms, where the steps do not occur in pairs. The Daubechies 4 transform is an example, where there is a $U$ operation followed by a $P$ operation and another $U$ operation. Furthermore, in the last two steps, in (16) and (17), we add a new type of operation which is called normalization, or

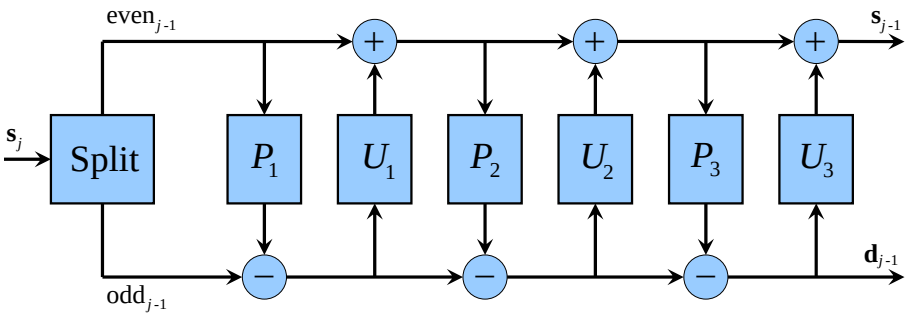



Figure 6: Multiple prediction and update steps.

sometimes rescaling. The resulting algorithm is applied to a signal $\{s_j[n]\}_{n\in\mathbf{Z}}$ as follows

$$s^{(1)}_{j-1}[n] = s_j[2n] + \sqrt{3}s_j[2n+1] \tag{13}$$

$$d^{(1)}_{j-1}[n] = s_j[2n+1] - \tfrac{1}{4}\sqrt{3}s^{(1)}_{j-1}[n] - \tfrac{1}{4}(\sqrt{3}-2)s^{(1)}_{j-1}[n-1] \tag{14}$$

$$s^{(2)}_{j-1}[n] = s^{(1)}_{j-1}[n] - d^{(1)}_{j-1}[n+1] \tag{15}$$

$$s_{j-1}[n] = \frac{\sqrt{3}-1}{\sqrt{2}}s^{(2)}_{j-1}[n] \tag{16}$$

$$d_{j-1}[n] = \frac{\sqrt{3}+1}{\sqrt{2}}d^{(1)}_{j-1}[n]\ . \tag{17}$$

Since there is more than one $U$ operation, we have used superscripts on the $\mathbf{s}$ and $\mathbf{d}$ signals in order to tell them apart. Note that in the normalization steps we have

$$\frac{\sqrt{3}-1}{\sqrt{2}}\cdot\frac{\sqrt{3}+1}{\sqrt{2}} = 1\ .$$

Normalization does not change the information on the signal, but is merely a means to have certain transform properties satisfied.

The Daubechies 4 equations are derived from the classical wavelet theory by a procedure called Euclidean factorization, see Section VI. While the Daubechies 4 transform does have a well-described origin in mathematics (see for instance Ten Lectures on Wavelets [1] by Daubechies), it becomes somewhat unclear in the lifting step version, what exactly this transform can provide. In particular, starting with an update step does not conform with our previous exposition of the lifting approach. However, this transform has one important property not possessed by the linear prediction transform CDF(2,2), or indeed any predicting lifting steps based on polynomial interpolation; in a frequency interpretation of the wavelet transforms the Daubechies 4 behaves somewhat nicer than CDF(2,2). This is because the former is an orthogonal transform, while the latter is a

biorthogonal transform. We refer to other wavelet articles in this collection for more information on this subject.

While the Daubechies 4 is lacking a proper interpretation in terms of prediction and update, and thus may appear to be less appealing to include in the lifting concept, it's ability to be written as lifting step is in fact a consequence of the pleasant fact that all wavelet transforms (not matter their origin and design criteria) can be written as lifting steps. The two examples then show that some lifting steps comes from a sample-by-sample design method, while others may come from completely different design methods (the Daubechies transform was designed within the field of filter theory to have certain nice frequency related properties). This article touches upon the filter subject, see section VI Lifting and Filter Banks. For a more in-depth coverage of the subject, we refer the reader to other articles in this collection..

To find the inverse transform we have to use the prescription given above. We do the steps in reverse order and with the signs reversed. The normalization is undone by multiplication by the inverse constants. The result is

$$d_{j-1}^{(1)}[n] = \frac{\sqrt{3}-1}{\sqrt{2}} d_{j-1}[n] \tag{18}$$

$$s_{j-1}^{(2)}[n] = \frac{\sqrt{3}+1}{\sqrt{2}} s_{j-1}[n] \tag{19}$$

$$s_{j-1}^{(1)}[n] = s_{j-1}^{(2)}[n] + d_{j-1}^{(1)}[n+1] \tag{20}$$

$$s_j[2n+1] = d_{j-1}^{(1)}[n] + \tfrac{1}{4}\sqrt{3} s_{j-1}^{(1)}[n] + \tfrac{1}{4}(\sqrt{3}-2) s_{j-1}^{(1)}[n-1] \tag{21}$$

$$s_j[2n] = s_{j-1}^{(1)}[n] - \sqrt{3} s_j[2n+1] \ . \tag{22}$$

Incidentally, this transform illustrates one of the problems that has to be faced in computer implementations. For example, to compute $d_{j-1}^{(1)}[0]$ we need to know $s_{j-1}^{(1)}[0]$ and $s_{j-1}^{(1)}[-1]$. But to compute $s_{j-1}^{(1)}[-1]$ one needs the values $s_j[-2]$ and $s_j[-1]$, which are not defined. The easiest solution to this problem is to use zero padding to get a sample at this index value (zero padding means that all undefined samples are defined to be 0). There exists other more sophisticated methods, but that topic is beyond the scope of this article.

Finally, let us repeat our first example in the above notation. We also add a normalization step. In this form the transform is known as the Haar transform in the literature (we used this term previously, but in fact the scaling needs to be including for the transform to be the true Haar transform). The direct transform is

$$d_{j-1}^{(1)}[n] = s_j[2n+1] - s_j[2n] \tag{23}$$

$$s_{j-1}^{(1)}[n] = s_j[2n] + \tfrac{1}{2} d_{j-1}^{(1)}[n] \tag{24}$$

$$s_{j-1}[n] = \sqrt{2} s_{j-1}^{(1)}[n] \tag{25}$$

$$d_{j-1}[n] = \frac{1}{\sqrt{2}} d_{j-1}^{(1)}[n] \tag{26}$$

and the inverse transform is given by

$$d_{j-1}^{(1)}[n] = \sqrt{2}d_{j-1}[n] \tag{27}$$

$$s_{j-1}^{(1)}[n] = \frac{1}{\sqrt{2}}s_{j-1}[n] \tag{28}$$

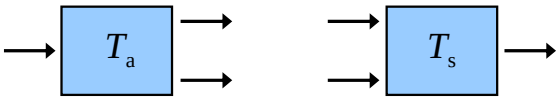
$$s_j[2n] = s_{j-1}^{(1)}[n] - \tfrac{1}{2}d_{j-1}^{(1)}[n] \tag{29}$$

$$s_j[2n+1] = s_j[2n] + d_{j-1}^{(1)}[n]\ . \tag{30}$$

We note that this transform can be applied to a signal of length $2^j$ without using zero padding. It turns out to be essentially the only transform with this property.

### IV.3 Discrete Wavelet Transform

We have now established a method for transforming one signal into another signal that has a natural division into two parts of equal length. This transform is the discrete wavelet transform (DWT). In many cases one is not interested in the actual implementation of the transform, though, and the DWT is in those cases often shown as a box with one input and two outputs. The direct transform is sometimes called 'analysis', and we will represent such a direct transform by the symbol $T_\mathrm{a}$ ('a' for analysis).




In our notation the input would be $\mathbf{s}_j$ and the outputs would be $\mathbf{s}_{j-1}$ and $\mathbf{d}_{j-1}$. The inverse transform is then naturally shown as another box with two inputs and one output, and is sometimes called 'synthesis'. Therefore, the inverse transform is represented by the symbol $T_\mathrm{s}$.

The contents of the $T_\mathrm{a}$ box could be the direct Haar transform as given by (23)–(26), and the contents of the $T_\mathrm{s}$ box could be the inverse Haar transform as given by (27)–(30). Obviously, we must make sure to use the inverse transform corresponding to the applied direct transform. Otherwise, the results will be meaningless.

We can now combine these building blocks to get a multi-scale discrete wavelet transforms. We perform the transform over a certain number of scales $k$, meaning that we combine $k$ of the building blocks as shown in Figure 3 in the case of 2 scales, and with the upper-most diagram in Figure 7 in the case of 4 scales. In the latter figure we use the building block representation of the individual steps. It is possible to apply the DWT to the difference output of the transform as well, instead of just the average (or means) output. This is known as the wavelet packet transform, and is commonly found in the literature, and is often used in practical implementations. This concept introduces new possibilities and methods. We refer to other articles presenting this subject.

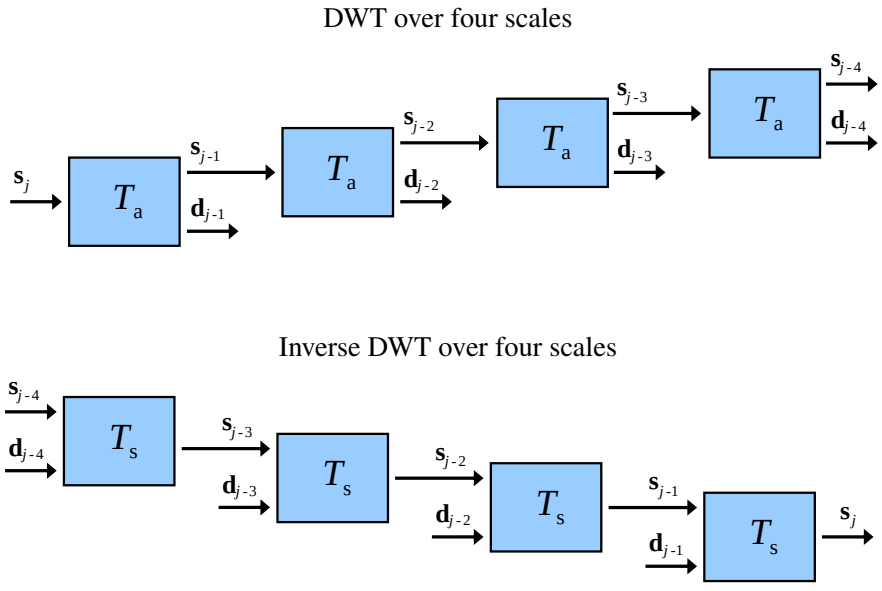



Figure 7: DWT and inverse DWT over four scales.

# V Interpretation

The main topic for this collection of articles is 'Wavelets'. This term refers to a function, often denoted by $\psi$ in the wavelet literature, that takes many shapes and determines the properties of the wavelet transform. So what is the relation between a series of lifting steps and this wavelet function, and is it useful to know this relation, if one just wants to use the wavelet transform? After all, the lifting step method works quite well without any apparent need for knowledge of the wavelet function.

It turns out that while the transform can be implemented easily with just a few equations, the understanding of what is actually happening with a signal, when it is transformed, relies heavily on a proper interpretation of the transform. Here we will present an interpretation based on linear algebra, which will provide some useful insights. However, this presentation is incomplete; we state some results from the general theory, and illustrate them with explicit computations, but we do not discuss the general theory in detail, since this is beyond the scope of this article.

In the following discussion we will continue to use the Haar transform example from the Introduction, partly because it allows for simple and yet convincing computations, partly because we believe that the reader has become familiar with this example by now. We will end with an interpretation of the Daubechies 4 transform.

## V.1 Using Linear Algebra

We saw above that the first of the eight numbers in the transformed signal in the first example in the Introduction, that is, the number 35, had a special meaning; it was the mean of the entire signal. This property (though not the number itself) was actually independent of the signal; for any signal that first number $s_0[0]$ will always be the mean of the signal. In fact, it was a property derived from the transform, and it provided an interpretation of the number 35 in the transformed signal. Similar interpretations can be made of the other numbers in the transformed signal.

First, we examine the original example in Table 1 once more.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 56 | 40 | 8 | 24 | 48 | 48 | 40 | 16 |
| 48 | 16 | 48 | 28 | **8** | **−8** | **0** | **12** |
| 32 | 38 | **16** | **10** | **8** | **−8** | **0** | **12** |
| 35 | **−3** | **16** | **10** | **8** | **−8** | **0** | **12** |

This table was constructed from top to bottom by multiple Haar transforms, and the 'signal mean' property was derived using equations, see for instance (9). However, we can reach the same conclusion by examining signal samples, and by starting at the bottom. To isolate the property of interest here, we start with a bottom row consisting of zeros at the entries we are not interested in interpreting, and a 1 at the entry we would like to interpret, that is, the first of the eight numbers. Then we do a three scale synthesis of the signal and get

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | **0** | **0** | **0** | **0** |
| 1 | 1 | **0** | **0** | **0** | **0** | **0** | **0** |
| 1 | **0** | **0** | **0** | **0** | **0** | **0** | **0** |

Note that the normalization steps (25) and (26) has been omitted to make the table easier to read (this does not interfere with the interpretation, even though this fact may not be clear at this point). It is obvious that if we had started with any other number than 1, this number would then be the one repeated eight times in the top row after the three scale synthesis. This computation indicates that the entry at location 0 (recall that our indexing starts with 0) in the fourth row has an equal contribution from all eight original samples, and thus is the mean value.

Let us repeat this procedure with the remaining seven entries in the fourth row. Here are the first two, having a 1 at entry number 1 and 2.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | −1 | −1 | −1 | −1 |
| 1 | 1 | −1 | −1 | **0** | **0** | **0** | **0** |
| 1 | −1 | **0** | **0** | **0** | **0** | **0** | **0** |
| 0 | **1** | **0** | **0** | **0** | **0** | **0** | **0** |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | −1 | −1 | 0 | 0 | 0 | 0 |
| 1 | −1 | 0 | 0 | **0** | **0** | **0** | **0** |
| 0 | 0 | **1** | **0** | **0** | **0** | **0** | **0** |
| 0 | **0** | **1** | **0** | **0** | **0** | **0** | **0** |

Examining the first row in the left-most table reveals that the value at entry number 1, i.e. the value 1 in the fourth row, can be interpreted as the mean of the first four sample minus the mean of the last four sample. In filter terms this is roughly equal to a band pass filtering just above DC, i.e. DC does not contribute to entry number 1, and neither would any frequency higher than a single cycle.

The result of all eight computations are now represented using notation from linear algebra; they are inserted as columns in a matrix

$$\mathbf{W}_{\mathrm{s}}^{(3)} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & -1 & 0 & 0 & 0 \\ 1 & 1 & -1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & -1 & 0 & 0 & -1 & 0 & 0 \\ 1 & -1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & -1 & 0 & 1 & 0 & 0 & -1 & 0 \\ 1 & -1 & 0 & -1 & 0 & 0 & 0 & 1 \\ 1 & -1 & 0 & -1 & 0 & 0 & 0 & -1 \end{bmatrix}, \tag{31}$$

which is denoted by $\mathbf{W}$ with a subscript $\mathrm{s}$ for synthesis for the following reason. A signal of length $8$ is a vector in the vector space $\mathbf{R}^8$. The process described above of reconstructing the signal, whose transform is one of the canonical basis vectors in $\mathbf{R}^8$, is the same as finding the columns in the matrix (with respect to the canonical basis) of the three scale synthesis transform. This is just the well known definition from linear algebra of the matrix of a linear transform. This in turn means that applying this matrix to any length 8 signal performs the inverse Haar transform. For example, multiplying it with the fourth row of Table 1 (regarded as a column vector) produces the first row of that same table.

The matrix of the direct three scale transform is obtained in a similar fashion; by computing the transforms of the eight canonical basis vectors in $\mathbf{R}^8$. In other words, we start with the signal $[1, 0, 0, 0, 0, 0, 0, 0]$ and carry out the three scale transform as shown here.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $1$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ | $0$ |
| $\frac{1}{2}$ | $0$ | $0$ | $0$ | $\mathbf{\frac{1}{2}}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ |
| $\frac{1}{4}$ | $0$ | $\mathbf{\frac{1}{4}}$ | $\mathbf{0}$ | $\mathbf{\frac{1}{2}}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ |
| $\frac{1}{8}$ | $\mathbf{\frac{1}{8}}$ | $\mathbf{\frac{1}{4}}$ | $\mathbf{0}$ | $\mathbf{\frac{1}{2}}$ | $\mathbf{0}$ | $\mathbf{0}$ | $\mathbf{0}$ |

Computing the other seven transforms and inserting the resulting signals as column vectors gives

$$\mathbf{W}_{\mathrm{a}}^{(3)} = \begin{bmatrix} \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} \\ \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & -\frac{1}{8} & -\frac{1}{8} & -\frac{1}{8} & -\frac{1}{8} \\ \frac{1}{4} & \frac{1}{4} & -\frac{1}{4} & -\frac{1}{4} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{4} & \frac{1}{4} & -\frac{1}{4} & -\frac{1}{4} \\ \frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & -\frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & -\frac{1}{2} \end{bmatrix}. \tag{32}$$

Multiplying the matrices we find $\mathbf{W}_{\mathrm{s}}^{(3)} \cdot \mathbf{W}_{\mathrm{a}}^{(3)} = \mathbf{I}$ and $\mathbf{W}_{\mathrm{a}}^{(3)} \cdot \mathbf{W}_{\mathrm{s}}^{(3)} = \mathbf{I}$, where $\mathbf{I}$ denotes the $8 \times 8$ identity matrix. This is the linear algebra formulation of perfect reconstruction, or of the invertibility of the three scale transform.

Another interesting property to note here is that the structures of the two matrices are quite similar. The transpose of the one matrix bears a striking resembles the other matrix. In fact, if the above computations of the direct and inverse transforms of the canonical basis vectors had used the normalization steps, we would get an orthogonal matrix and its transpose, which by elementary linear algebra equals its inverse. Any wavelet transform with this property is called an orthogonal transform (i.e. all columns are orthogonal to each other and all have norm 1). All transforms in the Daubechies family have this property, and so do other well-known families like Coiflets and Symlets. The CDF families are not orthogonal (but rather bi-orthogonal), and consequently the structure of the CDF matrices is slightly more complicated.

It is clear that analogous constructions can be carried out for signal of length $2^j$ and transforms to all scales $k = 1, \ldots, j$. The linear algebra point of view is useful in understanding the theory. However, if one is trying to carry out numerical computations, then it is a bad idea to use the matrix formulation. The direct $k$ scale wavelet transform using the lifting steps requires a number of operations (additions, multiplications, etc.) on the computer, which is proportional to the length $L$ of the signal. If we perform the transform using its matrix, then in general a number of operations proportional to $L^2$ is needed. For longer signals this makes a huge difference for the computation time.

## V.2 From Transform to Wavelets

Let us now consider the column vectors in (31) as a sampling of continuous, piecewise constant signals on the interval $[0; 1]$. These continuous signals are shown in Figure 8. In this figure the signals have been group according to their origin. Recall that the first column came from a three scale synthesis of $[1, 0, 0, 0, 0, 0, 0, 0]$. This is equivalent to letting $\mathbf{s}_0 = 1$ and $\mathbf{d}_0 = 0$, $\mathbf{d}_1 = \mathbf{0}$, and $\mathbf{d}_2 = \mathbf{0}$ and using a three scale inverse DWT as shown in Figure 7. The four right-most graphs in Figure 8 can be obtained by inverse transforms where $\mathbf{d}_2$ equals $[1, 0, 0, 0]$, $[0, 1, 0, 0]$, $[0, 0, 1, 0]$, and $[0, 0, 0, 1]$, respectively, and $\mathbf{s}_0 = 0$, $\mathbf{d}_0 = 0$, and $\mathbf{d}_1 = \mathbf{0}$ in all four cases. The $\mathbf{s}$ group

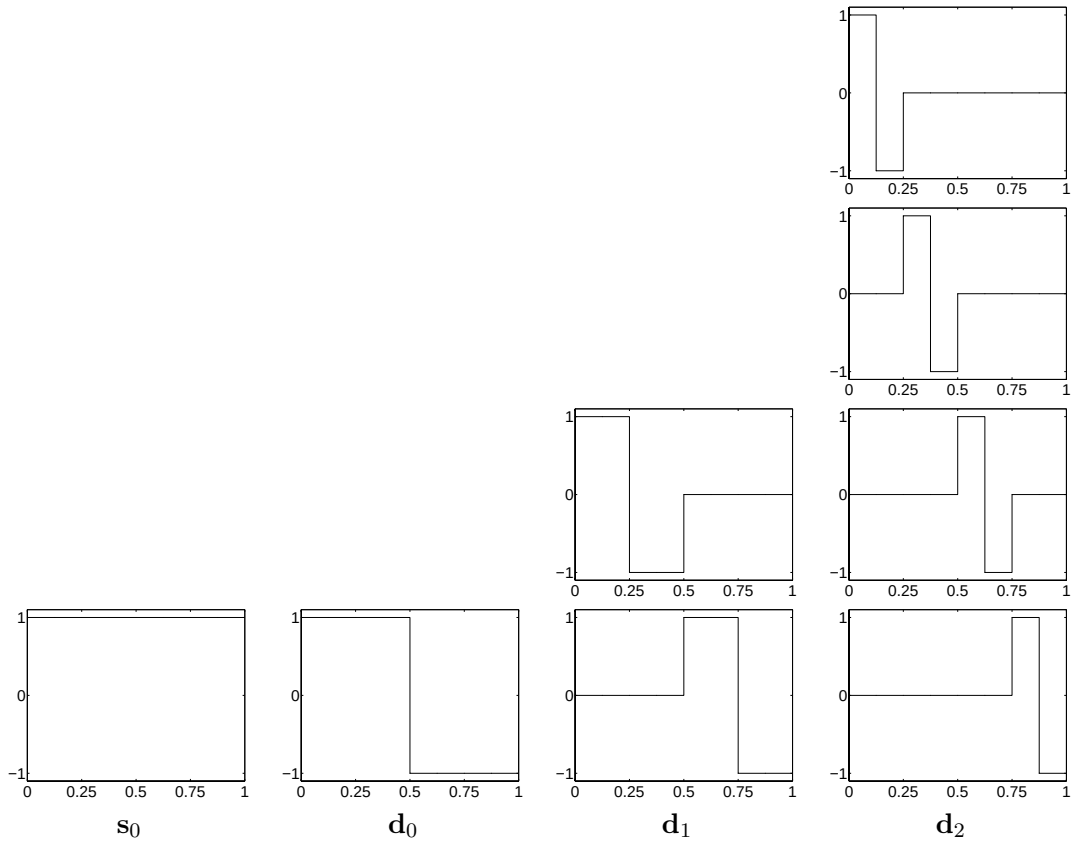



Figure 8: Graphs showing the 8 column vectors in (31) on the interval $[0; 1]$. The graphs are sorted according to the multiscale decomposition shown in Figure 7.

consists of only one signal, which is constant. The $\mathbf{d}$ groups all consists of signals with the same shape (positive signal value followed by negative signal value). The three $\mathbf{d}$ differ by a factor 2 scaling in the $x$ axis direction, and the signals the $\mathbf{d}_1$ and $\mathbf{d}_2$ groups differ by translation along the $x$ axis.

This pattern can be contained in a single equation. First, define the function

$$\psi(t) = \begin{cases} 1, & t \in [0; \frac{1}{2}[ \ , \\ -1, & t \in [\frac{1}{2}; 1] \ , \\ 0, & \text{otherwise} \ . \end{cases} \tag{33}$$

This is in fact the $\mathbf{d}_0$ signal. Now all the other $\mathbf{d}$ signal can be obtained by

$$\psi_{j,k}(t) = \psi(2^j t - k) \ ,$$

where $j$ is equal to 0, 1, or 2 for the groups $\mathbf{d}_0$, $\mathbf{d}_1$, and $\mathbf{d}_2$, and $k$ ranges from 0 to $2^j - 1$. The $\psi$ function is the wavelet. Sometimes it is called the mother wavelet to signify that a wavelet transform consists of numerous similar functions originating from one function. If we include the normalization step the equation reaches its final form

$$\psi_{j,k}(t) = 2^{-j/2}\psi(2^j t - k) \ .$$

This framework does not include the $\mathbf{s}_0$. However, another equation combines $\mathbf{s}_0$ and $\mathbf{d}_0$. We noticed previously that while the vector $[1, 1, 1, 1, 1, 1, 1, 1]$, the first column in (31), represents the mean of the original signal, the vector $[1, 1, 1, 1, -1, -1, -1, -1]$ is mean of the first half of the original signal minus the mean of the other half of the signal. This link can be expressed with continuous functions. First, define

$$\phi(t) = \begin{cases} 1, & t \in [0; 1] \ , \\ 0, & \text{otherwise} \ , \end{cases}$$

that is, the function corresponding to the continuous $\mathbf{s}_0$ signal. This function is called the scaling function in wavelet literature. Now we can link $\mathbf{s}_0$ and $\mathbf{d}_0$ with

$$\psi(t) = \phi(2t) - \phi(2t - 1) \ . \tag{34}$$

This equation is called the two-scale equation, and is found in virtually any literature presenting the wavelet theory. Its general form is

$$\psi(t) = \sum_n g_n \phi(2t - n) \ , \tag{35}$$

and it is valid for all orthogonal wavelet transforms. For the Haar transform we have $g_0 = 1$ and $g_1 = -1$ and $g_n = 0$ for all indices other than 0 and 1. The $g_n$ for the Daubechies 4 transform are

$$g_0 = -\frac{1 - \sqrt{3}}{4\sqrt{2}}, \quad g_1 = \frac{3 - \sqrt{3}}{4\sqrt{2}}, \quad g_2 = -\frac{3 + \sqrt{3}}{4\sqrt{2}}, \quad g_3 = \frac{1 + \sqrt{3}}{4\sqrt{2}} \ .$$

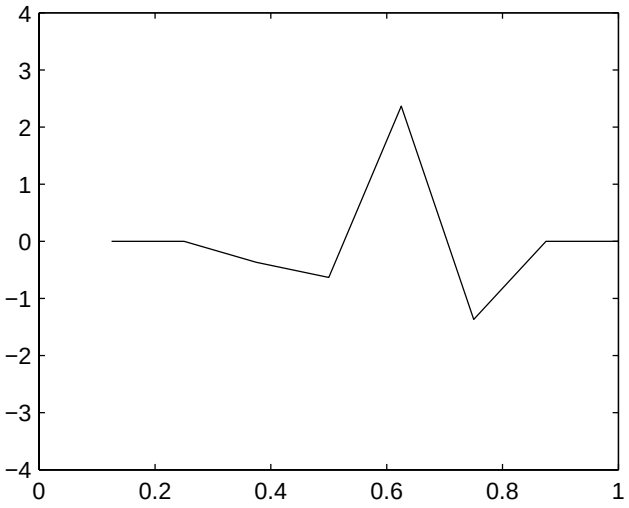



Figure 9: Inverse Daubechies 4 of $[0, 0, 0, 0, 0, 1, 0, 0]$ over three scales.

There is also a two-scale equation that produces the scaling function.

$$\phi(t) = \sum_n h_n \phi(2t - n) \ . \tag{36}$$

For orthogonal transforms $\mathbf{h}$ is obtained from $\mathbf{g}$ by $h_n = (-1)^n g_{1-n}$, that is, reverse sequence order with alternating change of sign and index shift by 1.

## V.3 Wavelet and Scaling Function for Daubechies 4

The appearance of the Haar wavelet function through transform synthesis in the previous section may appear to be a special property of the simple Haar transform. However, the principle applies to all wavelets, and the method for generating the wavelet function can be used to obtain any wavelet, at least as a graph. Actually, in many cases no closed form of the wavelet exist. This includes the Daubechies family.

Thus, to see the shape of the Daubechies 4 wavelet we start again with an 'all but one entry is zero'-signal. We saw that in fact when obtaining the wavelet graph it does not matter which entry is non-zero, as long as it is not the first entry. The only difference in the output is the dilation and translation of the resulting function. As an example we therefore choose $[0, 0, 0, 0, 0, 1, 0, 0]$. We perform a inverse three scale transform using the inverse Daubechies 4 equations (18) – (22) and plot the result, see Figure 9. Note that the problem with finite signals needs to be addressed in this inverse transform. For our purpose zero padding is sufficient. This figure contains very little information. But let us now repeat the procedure for vectors of length $8$, $32$, $128$, and $512$, applied to a vector with a single $1$ as its sixth entry. This requires inverse transforms over $3$, $5$, $7$, and

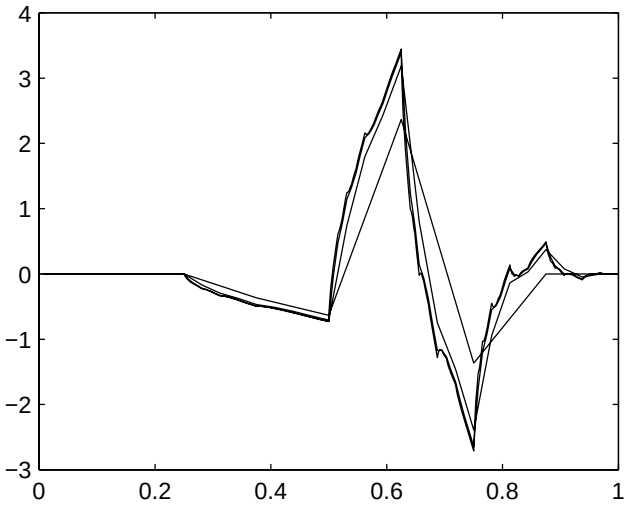



Figure 10: Inverse Daubechies 4 of sixth basis vector, length $8$, $32$, $128$ and $512$.

$9$ scales, respectively. We fit each transform to the same interval, as if we have a finer and finer resolution. The result is shown in Figure 10. This figure shows that the graphs rapidly approach a limiting graph, as we increase the length of the vector. This is a result that can be established rigorously, but it is not easy to do so, and it is beyond the scope of this article.

One can interpret the limiting function in Figure 10 as a function whose values, sampled at appropriate points, represent the entries in the inverse transform of a vector of length $2^N$, with a single 1 as its sixth entry. For $N$ just moderately large, say $N = 12$, this is a very good approximation to the actual value. See Figure 11 for the result for $N = 12$, i.e. a vector of length $4096$.

## V.4 Wavelet and Scaling function for CDF(2,2)

Finally, let us repeat the procedure for the CDF(2,2) transform in (10) and (11), and at the same time illustrate how the wavelet is translated depending on the placement of the $1$ in the otherwise $0$ vector. An example is given in Figure 12. The difference in the graphs in Figure 12 and Figure 11 is striking. It reflects the result that the Daubechies 4 wavelet has very little regularity (it is not differentiable), whereas the CDF(2,2) wavelet is a piecewise linear function. Recall that the Haar and Daubechies 4 transform are orthogonal transforms, which means that in linear algebra terms the transpose of the synthesis matrix equals the analysis matrix. A consequence of this is that the same wavelet and scaling function are used for analysis and for synthesis. This is not the case for CDF(2,2), which is a biorthogonal transform. Therefore, it has one synthesis pair of scaling function and wavelets, and an analysis pair. If we use the forward transform instead of the inverse we can obtain the analysis functions. They are shown in Figure 13. These functions are quite

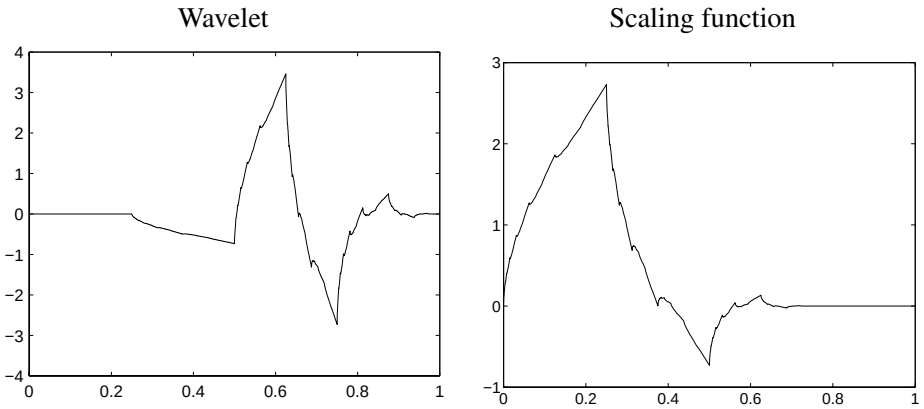



Figure 11: Inverse Daubechies 4 of sixth and first basis vectors, both length $4096$. The result is the Daubechies 4 wavelet on the left and the scaling function on the right.

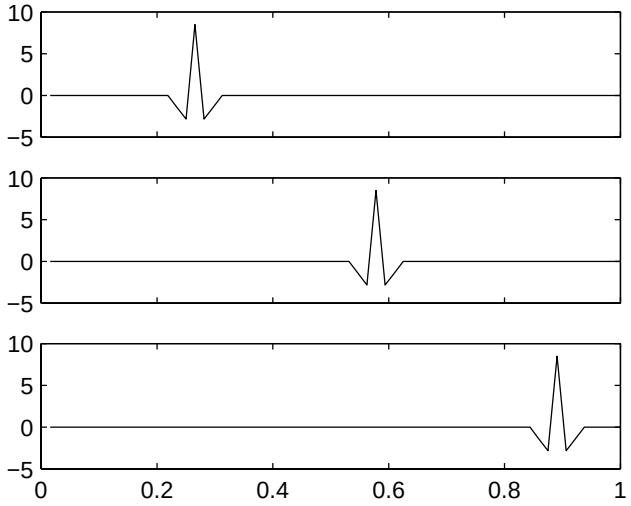


Figure 12: Inverse CDF(2,2) of three basis vectors of length $64$, entry $40$, or $50$, or $60$, equal to $1$ and the remaining entries equal to zero. The result is the same function (the wavelet) with different translations.

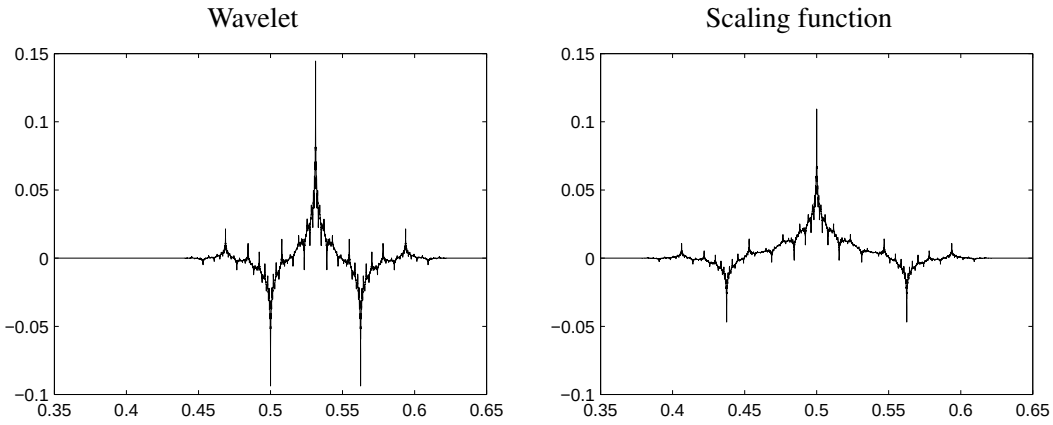



Figure 13: Wavelet and Scaling function for CDF(2,2).

complicated, and it is interesting to see that while the analysis wavelet and scaling function are very simple functions (we have not shown the scaling function of CDF(2,2) though) the inverse of that same transform have some rather complex wavelet and scaling functions. As an end remark notice that all CDF(2,2) functions are symmetrical. This is a special property that can only be achieved by biorthogonal wavelets. Orthogonal wavelets can come close to symmetry, but they can never become completely symmetrical. An almost symmetrical family has been constructed, called the Symlets, see for example [4].

## V.5 The General Case

The above computations lead us to the conclusion that there are just two functions underlying the direct transform, and another two functions underlying the inverse transform, in the sense that if we take sufficiently long vectors, say $2^N$, and perform a $k$ scale transform, with $k$ large, then we get values that are sampled values of one of the underlying functions. More precisely, inverse transforms of unit vectors with a one in places from 1 to $2^{N-k}$ yield translated copies of the scaling function. Inverse transforms of unit vectors with a one in places from $2^{N-k}+1$ to $2^{N-k+1}$ yield translated copies of the wavelet. Finally, inverse transforms of unit vectors with a one at places from $2^{N-k+1}+1$ to $2^N$ yield scaled and translated copies of the wavelet.

These results are strictly correct only in a limiting sense, and they are not easy to establish. There is one further complication which we have omitted to state clearly. If one performs the procedure above with a 1 close to the start or end of the vector, then there will in general be some strange effects, depending on how the transform has been implemented. We refer to these as boundary effects. They depend on how one makes up for missing samples in computations near the start or end of a finite vector, the so-called boundary corrections. We have already mentioned zero padding as one of the correction methods. This is what we have used indirectly in plotting for example Figure 11, where we have taken a vector with a one at place 24, and zeroes everywhere

else.

Readers interested in a rigorous treatment of the interpretation of the transforms given here, and with the required mathematical background, are referred to the literature, for example the books by I. Daubechies [1], S. Mallat [4], and M. Vetterli-J. Kovačević [10, 11]. Note that these books base their treatment of the wavelet transforms on the concepts of multiresolution analysis and filter theory rather than lifting.

# VI Lifting and Filter Banks

So far, the lifting technique has been carried out in the time domain; the correlation of samples was based on odd and even entries, interpretation was based on the signal in the time domain, and every reference to the signal was made by $\mathbf{s}$ and $\mathbf{d}$, i.e. time representations of the signals. However, there is much to be learned about wavelets and about lifting when turning to the frequency domain.

The lifting method is a special case of a standard frequency-based signal processing method called filter banks. A filter bank is a series of bandpass filters which separates the input signal into a number of components, each with a distinct range of frequencies from the original signal. Often, in a filter bank, the filters are designed such that no information is lost, that is, it is possible to reconstruct the original signal from the components. The outputs of a filter bank are called subband signals, and it has as many subbands as there are filters in the filter bank.

The following introduction to lifting in the frequency domain requires some knowledge of the $z$-transform as well as basic Fourier theory. To ease the reading, we briefly state some important relations for the $z$-transform. The transform maps a signal $\mathbf{x} = \{x[n]\}$ to a function defined in the complex plane

$$X(z) = \sum_{n \in \mathbf{Z}} x[n] z^{-n} \ .$$

This is equivalent to the Fourier transform, when $z = e^{j\omega}$, and we use capital $X$ to denote the $z$-transform of $\mathbf{x}$. In the following exposition we will needed up and down sampling by two of a signal. In the time domain this is accomplished by inserted zeros between all samples and by removing every other sample, respectively. Explicitly, if $\mathbf{x} = \{x[n]\}$, then the down sampled sequence $\mathbf{x}_{2\downarrow}$ is given by $x_{2\downarrow}[n] = x[2n]$. The up sampled sequence $\mathbf{x}_{2\uparrow}$ is given by $x_{2\uparrow}[n] = x[n/2]$, if $n$ is even, and $x_{2\uparrow}[n] = 0$, if $n$ is odd.

In the $z$-transform up sampling of $X(z)$ is accomplished by $X(z^2)$, and down sampling is accomplished by

$$\tfrac{1}{2}\big(X(z^{1/2}) + X(-z^{1/2})\big) \ . \tag{37}$$

Finally, we note that in this section all signals have finite length, and thus the $z$-transform has a finite number of non-zero terms. The $z$-transform is therefore a Laurent polynomial, which is a polynomial in the variables $z$ and $z^{-1}$.

## VI.1 Lifting in the z-Transform Representation

The lifting method consists of a three basic operations; it is composed of splitting the signal in odd and even samples, predicting an odd sample, and updating an even sample. This is shown in Figure 3. The splitting is given in the $z$-transform representation by

$$X(z) = X_0(z^2) + z^{-1}X_1(z^2)\ , \tag{38}$$

where

$$X_0(z) = \sum_n x[2n]z^{-n} = \tfrac{1}{2}\big(X(z^{1/2}) + X(-z^{1/2})\big)\ , \tag{39}$$

$$X_1(z) = \sum_n x[2n+1]z^{-n} = \tfrac{1}{2}z^{1/2}\big(X(z^{1/2}) - X(-z^{1/2})\big)\ , \tag{40}$$

are the $z$-transforms of the even and odd samples. Note that the second equality in both formulas comes from (37). We represent this decomposition by the left side of the diagram in Figure 14. In the diagram the time shift is inserted prior to the down sampling, rather than after as in (40), since this gives multiplication with $z$ instead of $z^{1/2}$. The decomposition in (38) is called a polyphase decomposition (with two components). The inverse operation is obtained by reading equation (38) from right to left. The equation tells us that we can obtain $X(z)$ from $X_0(z)$ and $X_1(z)$ by first up sampling the two components by 2, then shifting $X_1(z^2)$ one time unit right (by multiplication by $z^{-1}$), and finally adding the two components. We represent this reconstruction by the right hand side of the diagram in Figure 14. Notice that as with the lifting method this inversion simply 'undoes' the changes made to the signal. The transform pair represented in Figure 14 is sometimes in the literature called the 'lazy' wavelet transform and its inverse.

### Prediction and Update Steps

Let us now see how we can implement the prediction step from Section III in the $z$-transform representation. The prediction technique was to form a linear combination of the even entries and

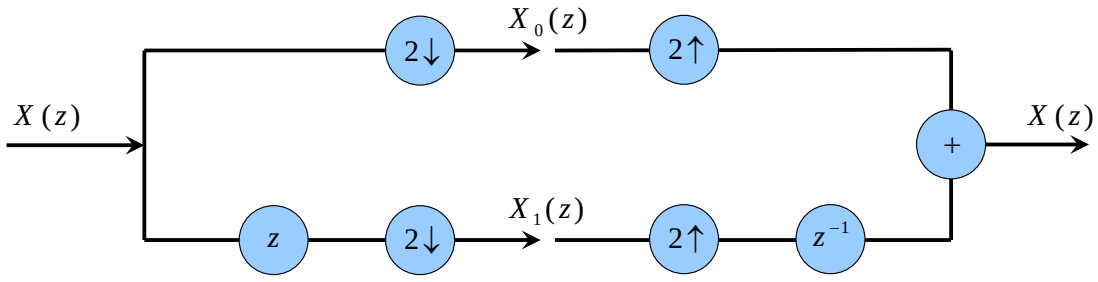



Figure 14: Splitting in even and odd components by means of up sampling followed by reconstruction of the signal from the odd and even components.

then subtract the result from the odd entry under consideration. The linear combination was formed independently of the index of the odd sample under consideration, and based only on the relative location of the even entries. For example, in the CDF(2,2) transform the first step in (10) can be implemented as $X_1(z) - T(z)X_0(z)$, where $T(z) = \frac{1}{2}(1+z)$. This is because $T(z)X_0(z)$ is the convolution $\mathbf{t} * \mathbf{x}_0$ in the time domain, which is exactly a linear combination of the even entries with weights $t_n$. Explicitly,

$$\begin{aligned} X_1(z) - T(z)X_0(z) &= X_1(z) - \tfrac{1}{2}(1+z)\sum_n x[2n]z^{-n} \\ &= X_1(z) - \tfrac{1}{2}\sum_n x[2n]z^{-n} + \tfrac{1}{2}\sum_n x[2n]z^{-n+1} \\ &= \sum_n x[2n+1] - \sum_n \tfrac{1}{2}(x[2n] + x[2n+2])z^{-n} \\ &= \sum_n \left(x[2n+1] - \tfrac{1}{2}(x[2n] + x[2n+2])\right) z^{-n} \,, \end{aligned}$$

which is the $z$-transform representation of the right hand side of (10). The transition is described by matrix multiplication as in

$$\begin{bmatrix} X_0(z) \\ X_1(z) - T(z)X_0(z) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -T(z) & 1 \end{bmatrix} \begin{bmatrix} X_0(z) \\ X_1(z) \end{bmatrix}.$$

Here we use $2 \times 2$ matrices whose entries are Laurent polynomials. An entirely analogous computation shows that if we define $S(z) = \frac{1}{4}(1+z^{-1})$, then the update step in (11) is implemented in the $z$-transform representation as multiplication by the matrix

$$\begin{bmatrix} 1 & S(z) \\ 0 & 1 \end{bmatrix}.$$

The normalization step, as for example given in (26) and (25), can be implemented by multiplication by a matrix of the form

$$\begin{bmatrix} K & 0 \\ 0 & K^{-1} \end{bmatrix},$$

where $K > 0$ is a constant.

**Entire Transform**

Since every prediction step is subtraction of a linear combination of odd samples, any prediction step can be implemented by means of multiplication by a matrix of the form

$$\mathbf{P}(z) = \begin{bmatrix} 1 & 0 \\ -T(z) & 1 \end{bmatrix}. \tag{41}$$

The same result applies to any update step, which can be implemented by multiplication by a matrix of the form

$$\mathbf{U}(z) = \begin{bmatrix} 1 & S(z) \\ 0 & 1 \end{bmatrix}. \tag{42}$$

Here $T(z)$ and $S(z)$ are both Laurent polynomials.

The general one scale DWT is a combination of multiple prediction and updates steps and one normalization step. This is shown in Figure 6 (although the normalization is not included in the figure). In the $z$-transform representation this becomes

$$\mathbf{H}(z) = \begin{bmatrix} K & 0 \\ 0 & K^{-1} \end{bmatrix} \begin{bmatrix} 1 & S_N(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -T_N(z) & 1 \end{bmatrix} \cdots \begin{bmatrix} 1 & S_1(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -T_1(z) & 1 \end{bmatrix}. \tag{43}$$

The order of the factors is determined by the order in which the steps are applied. First a prediction step, then an update step, repeated $N$ times, and then finally the normalization step.

An important property of the DWT implemented via lifting steps was the invertibility of the transform, as illustrated for example in Figure 4. It is easy to verify that we have

$$\mathbf{P}(z)^{-1} = \begin{bmatrix} 1 & 0 \\ T(z) & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{U}(z)^{-1} = \begin{bmatrix} 1 & -S(z) \\ 0 & 1 \end{bmatrix}. \tag{44}$$

We note that the inverse matrix is again a matrix with Laurent polynomials as entries. Thus, the inversion of the transform is accomplished simply by inverting each $2 \times 2$ matrix and reversing their order.

$$\mathbf{H}^{-1}(z) = \mathbf{G}(z) = \begin{bmatrix} 1 & 0 \\ T_1(z) & 1 \end{bmatrix} \begin{bmatrix} 1 & -S_1(z) \\ 0 & 1 \end{bmatrix} \cdots \begin{bmatrix} 1 & 0 \\ T_N(z) & 1 \end{bmatrix} \begin{bmatrix} 1 & -S_N(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} K^{-1} & 0 \\ 0 & K \end{bmatrix}. \tag{45}$$

Multiplying out the matrices in the product defining $\mathbf{H}(z)$ in (43), we get a $2 \times 2$ matrix with entries, which are Laurent polynomials. We use the notation

$$\mathbf{H}(z) = \begin{bmatrix} H_{00}(z) & H_{01}(z) \\ H_{10}(z) & H_{11}(z) \end{bmatrix}, \tag{46}$$

for such a general matrix. Written in matrix notation the DWT is given as

$$\begin{bmatrix} Y_0(z) \\ Y_1(z) \end{bmatrix} = \begin{bmatrix} H_{00}(z) & H_{01}(z) \\ H_{10}(z) & H_{11}(z) \end{bmatrix} \begin{bmatrix} X_0(z) \\ X_1(z) \end{bmatrix}, \tag{47}$$

and we can then represent the implementation of the complete one scale DWT in the $z$-transform representation by the diagram in Figure 15. This representation of a two channel filter bank, without any reference to lifting, is in the signal analysis literature called the polyphase representation, see for example [7, 10, 11].

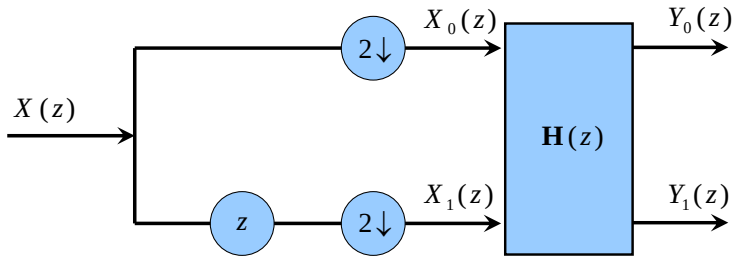



Figure 15: One scale DWT in the $z$-representation as given in (47). This is the polyphase representation and it comes directly from the lifting step method.

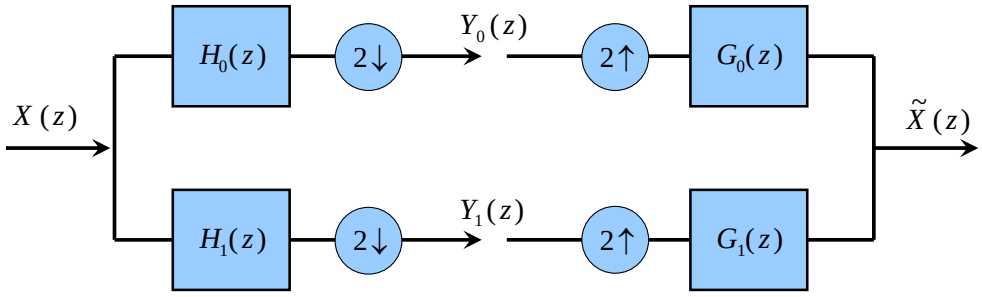



Figure 16: Two channel analysis and synthesis. This is the standard filter bank representation.

## VI.2 Two Channel Filter Banks with Perfect Reconstruction

We have now seen how the time-based lifting method has an equivalent representation in the frequency domain. Lifting allows for perfect reconstruction of the signal after transformation, and we will now use this property (which is preserved in the frequency representation) to better understand the link between lifting and filter banks. We assume the reader is familiar with the concept of filtering and filter banks. Details on filtering can be found in any book on signal analysis, for example in [7, 6]. The filter bank approach to wavelets is the one used in most introductions to the subject.

A two channel filter bank starts with two analysis filters, here denoted by the filter taps $\mathbf{h}_0$ and $\mathbf{h}_1$, and two synthesis filters, denoted by $\mathbf{g}_0$ and $\mathbf{g}_1$. Usually the filters with index 0 are chosen to be low pass filters, and the filters with index 1 to be high pass filters. The analysis and synthesis parts of the filter bank are shown in Figure 16. The analysis part transforms the input $X(z)$ to the output pair $Y_0(z)$ and $Y_1(z)$. The synthesis part then transforms this pair to the output $\widetilde{X}(z)$. The filtering scheme is said to have the perfect reconstruction property, if $X(z) = \widetilde{X}(z)$ for any $X(z)$.

The main point that we will now establish is that the two figures 15 and 16 do indeed show the same thing, and that it is possible to get the one from the other.

**Conditions for Perfect Reconstruction**

To do this we first analyze which conditions are needed on the four filters in Figure 16 in order to obtain the perfect reconstruction property. Filtering by $\mathbf{h}_0$ transforms $X(z)$ to $H_0(z)X(z)$, and we then use (37) to down sample by two. Thus we have

$$Y_0(z) = \tfrac{1}{2}\Big(H_0(z^{1/2})X(z^{1/2}) + H_0(-z^{1/2})X(-z^{1/2})\Big)\ , \tag{48}$$

$$Y_1(z) = \tfrac{1}{2}\Big(H_1(z^{1/2})X(z^{1/2}) + H_1(-z^{1/2})X(-z^{1/2})\Big)\ . \tag{49}$$

Up sampling by two followed by filtering by the $G$-filters, and addition of the results leads to a reconstructed signal

$$\widetilde{X}(z) = G_0(z)Y_0(z^2) + G_1(z)Y_1(z^2)\ . \tag{50}$$

Perfect reconstruction means that $\widetilde{X}(z) = X(z)$. We combine the above expressions and then regroup terms to get

$$\begin{aligned}\widetilde{X}(z) = \quad & \tfrac{1}{2}\big[G_0(z)H_0(z) + G_1(z)H_1(z)\big]X(z) \\ & + \tfrac{1}{2}\big[G_0(z)H_0(-z) + G_1(z)H_1(-z)\big]X(-z)\ .\end{aligned}$$

To fulfill the condition $\widetilde{X}(z) = X(z)$ we need

$$G_0(z)H_0(z) + G_1(z)H_1(z) = 2\ , \tag{51}$$

$$G_0(z)H_0(-z) + G_1(z)H_1(-z) = 0\ . \tag{52}$$

These conditions mean that the four filters cannot be chosen independently, if we want to have perfect reconstruction. To determine what conditions these equations impose on $H$ and $G$ it is convenient to write them as a matrix equation,

$$\begin{bmatrix} H_0(z) & H_1(z) \\ H_0(-z) & H_1(-z)\end{bmatrix}\begin{bmatrix} G_0(z) \\ G_1(z)\end{bmatrix} = \begin{bmatrix} 2 \\ 0\end{bmatrix}. \tag{53}$$

We want to solve this equation, that is, determine $G_0(z)$ and $G_1(z)$ as functions of $H_0(z)$ and $H_1(z)$. Since the matrix is invertible (it performs a DWT that can be inverted), we can use Cramer's rule to get

$$G_0(z) = \frac{\begin{vmatrix} 2 & H_1(z) \\ 0 & H_1(-z)\end{vmatrix}}{d(z)} = 2d(z)^{-1}H_1(-z)\ .$$

The determinant $d(z)$ of an invertible $2\times 2$ matrix with Laurent polynomials as entries is a monomial (see [3]), which in this case has the property that $d(-z) = -d(z)$. Therefore, the determinant can be assumed to be of the form $d(z) = \frac{1}{2}C^{-1}z^{-2k-1}$ for some integer $k$ and some real constant

$C$. Thus,

$$G_0(z) = \frac{\begin{vmatrix} 2 & H_1(z) \\ 0 & H_1(-z) \end{vmatrix}}{\frac{1}{2}C^{-1}z^{-2k-1}} = Cz^{2k+1}H_1(-z)\ , \tag{54}$$

$$G_1(z) = \frac{\begin{vmatrix} H_0(z) & 2 \\ H_0(-z) & 0 \end{vmatrix}}{\frac{1}{2}C^{-1}z^{-2k-1}} = -Cz^{2k+1}H_0(-z)\ . \tag{55}$$

These equations show that we can choose either the $H$-filter pair or the $G$-filter pair. We will assume that we have filters $H_0$ and $H_1$, subject to the condition that

$$d(z) = H_0(z)H_1(-z) - H_0(-z)H_1(z) = \tfrac{1}{2}C^{-1}z^{-2k-1} \tag{56}$$

for some integer $k$ and nonzero constant $C$. Then $G_0$ and $G_1$ are determined by the equations (54) and (55), which implies that they are unique up to a scaling factor and an odd shift in time. Note that this argument is valid for any two channel filter bank, subject to the condition (56).

## VI.3 Lifting and Two Channel Filter Banks with Perfect Reconstruction are Equivalent

Many presentations of the wavelet transform start with a two channel filter bank with the perfect reconstruction property. The analysis part is then used to define the direct one scale DWT, and the synthesis part is used for reconstruction. It is an important result that the filtering approach, and the one based on lifting, actually are identical. Thus any set of lifting steps leads to a two channel filter bank with perfect reconstruction, and, quite remarkably, vice versa. This means that they are just two different ways of describing the same transformation from $X(z)$ to $Y_0(z)$ and $Y_1(z)$. This equivalence will be the subject of the remainder of the article.

The first step is to show that the analysis step in Figure 16 coming from the two channel filter bank is equivalent to the analysis step summarized in Figure 15 and in (47), i.e. coming from the lifting method. Thus, we want to find the equations relating the coefficients in the $\mathbf{H}$ matrix (46) and the filters $H_0$, $H_1$, $G_0$, and $G_1$. The analysis step by both methods should yield the same result. To avoid the square root terms we compare the results after up sampling by two. We start with the equality

$$\begin{bmatrix} Y_0(z^2) \\ Y_1(z^2) \end{bmatrix} = \mathbf{H}(z^2) \begin{bmatrix} \frac{1}{2}\left(X(z) + X(-z)\right) \\ \frac{1}{2}z\left(X(z) - X(-z)\right) \end{bmatrix},$$

where the left hand side is from the filter bank approach (48) and (49), and the right hand side from the lifting approach with polyphase matrix (46) and up sampled $X_0(z)$ and $X_1(z)$. The first equation can then be written as

$$\tfrac{1}{2}\big(H_0(z)X(z) + H_0(-z)X(-z)\big) = H_{00}(z^2)\tfrac{1}{2}\big(X(z) + X(-z)\big) + H_{01}(z^2)\tfrac{1}{2}z\big(X(z) - X(-z)\big)\ .$$

This leads to the relation

$$H_0(z) = H_{00}(z^2) + zH_{01}(z^2) \ .$$

The relation for $H_1$ is found analogously, and then the relations for $G_0$ and $G_1$ can be found using the perfect reconstruction conditions (54) and (55) in the two cases. The results are summarized here.

$$H_0(z) = H_{00}(z^2) + zH_{01}(z^2) \ , \tag{57}$$

$$H_1(z) = H_{10}(z^2) + zH_{11}(z^2) \ , \tag{58}$$

$$G_0(z) = G_{00}(z^2) + z^{-1}G_{01}(z^2) \ , \tag{59}$$

$$G_1(z) = G_{10}(z^2) + z^{-1}G_{11}(z^2) \ . \tag{60}$$

Note the difference in the decomposition of the $H$-filters and the $G$-filters. Thus in the polyphase representation we use

$$\mathbf{H}(z) = \begin{bmatrix} H_{00}(z) & H_{01}(z) \\ H_{10}(z) & H_{11}(z) \end{bmatrix} ,$$

$$\mathbf{G}(z) = \mathbf{H}(z)^{-1} = \begin{bmatrix} G_{00}(z) & G_{10}(z) \\ G_{01}(z) & G_{11}(z) \end{bmatrix} .$$

Note the placement of entries in $\mathbf{G}(z)$, which differs from the usual notation for matrices. The requirement of perfect reconstruction in the polyphase formulation was the requirement that $\mathbf{G}(z)$ should be the inverse of $\mathbf{H}(z)$.

**From Filters to Lifting Steps**

It is easy to start with a filter bank and derive $\mathbf{H}(z)$ using (57) and (58). However, going the other way is somewhat more tricky; given $\mathbf{H}(z)$ it is necessary to factorize it into a number of $2 \times 2$ matrices of a particular structure to obtain the lifting steps. The remarkable result is that this is always possible. This result was obtained by I. Daubechies and W. Sweldens in 1998 in the paper [2]. The factorization result for $2 \times 2$ matrices with Laurent entries was previously known in the mathematical literature. The importance of the paper by I. Daubechies and W. Sweldens lies in its impact on signal processing.

**Theorem VI.1** (Daubechies and Sweldens 1998)**.** *Assume that $\mathbf{H}(z)$ is a $2 \times 2$ matrix of Laurent polynomials, normalized to $\det \mathbf{H}(z) = 1$. Then there exists a constant $K \neq 0$ and Laurent polynomials $S_1(z), \ldots, S_N(z)$, $T_1(z), \ldots, T_N(z)$, such that*

$$\mathbf{H}(z) = \begin{bmatrix} K & 0 \\ 0 & K^{-1} \end{bmatrix} \begin{bmatrix} 1 & S_N(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ T_N(z) & 1 \end{bmatrix} \cdots \begin{bmatrix} 1 & S_1(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ T_1(z) & 1 \end{bmatrix} . \tag{61}$$

The normalization $\det \mathbf{H}(z) = 1$ in the theorem can always be satisfied. As seen above, in the general case $\det \mathbf{H}(z) = Cz^{2k+1}$. One can always get the determinant equal to one by scaling and an odd shift in time. Therefore the result is stated with $\det \mathbf{H}(z) = 1$ without loss of generality.

The proof of this theorem is constructive. It gives an algorithm for finding the Laurent polynomials $S_1(z), \ldots, S_N(z)$, $T_1(z), \ldots, T_N(z)$ in the factorization. It is important to note that the factorization is not unique. Once we have a factorization, we can translate it into lifting steps.

The advantage of the lifting approach, compared to the filter approach, is that it is very easy to find perfect reconstruction filters $H_0$, $H_1$, $G_0$, and $G_1$. It is just a matter of multiplying the lifting steps as in (43), and then assemble the filters according to the equations (57)–(60). There can also be algorithmic advantages in implementing a transform using lifting steps, since it may result in fewer arithmetic operations than a filter implementation.

This approach should be contrasted with the traditional signal analysis approach, where one tries to find (approximate numerical) solutions to the equations (51) and (52), using for example spectral factorization. The weakness in constructing a transform based solely on the lifting technique is that it is based entirely on considerations in the time domain. Sometimes it is desirable to design filters with certain properties in the frequency domain, and once filters have been constructed in the frequency domain, we can use the constructive proof of the theorem to derive a lifting implementation.

We should mention that the numerical stability of transforms designed using lifting can be difficult to analyze.

## VI.4 Examples of Lifting Steps in the Frequency Domain

We will now see how the Daubechies 4 transform looks in the frequency domain, that is, what frequency response we would get when passing a signal through the Daubechies 4 transform. First, we need to find $\mathbf{H}(z)$ by multiplying the lifting steps originating in the Daubechies 4 equations (13) through (17). Each equation has an equivalent lifting step matrix (41) or (42). The first of the Daubechies 4 equations

$$s_{j-1}^{(1)}[n] = s_j[2n] + \sqrt{3}s_j[2n+1]$$

is an update step, and it simply multiplies the odd sample with $\sqrt{3}$ and add the result to the even sample. In the $z$-transform this is achieved by multiplying with

$$\begin{bmatrix} 1 & \sqrt{3} \\ 0 & 1 \end{bmatrix},$$

which makes this matrix the first transform step. The second equation

$$d_{j-1}^{(1)}[n] = s_j[2n+1] - \tfrac{1}{4}\sqrt{3}s_{j-1}^{(1)}[n] - \tfrac{1}{4}(\sqrt{3}-2)s_{j-1}^{(1)}[n-1]$$

is a prediction step that uses the present value (index $n$) and the first previous value (index $n-1$) of the signal. This is achieved in the $z$-transform by

$$\begin{bmatrix} 1 & 0 \\ -\frac{\sqrt{3}}{4} - \frac{\sqrt{3}-2}{4} z^{-1} & 1 \end{bmatrix}.$$

The third equation converts to $z$-transform similarly. The two scaling equation (16) and (17) are joined in one scaling matrix

$$\begin{bmatrix} \frac{\sqrt{3}-1}{\sqrt{2}} & 0 \\ 0 & \frac{\sqrt{3}+1}{\sqrt{2}} \end{bmatrix}.$$

Now, the polyphase matrix $\mathbf{H}(z)$ can be written

$$\begin{aligned} \mathbf{H}(z) &= \begin{bmatrix} \frac{\sqrt{3}-1}{\sqrt{2}} & 0 \\ 0 & \frac{\sqrt{3}+1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 1 & -z \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\frac{\sqrt{3}}{4} - \frac{\sqrt{3}-2}{4} z^{-1} & 1 \end{bmatrix} \begin{bmatrix} 1 & \sqrt{3} \\ 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \frac{3-\sqrt{3}}{4\sqrt{2}} z + \frac{1+\sqrt{3}}{4\sqrt{2}} & \frac{1-\sqrt{3}}{4\sqrt{2}} z + \frac{3+\sqrt{3}}{4\sqrt{2}} \\ -\frac{3+\sqrt{3}}{4\sqrt{2}} - \frac{1-\sqrt{3}}{4\sqrt{2}} z^{-1} & \frac{1+\sqrt{3}}{4\sqrt{2}} + \frac{3-\sqrt{3}}{4\sqrt{2}} z^{-1} \end{bmatrix} \\ &= \begin{bmatrix} H_{00}(z) & H_{01}(z) \\ H_{10}(z) & H_{11}(z) \end{bmatrix} \end{aligned}$$

Thus, from (57) and (58) it follows that the low and high pass filter taps (in $z$-transform) are given by

$$\begin{aligned} H_0(z) &= H_{00}(z^2) + z H_{01}(z^2) = \frac{1+\sqrt{3}}{4\sqrt{2}} + \frac{3+\sqrt{3}}{4\sqrt{2}} z + \frac{3-\sqrt{3}}{4\sqrt{2}} z^2 + \frac{1-\sqrt{3}}{4\sqrt{2}} z^3 \,, \\ H_1(z) &= H_{10}(z^2) + z H_{11}(z^2) = -\frac{1-\sqrt{3}}{4\sqrt{2}} z^{-2} + \frac{3-\sqrt{3}}{4\sqrt{2}} z^{-1} - \frac{3+\sqrt{3}}{4\sqrt{2}} + \frac{1+\sqrt{3}}{4\sqrt{2}} z \,. \end{aligned}$$

To determine the frequency response of these transfer functions, we simply replace $z$ with $e^{j\omega}$, which gives the Fourier transform of the filter taps. We then plot $|H_0(e^{j\omega})|$ and $|H_1(e^{j\omega})|$ as function of $\omega$ to show the amplitude characteristics of the transfer functions $H_0(z)$ and $H_1(z)$ on the unit circle in the complex plane. This plot is shown in Figure 17.

# VII Making Lifting Steps from Filters

There are basically three ways of representing the building block in a DWT: 1) The transform can be represented by a pair of filters (usually low pass and high pass filters) satisfying the perfect reconstruction conditions, or it can be given as lifting steps, which are either given 2) in the time domain as a set of equations, or 3) in the frequency domain as a factored matrix of Laurent polynomials.

As an example the Daubechies 4 transform can be given in these three forms, as shown below.

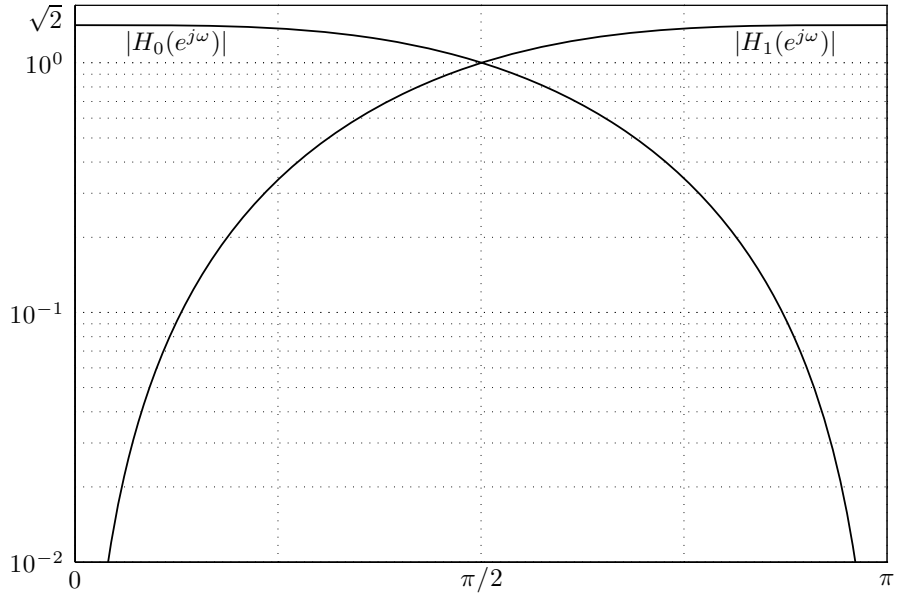



Figure 17: Frequency response of the Daubechies 4 lifting steps.

**Equation form:**

$$s^{(1)}[n] = S[2n] + \sqrt{3}S[2n+1] \,, \tag{62}$$

$$d^{(1)}[n] = S[2n+1] - \frac{1}{4}\sqrt{3}s^{(1)}[n] - \frac{1}{4}(\sqrt{3}-2)s^{(1)}[n-1] \,, \tag{63}$$

$$s^{(2)}[n] = s^{(1)}[n] - d^{(1)}[n+1] \,, \tag{64}$$

$$\tilde{s}[n] = \frac{\sqrt{3}-1}{\sqrt{2}}s^{(2)}[n] \,, \tag{65}$$

$$\tilde{d}[n] = \frac{\sqrt{3}+1}{\sqrt{2}}d^{(1)}[n] \,. \tag{66}$$

**Matrix form:**

$$\mathbf{H}(z) = \begin{bmatrix} \frac{\sqrt{3}-1}{\sqrt{2}} & 0 \\ 0 & \frac{\sqrt{3}+1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 1 & -z \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\frac{\sqrt{3}}{4} - \frac{\sqrt{3}-2}{4}z^{-1} & 1 \end{bmatrix} \begin{bmatrix} 1 & \sqrt{3} \\ 0 & 1 \end{bmatrix} . \tag{67}$$

**Filter form:**

$$\mathbf{h} = \frac{1}{4\sqrt{2}} \left[1+\sqrt{3}, \quad 3+\sqrt{3}, \quad 3-\sqrt{3}, \quad 1-\sqrt{3}\,\right] , \tag{68}$$

$$\mathbf{g} = \frac{1}{4\sqrt{2}} \left[1-\sqrt{3}, \quad -3+\sqrt{3}, \quad 3+\sqrt{3}, \quad -1-\sqrt{3}\,\right] . \tag{69}$$

The equation form was presented in (13) through (17). Converting this to the matrix form has not been shown directly. However, using (41), (42), and (43) it is not difficult to derive the matrices from the equations. It is not difficult either to obtain the filter taps $\mathbf{h}$ and $\mathbf{g}$ from the matrix form. Simply multiply the matrix to get the polyphase matrix (46), and the odd entries in $\mathbf{h}$ will be the coefficient in $H_{00}(z)$, and the even entries the coefficients of $H_{01}(z)$. Likewise, the odd and even entries of $\mathbf{g}$ is the coefficients of $H_{10}(z)$ and $H_{11}(z)$.

Now, the hard part is to go in the other direction. Suppose we have the filter taps and want to find the lifting steps (in either time or frequency domain). That it is theoretical possible to obtain the lifting steps was made evident with Theorem VI.1. Now we want to show how to actually obtain the lifting steps.

## VII.1 The Euclidean Algorithm

The Euclidean algorithm is usually first presented as an algorithm for finding the greatest common divisor of two integers. But it can be applied to many other analogous problems. One application is to finding the greatest common divisor of two Laurent polynomials. This turns out to be the key step in factorizing a polyphase matrix.

Take two Laurent polynomials $a(z)$ and $b(z) \neq 0$ with $|a(z)| \geq |b(z)|$. Then there always exist a Laurent polynomial $q(z)$, the quotient, with $|q(z)| = |a(z)| - |b(z)|$, and a Laurent polynomial $r(z)$, the remainder, with $|r(z)| < |b(z)|$, such that

$$a(z) = b(z)q(z) + r(z) \; . \tag{70}$$

We use the notation $q(z) = a(z)/b(z)$ as 'division disregarding the remainder'. Now, let $a_0(z) = a(z)$ and $b_0(z) = b(z)$, and iterate the following steps starting from $n = 0$

$$\begin{aligned} a_{n+1}(z) &= b_n(z) \; , \\ b_{n+1}(z) &= a_n(z) - q_{n+1}(z) b_n(z) \end{aligned}$$

which in matrix form is

$$\begin{bmatrix} a_{n+1}(z) \\ b_{n+1}(z) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & -q_{n+1}(z) \end{bmatrix} \begin{bmatrix} a_n(z) \\ b_n(z) \end{bmatrix} ,$$

where $q_{n+1}(z) = a_n(z)/b_n(z)$. Let $N$ denote the smallest integer with $N \leq |b(z)| + 1$, for which $b_N(z) = 0$. Then $a_N(z)$ is a greatest common divisor for $a(z)$ and $b(z)$. In short, for two Laurent

polynomials $a(z)$ and $b(z)$ with the given constraints we have

$$\begin{bmatrix} a_N(z) \\ 0 \end{bmatrix} = \prod_{n=N}^{1} \begin{bmatrix} 0 & 1 \\ 1 & -q_n(z) \end{bmatrix} \begin{bmatrix} a(z) \\ b(z) \end{bmatrix}.$$

We note that there is no unique greatest common divisor for $a(z)$ and $b(z)$, since if $d(z)$ divides both $a(z)$ and $b(z)$, and is of maximal degree, then $\alpha z^m d(z)$ is also a divisor of the same degree.

Note the order of the terms in this product, as given by the limits in the product. Inverting the matrices and moving them to the other side, we get (products in reverse order, as shown by the limits)

$$\begin{bmatrix} a(z) \\ b(z) \end{bmatrix} = \prod_{n=1}^{N} \begin{bmatrix} q_n(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} a_N(z) \\ 0 \end{bmatrix}. \tag{71}$$

Now, take $a(z) = H_{00}(z)$ and $b(z) = H_{01}(z)$ from the polyphase matrix (46). We then get

$$\begin{bmatrix} H_{00}(z) \\ H_{01}(z) \end{bmatrix} = \prod_{n=1}^{N} \begin{bmatrix} q_n(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} K \\ 0 \end{bmatrix}. \tag{72}$$

The greatest common divisor $a_N(z)$ of $H_{00}(z)$ and $H_{01}(z)$ is a constant. This is because of the determinant assumption

$$H_{00}(z)H_{11}(z) - H_{01}(z)H_{10}(z) = 1\ ,$$

since $a_N(z)$ divides the left hand side, it also divides the right hand side. Thus, $a_N(z) = Kz^\ell$. By shifting the indices in the $z$-transform of $H_0(z)$ with $\ell$ steps, the greatest common divisor becomes a constant.

Observing that

$$\begin{bmatrix} q(z) & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & q(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ q(z) & 1 \end{bmatrix},$$

and replacing

$$\begin{bmatrix} K \\ 0 \end{bmatrix} \quad \text{with} \quad \begin{bmatrix} K & 0 \\ 0 & K^{-1} \end{bmatrix},$$

and letting $M = \lceil N/2 \rceil$, we can rewrite (72) to

$$\begin{bmatrix} H_{00}(z) & H'_{10}(z) \\ H_{01}(z) & H'_{11}(z) \end{bmatrix} = \prod_{n=1}^{M} \begin{bmatrix} 1 & q_{2n-1}(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ q_{2n}(z) & 1 \end{bmatrix} \begin{bmatrix} K & 0 \\ 0 & K^{-1} \end{bmatrix},$$

where these equations *define* $H'_{10}(z)$ and $H'_{11}(z)$. If $N$ is odd, let $q_{2M}(z) = 0$. Transposing both sides yields

$$\begin{bmatrix} H_{00}(z) & H_{01}(z) \\ H'_{10}(z) & H'_{11}(z) \end{bmatrix} = \begin{bmatrix} K & 0 \\ 0 & K^{-1} \end{bmatrix} \prod_{n=M}^{1} \begin{bmatrix} 1 & q_{2n}(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ q_{2n-1}(z) & 1 \end{bmatrix}. \tag{73}$$

which is clearly resembles the factorization in Theorem VI.1, which we are trying to achieve. All we need to do now is to find out how $H'_{10}(z)$ and $H'_{11}(z)$ are connected with $H_{10}(z)$ and $H_{11}(z)$. By using the fact that both polyphase matrices (46) and (73) must have determinant 1, it is possible to show that

$$\begin{bmatrix} 1 & 0 \\ -t(z) & 1 \end{bmatrix} \begin{bmatrix} H_{00}(z) & H_{01}(z) \\ H'_{10}(z) & H'_{11}(z) \end{bmatrix} = \begin{bmatrix} H_{00}(z) & H_{01}(z) \\ H_{10}(z) & H_{11}(z) \end{bmatrix}$$

for

$$t(z) = H'_{10}(z)H_{11}(z) - H'_{11}(z)H_{10}(z) \; . \tag{74}$$

Consequently, we have the relation

$$\begin{bmatrix} H_{00}(z) & H_{01}(z) \\ H_{10}(z) & H_{11}(z) \end{bmatrix} = \begin{bmatrix} K & 0 \\ 0 & K^{-1} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -K^2 t(z) & 1 \end{bmatrix} \prod_{n=M}^{1} \begin{bmatrix} 1 & q_{2n}(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ q_{2n-1}(z) & 1 \end{bmatrix}, \tag{75}$$

which by a suitable re-indexing of the $q$ polynomials (and at the same time making $K^2 t(z)$ one of them), determines the $S_n(z)$ and $T_n(z)$ in Theorem VI.1.

## VII.2 Practical Implementation of the Euclidean Algorithm

We now have a step-by-step method for obtaining the lifting steps from any filter. It goes as follows.

1. Find the $z$-transform $H_0(z)$ of the low pass filter taps $\mathbf{h}$.
2. Determine $H_{00}(z)$ and $H_{01}(z)$ by means of (57).
3. Assign $a_0(z) = H_{00}(z)$ and $b_0(z) = H_{01}(z)$, and let $n = 0$.
4. Determine the quotient $q_{n+1}(z) = a_n(z)/b_n(z)$. There may be multiple solutions to this step. Choose one.
5. Determine the remainder as $b_{n+1} = a_n(z) - q_{n+1}(z)b_n(z)$.
6. Assign $a_{n+1}(z) = b_n(z)$.
7. Increment n. If $b_n \neq 0$, go to step 3.
8. Let $K = a_n(z)$. Determine $H'_{10}(z)$ and $H'_{11}(z)$ using (73).
9. Find the $z$-transform $H_1(z)$ of $\mathbf{g}$ such that it is index-shifted an odd number compared to $H_0(z)$.
10. Determine $H_{10}(z)$ and $H_{11}(z)$ by means of (58).
11. Determine $t(z)$ from (74).
12. Let $T_n(z) = q_{2n-1}(z)$ for $n = 1, \ldots, M$.

13. Let $S_n(z) = q_{2n}(z)$ for $n = 1, \ldots, M$.

14. Let $T_{M+1}(z) = -K^2 t(z)$.

15. If $K$ has non-zero power of $z$, you may discard this. Alternatively, choosing the right $z$-transform in step 2 will give a 'zero power' $K$.

This procedure will provide all the ingredients for building the lifting step matrices. Note that this will always give a prediction step as the first matrix, and as such this procedure seems unable to generate lifting steps for, say, Daubechies 4 which starts with an update step. However, in the polyphase representation prediction and update steps are merely linear combinations of even and odd samples applied to odd and even samples, and as such predictions and updates can be interchanged simply by an odd shift of the $z$-transform of the signal.

### VII.3 Factoring Daubechies 4 into Lifting Steps

We now give an example of creating lifting steps using the algorithm presented in the previous section. The example is factorization of Daubechies 4. This means that we show how to get the matrix form (67) from the filter tap form (68) and (69). We will follow the step-by-step method laid out in the previous section.

The Daubechies 4 filter taps are given by

$$\mathbf{h} = \begin{bmatrix} \frac{1+\sqrt{3}}{4\sqrt{2}} & \frac{3+\sqrt{3}}{4\sqrt{2}} & \frac{3-\sqrt{3}}{4\sqrt{2}} & \frac{1-\sqrt{3}}{4\sqrt{2}} \end{bmatrix} . \tag{76}$$

In the $z$-transform in step 1 we are free to choose a shift. For instance, we could choose

$$H_0(z) = h[0] + h[1]z + h[2]z^2 + h[3]z^3 . \tag{77}$$

We know that eventually this choice will provide a monomial $K$ that may or may not have zero power. As we will see later, this choice does in fact lead to a real number $K$. $H_0(z)$ must be separated into even and odd indices. This happens in step 2. Combining this with assignment to $a_0(z)$ and $b_0(z)$ in step 3 we get

$$a_0(z) = H_{00}(z) = h[0] + h[2]z = \frac{1+\sqrt{3}}{4\sqrt{2}} + \frac{3-\sqrt{3}}{4\sqrt{2}} z , \tag{78}$$

$$b_0(z) = H_{01}(z) = h[1] + h[3]z = \frac{3+\sqrt{3}}{4\sqrt{2}} + \frac{1-\sqrt{3}}{4\sqrt{2}} z . \tag{79}$$

Note that the concept of even and odd applies to the chosen $z$-transform of the filters taps (77), not the filter tap vector itself (76). Then step 4 is to find $q_1(z)$. Since $a_0(z)$ and $b_0(z)$ have the same degree, the quotient is a monomial. Matching the $z$ coefficients yields

$$q_1(z) = \frac{z \text{ coefficient of } a_0(z)}{z \text{ coefficient of } b_0(z)} = \frac{\frac{3-\sqrt{3}}{4\sqrt{2}}}{\frac{1-\sqrt{3}}{4\sqrt{2}}} = \frac{3-\sqrt{3}}{1-\sqrt{3}} = \frac{(3-\sqrt{3})(1+\sqrt{3})}{(1-\sqrt{3})(1+\sqrt{3})} = \frac{2\sqrt{3}}{-2} = -\sqrt{3} .$$

The remainder is then

$$\begin{aligned} b_1(z) &= a_0(z) - b_0(z)q_1(z) \\ &= \Big(\frac{1+\sqrt{3}}{4\sqrt{2}} + \frac{3-\sqrt{3}}{4\sqrt{2}}z\Big) - \Big(\frac{3+\sqrt{3}}{4\sqrt{2}} + \frac{1-\sqrt{3}}{4\sqrt{2}}z\Big)\cdot(-\sqrt{3}) \\ &= \frac{1+\sqrt{3}+3\sqrt{3}+3}{4\sqrt{2}} \\ &= \frac{1+\sqrt{3}}{\sqrt{2}}\,, \end{aligned}$$

thus completing step 5. In step 6 we assign

$$a_1(z) = b_0(z) = \frac{3+\sqrt{3}}{4\sqrt{2}} + \frac{1-\sqrt{3}}{4\sqrt{2}}z\,.$$

In step 7 we increment $n = 0$ to $n = 1$, and since $b_1 \neq 0$ we repeat from step 4. This time the quotient has degree 1, since $b_1(z)$ is one degree less than $a_1(z)$. More specifically, $q_2(z)$ must be on the form $c + dz$. Further, since the degree of the remainder decreases, and $|b_1(z)| = 0$, the remainder must be zero this time. So

$$b_1(z)q_2(z) = \frac{1+\sqrt{3}}{\sqrt{2}}(c+dz) = a_1(z) = \frac{3+\sqrt{3}}{4\sqrt{2}} + \frac{1-\sqrt{3}}{4\sqrt{2}}z\,.$$

Thus,

$$\begin{aligned} c &= \frac{\frac{3+\sqrt{3}}{4\sqrt{2}}}{\frac{1+\sqrt{3}}{\sqrt{2}}} = \frac{3+\sqrt{3}}{4(1+\sqrt{3})} = \frac{\sqrt{3}}{4} \\ d &= \frac{\frac{1-\sqrt{3}}{4\sqrt{2}}}{\frac{1+\sqrt{3}}{\sqrt{2}}} = \frac{1-\sqrt{3}}{4(1+\sqrt{3})} = \frac{\sqrt{3}-2}{4}\,. \end{aligned}$$

Therefore,

$$\begin{aligned} q_2(z) &= \frac{\sqrt{3}}{4} + \frac{\sqrt{3}-2}{4}z\,, \\ b_2(z) &= 0\,, \\ a_2(z) &= b_1(z) = \frac{1+\sqrt{3}}{\sqrt{2}}\,. \end{aligned}$$

Now, this time in step 7 we have $b_2(z) = 0$. Thus, we continue to step 8, assign

$$K = \frac{1+\sqrt{3}}{\sqrt{2}}\,,$$

and use (73) to find $H'_{10}$ and $H'_{11}$.

$$\begin{aligned}\begin{bmatrix} H_{00}(z) & H_{01}(z) \\ H'_{10}(z) & H'_{11}(z) \end{bmatrix} &= \begin{bmatrix} K & 0 \\ 0 & K^{-1} \end{bmatrix} \begin{bmatrix} 1 & q_2(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ q_1(z) & 1 \end{bmatrix} \\ &= \begin{bmatrix} \frac{\sqrt{3}+1}{\sqrt{2}} & 0 \\ 0 & \frac{\sqrt{3}-1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 1 & \frac{\sqrt{3}-2}{4}z + \frac{\sqrt{3}}{4} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\sqrt{3} & 1 \end{bmatrix} \\ &= \begin{bmatrix} \frac{1+\sqrt{3}}{4\sqrt{2}} + \frac{3-\sqrt{3}}{4\sqrt{2}}z & \frac{3+\sqrt{3}}{4\sqrt{2}} + \frac{1-\sqrt{3}}{4\sqrt{2}}z \\ \frac{3-\sqrt{3}}{\sqrt{2}} & \frac{\sqrt{3}-1}{\sqrt{2}} \end{bmatrix}. \end{aligned} \tag{80}$$

Now that we have $H'_{10}$ and $H'_{11}$, we need $H_{10}$ and $H_{11}$ in order to determine $t(z)$. First, we need the $z$-transform of the high pass filter taps. The high pass filter $\mathbf{g}$ is the time reversed of $\mathbf{g}$ with alternating signs. From (76) we therefore get

$$\mathbf{g} = \mathbf{H}_1 = \begin{bmatrix} -\frac{1-\sqrt{3}}{4\sqrt{2}} & \frac{3-\sqrt{3}}{4\sqrt{2}} & -\frac{3+\sqrt{3}}{4\sqrt{2}} & \frac{1+\sqrt{3}}{4\sqrt{2}} \end{bmatrix}. \tag{81}$$

The $z$-transform of $\mathbf{g}$ must have the index shifted by 1 (or some other odd number) compared to $H_0(z)$. For instance,

$$H_1(z) = g[0]z^{-2} + g[1]z^{-1} + g[2] + g[3]z\ , \tag{82}$$

where $g[3] = h[0]$ now is coefficient for an odd power, whereas $h[0]$ is coefficient for an even power in $H_0(z)$. Splitting this in even and odd coefficients (where even and odd again relates to the $z$-transform), we get from (58) that

$$\begin{aligned} H_{10}(z) &= g[0]z^{-1} + g[2] = -\frac{1-\sqrt{3}}{4\sqrt{2}}z^{-1} - \frac{3+\sqrt{3}}{4\sqrt{2}}\ , \\ H_{11}(z) &= g[1]z^{-1} + g[3] = \frac{3-\sqrt{3}}{4\sqrt{2}}z^{-1} + \frac{1+\sqrt{3}}{4\sqrt{2}}\ , \end{aligned}$$

thus completing step 10. We now insert these $H_{10}$ and $H_{11}$ together with $H'_{10}$ and $H'_{11}$ from (80) into (74).

$$\begin{aligned} t(z) &= H'_{10}(z)H_{11}(z) - H'_{11}(z)H_{10}(z) \\ &= \frac{\sqrt{3}-3}{\sqrt{2}}\Big(\frac{3-\sqrt{3}}{4\sqrt{2}}z^{-1} + \frac{1+\sqrt{3}}{4\sqrt{2}}\Big) - \frac{\sqrt{3}-1}{\sqrt{2}}\Big(-\frac{1-\sqrt{3}}{4\sqrt{2}}z^{-1} - \frac{3+\sqrt{3}}{4\sqrt{2}}\Big) \\ &= \frac{1}{8}\Big((\sqrt{3}-3)(3-\sqrt{3})z^{-1} + (\sqrt{3}-3)(1+\sqrt{3}) \\ &\qquad\qquad + (\sqrt{3}-1)(1-\sqrt{3})z^{-1} + (\sqrt{3}-1)(3+\sqrt{3})\Big) \\ &= \frac{z^{-1}}{8}\big((-9+6\sqrt{3}-3) + (-1+2\sqrt{3}-3)\big) \\ &= (\sqrt{3}-2)z^{-1}\ , \end{aligned}$$

which completes step 11. We can now determine the extra matrix we need, as shown in (75),

$$-K^2 t(z) = -\Big(\frac{1+\sqrt{3}}{\sqrt{2}}\Big)^2(\sqrt{3}-2)z^{-1} = z^{-1} \ .$$

It is now possible to determine all the matrix lifting steps that makes up the polyphase matrix, which is step 12, 13, and 14. We have $M = 1$, so we get one $S$ matrix and two $T$ matrices.

$$\begin{aligned} T_1(z) &= q_1(z) = -\sqrt{3} \\ S_1(z) &= q_2(z) = \frac{\sqrt{3}}{4} + \frac{\sqrt{3}-2}{4}z \\ T_2(z) &= -K^2 t(z) = z^{-1} \ . \end{aligned}$$

Since $K$ is a real number, we do not need to shift the powers of $z$ (as mentioned in step 15), and we can now write the complete factorization of the polyphase matrix $\mathbf{H}(z)$ as shown in (61) from Theorem VI.1.

$$\mathbf{H}(z) = \begin{bmatrix} \frac{\sqrt{3}+1}{\sqrt{2}} & 0 \\ 0 & \frac{\sqrt{3}-1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ z^{-1} & 1 \end{bmatrix} \begin{bmatrix} 1 & \frac{\sqrt{3}}{4} + \frac{\sqrt{3}-2}{4}z \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\sqrt{3} & 1 \end{bmatrix}.$$

But, this is fact not (67). The reason is that the presented algorithm for finding the lifting steps always will start with a prediction step, whereas the factorization in (67) starts with an update step. So while $\mathbf{H}(z)$ in (67) indeed is equal to $\mathbf{H}(z)$ above, the factorizations are not the same. This is a consequence of the fact that the distinction of the lifting matrices into update and prediction is useful for interpretation of the transform results, but not necessary for the implementation of the transform.

Actually, this is not the only 'non-uniqueness feature' of the wavelet transform. If instead of (77) and (82) we had chosen

$$\begin{aligned} H_0(z) &= h[0]z^{-3} + h[1]z^{-2} + h[2]z^{-1} + h[3] \ , \\ H_1(z) &= g[0]z^{1} + g[1]z^{2} + g[2]z^{3} + g[3]z^{4} \ , \end{aligned}$$

the even coefficients from before are now odd, and vice versa. That means that the relation between $\mathbf{h}$ and $\mathbf{g}$ changes signs, so for instance, $g[3] = -h[0]$ now, rather than $g[3] = h[0]$. The resulting factorization becomes

$$\mathbf{H}(z) = \begin{bmatrix} -\frac{\sqrt{3}-1}{\sqrt{2}} & 0 \\ 0 & -\frac{\sqrt{3}+1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ z^{2} & 1 \end{bmatrix} \begin{bmatrix} 1 & -\frac{\sqrt{3}}{4}z^{-1} - \frac{\sqrt{3}+2}{4}z^{-2} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \sqrt{3}z & 1 \end{bmatrix}.$$

There are more examples of factorizations in Jensen and la Cour-Harbo [3] and in the original paper by Sweldens and Daubechies [2].

# Bibliography


[1] I. Daubechies, *Ten Lectures on Wavelets*, CBSM-NSF Regional Conference Series in Applied Mathematics, vol. 60, SIAM, Philadelphia, Pa., 1992.

[2] I. Daubechies and W. Sweldens, *Factoring wavelet transforms into lifting steps*, J. Fourier Anal. Appl. **4** (1998), no. 3, 245 – 267.

[3] A. Jensen and A. la Cour-Harbo, *Ripples in Mathematics - The Discrete Wavelet Transform*, Springer, Heidelberg Berlin, June 2001.

[4] S. Mallat, *A Wavelet Tour of Signal Processing*, Academic Press Inc., San Diego, CA, 1998.

[5] C. Mulcahy, *Plotting and scheming with wavelets*, Mathematics Magazine **69** (1996), no. 5, 323–343.

[6] A. V. Oppenheim and R. Schafer, *Digital Signal Processing*, Prentice Hall Inc., Upper Saddle River, NJ, 1975.

[7] A. V. Oppenheim, R. Schafer, and J. R. Buck, *Distrete-Time Signal Processing*, Prentice Hall Inc., Upper Saddle River, NJ, 1999.

[8] W. Sweldens, *The lifting scheme: A custom-design construction of biorthogonal wavelets*, Appl. Comput. Harmon. Anal. **3** (1996), no. 2, 186–200.

[9] ———, *The lifting scheme: A construction of second generation wavelets*, SIAM J. Math. Anal. **29** (1997), no. 2, 511–546.

[10] M. Vetterli and J. Kovačević, *Wavelets and Subband Coding*, Prentice-Hall, 1995.

[11] ———, *Wavelets and Subband Coding*, second ed., Available via Open Access at `http://waveletsandsubbandcoding.org`, 2007.