Wavelets and Filter Banks

Gilbert Strang
Massachusetts Institute of Technology

Truong Nguyen
University of Wisconsin

Wellesley-Cambridge Press
Box 812060
Wellesley MA 02181

Chapter 1

Introduction

1.1 Overview and Notation

We begin with an overview of ***filters***, ***filter banks***, and ***wavelets***. We want to indicate, first in rough outline and then in detail, the connections between these three topics. Our immediate purpose is to open up the problem and the language — starting with the filter coefficients $h(n)$. The choice of those coefficients is the crucial decision. Their properties govern all that follows.

Each step is a natural development from the one before:

(1) A ***filter*** is a linear time-invariant operator. It acts on input vectors x. The output vector y is the convolution of x with a fixed vector h. The vector h contains the filter coefficients $h(0), h(1), h(2), \ldots$. Our filters are digital, not analog, so the coefficients $h(n)$ come at discrete times $t = nT$. The sampling period T is assumed to be 1 here. The inputs $x(n)$ and outputs $y(n)$ come at all times $t = 0, \pm 1, \pm 2, \ldots$:

$$y(n) \quad = \quad \sum_k h(k)\,x(n-k) \quad = \quad \text{convolution } h * x \text{ in the time domain.}$$

One input $x = (\ldots, 0, 1, 0, \ldots)$ has special importance — a unit impulse at time zero. The input has $x(n-k) = 0$ except when $n = k$. The sum in the convolution has only one term, and that term is $h(n)$. This output $y(n) = h(n)$ is the response at time n to the unit impulse $x(0) = 1$. It is the *impulse response* $h(0),\ h(1),\ \ldots,\ h(N)$.

In a moment the same filter will be described in the frequency domain. Convolution with the vector h will become *multiplication* by a function H. It is the simplicity of multiplication that makes this subject a success. The action of a filter in time and frequency is the foundation on which signal processing is built.

(2) A ***filter bank*** is a set of filters. The analysis bank often has two filters, lowpass and highpass. They separate the input signal into frequency bands. Those subsignals can be compressed much more efficiently than the original signal. Then they can be transmitted or stored. We are describing "subband coding" and its applications. At any time the signals can be recombined (by the *synthesis bank*).

It is not necessary to preserve the full outputs from the analysis filters. Normally they are *downsampled*. ***We keep only the even components of the lowpass and highpass filter outputs.***

If there are M filters, then keeping every Mth component of each output gives a total of the same length as the input. Critical sampling is the key to subband coding.

This book explains how two or more filters, with downsampling, can jointly achieve properties that are impossible for a single filter. We are particularly interested in "perfect reconstruction FIR filter banks". In this case the reconstructed output $\widehat{x}(n)$ from the synthesis bank is identical to the original input x to the analysis bank (with only a time delay). In matrix language, a banded matrix (for the analysis bank) has a banded inverse (the synthesis bank).

In the frequency domain, each filter leads to a multiplication. But downsampling is *not a time-invariant operation*. If we delay all components of y by one time unit, the output from downsampling is totally different. The new samples $y(-1), y(1), y(3)$ are entirely separate and independent from the original samples $y(0), y(2), y(4)$. Those two subsampled signals are two "phases" of y, not connected. Therefore downsampling alters the multiplication picture in the frequency domain. In fact it introduces *aliasing*.

Chapter 4 will show how the simplicity of multiplication can be rescued by looking at each phase separately. Each phase of y comes from filtering the phases of x (using phases of h). These separate pieces are multiplications in the frequency domain. The whole operation together, filtering followed by downsampling, becomes a matrix multiplication — by the *polyphase matrix*.

This is the foundation of filter bank theory (still to be explained in detail!). The analysis polyphase matrix H_p will reveal the correct synthesis bank for perfect reconstruction. That synthesis filter bank uses H_p^{-1}.

(3) ***Wavelets*** are basis functions $w_{jk}(t)$ in continuous time. A basis is a set of linearly independent functions that can be used to produce all admissible functions $f(t)$:

$$f(t) = \text{ combination of basis functions } = \sum_{j,k} b_{jk}\, w_{jk}(t). \tag{1.1}$$

The special feature of the wavelet basis is that all functions $w_{jk}(t)$ are constructed from a single mother wavelet $w(t)$. This wavelet is a small wave (a pulse). Normally it starts at time $t = 0$ and ends at time $t = N$.

The shifted wavelets w_{0k} start at time $t = k$ and end at time $t = k+N$. The rescaled wavelets w_{j0} start at time $t = 0$ and end at time $t = N/2^j$. Their graphs are compressed by the factor 2^j, where the graphs of w_{0k} are translated (shifted to the right) by k:

$$\textit{compressed:} \quad w_{j0} = w\left(2^j t\right) \qquad \textit{shifted:} \quad w_{0k}(t) = w(t-k).$$

A typical wavelet w_{jk} is compressed j times and shifted k times. Its formula is

$$w_{jk}(t) = w\left(2^j t - k\right).$$

The remarkable property that is achieved by many wavelets is *orthogonality*. The wavelets are orthogonal when their "inner products" are zero:

$$\int_{-\infty}^{\infty} w_{jk}(t)\, w_{JK}(t)\, dt = \text{ inner product of } w_{jk} \text{ and } w_{JK} = 0. \tag{1.2}$$

In this case the wavelets form an *orthogonal basis* for the space of admissible functions. This basis corresponds to a set of axes that meet at 90° angles — as most good axes do. Orthogonality leads to a simple formula for each coefficient b_{JK} in the expansion for $f(t)$. Multiply the expansion displayed in equation (1.1) by $w_{JK}(t)$ and integrate:

$$\int_{-\infty}^{\infty} f(t)\, w_{JK}(t)\, dt = b_{JK} \int_{-\infty}^{\infty} (w_{JK}(t))^2\, dt. \tag{1.3}$$

All other terms in the sum disappear because of orthogonality. Equation (1.2) eliminates all integrals of w_{jk} times w_{JK}, except the one term that has $j = J$ and $k = K$. That term produces $(w_{JK}(t))^2$. Then b_{JK} is the ratio of the two integrals in equation (1.3).

As we describe the connection between filter banks and wavelets, you will see that it is the "*highpass filter*" that leads to $w(t)$. The "*lowpass filter*" leads to a scaling function $\phi(t)$. In most constructions the lowpass filter comes first—***the scaling function is obtained before the wavelet***. In fact the scaling function (in continuous time) comes from infinite repetition $\boldsymbol{L\,L\,\ldots\,L}$ of the lowpass filter, with rescaling at each iteration. The wavelet follows from $\phi(t)$ by just *one* application of the highpass filter.

Multiresolution

At a given resolution of a signal or an image, the scaling functions $\phi\left(2^j t - k\right)$ are a basis for the set of signals. The level is set by j, and the time steps at that level are 2^{-j}. The new details at level j are represented by the wavelets $w\left(2^j t - k\right)$. Then the smooth signal plus the details, the ϕ's plus the w's, combine into a ***multiresolution*** of the signal at the finer level $j + 1$. Averages come from the scaling functions, details come from the wavelets:

$$\begin{array}{c} \text{signal at level } j \text{ (local averages)} \searrow \\ + \qquad\qquad\qquad\qquad \text{signal at level } j+1 \\ \text{details at level } j \text{ (local differences)} \nearrow \end{array}$$

That is multiresolution for one signal. When we apply it to all signals, we have multiresolution for *spaces* of functions:

$$\begin{array}{c} V_j = \text{scaling space at level } j \searrow \\ \oplus \qquad\qquad\qquad V_{j+1} = \text{scaling space at level } j+1 \\ W_j = \text{wavelet space at level } j \nearrow \end{array}$$

This idea of multiresolution is absolutely basic to wavelet analysis. Again, we are only introducing it. We are sending a coarse signal to the reader, not the details. You only have the input at level 1.

Thus the signal is divided into different *scales* of resolution, rather than different frequencies. The "time-scale plane" takes the place for wavelets that the "time-frequency plane" takes for filters. Multiresolution divides the frequencies into *octave bands*, from ω to 2ω, instead of uniform bands from ω to $\omega + \Delta\omega$. The compression of a graph, when $f(t)$ is replaced by $f(2t)$, means expansion of its Fourier transform from $F(\omega)$ to $\frac{1}{2}F\left(\frac{\omega}{2}\right)$. Frequencies shift upward by an octave, when time is rescaled by two. You will see how the time-frequency plane is partitioned naturally into *rectangles of constant area* (Figure 1.1).

This matching of long time with low frequency and short time with high frequency occurs in a natural way for wavelets. It is one of the attractions of a wavelet decomposition.

To the reader: We have reprinted in Appendix A an article on wavelets published in the American Scientist of May 1994. This article introduces wavelet notation through its correspondence with *musical notation*. In music, each note specifies a frequency and a position in time.

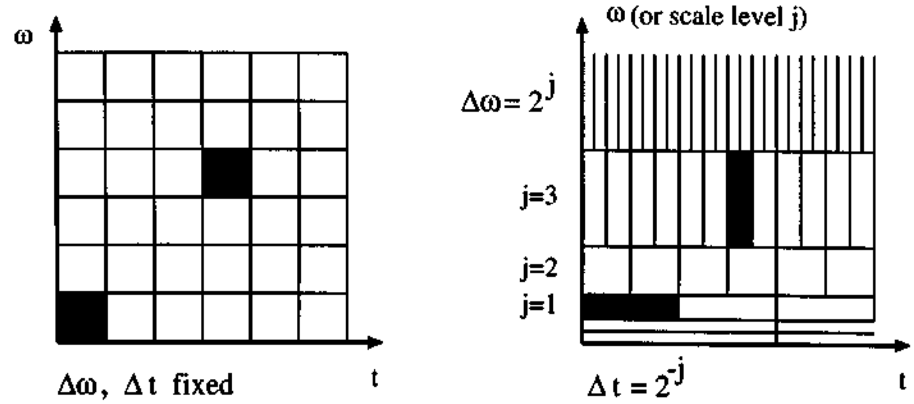


Figure 1.1: Time-frequency squares for Fourier decompositions become rectangles for wavelets. Short time intervals are natural for high frequencies.

Its vertical placement gives frequency, its horizontal placement indicates time. A musical score is almost a wavelet decomposition — except that it has fractional jumps in frequency. There are notes between middle C and high C, while wavelets jump by octaves. The shortest note I have seen is a 32nd note, corresponding to level $j = 5$ (because $2^{-5} = \frac{1}{32}$). Wavelets often stop there too, in practice. But in principle the scale level j goes to infinity.

That article on wavelets was written for a nontechnical audience, but it aims to explain the essential ideas. In the wavelet decomposition, all instruments play the same tune! They have different amplitudes and they play at different speeds and different times. The basses contribute the coarsest signals $b_{0k}w(t-k)$, starting at integer times $t = k$ and overlapping. The cellos play an identical tune but twice as fast. They contribute $b_{1k}w(2t-k)$, starting at half-integer times $t = k/2$ and again overlapping. The violas and violins add details at levels $j = 2$ and $j = 3$. Those details are wavelets $w(4t)$ and $w(8t)$ and their translates. It is the *orthogonality* of all these tunes, and especially the *localization* of each tune into a short time interval, that makes it possible to decompose the symphony efficiently.

Similarly it is orthogonality (or biorthogonality) and localization that make wavelet decompositions attractive for other signals.

Frequency Domain and Notation

To see a filter as a multiplication, we must take Fourier transforms. This will be the *discrete-time Fourier transform*, since the vectors $x(n)$ and $h(n)$ and $y(n)$ are discrete. The time index n goes from $-\infty$ to ∞. (A vector with zero components at all negative times is called *causal*.) The transform of x has two reasonable notations. They both stand for the same transform, which we denote by X:

$$X(e^{j\omega}) = \sum_{-\infty}^{\infty} x(n)\, e^{-jn\omega} \quad \text{(signal processing notation)}$$

$$X(\omega) = \sum_{-\infty}^{\infty} x(n)\, e^{-in\omega} \quad \text{(reduced notation).}$$

You see two differences. On the right side, one uses j and the other uses i. Both represent $\sqrt{-1}$. On the left side, the standard signal processing notation uses $e^{j\omega}$ while the reduced notation writes only ω. (Each has advantages and we are prepared to print the book both ways!)

The standard notation allows a direction conversion of the Fourier transform to the *z-transform*. The transform is still X but the variable becomes z:

$$X(z) = \sum_{-\infty}^{\infty} x(n)\, z^{-n}.$$

We simply replace $e^{j\omega}$ by z, extending the formal definition of X from $e^{j\omega}$ on the unit circle to z in the whole complex plane. (Remember: $e^{j\omega}$ has magnitude 1.) The Fourier transform will dominate the first part of the book, but the z-transform appears more frequently in the end.

The reduced notation has the advantage of outstanding simplicity. There will be many, many occasions to write $X(\omega)$ and $X(\omega+\pi)$ or to write $X(e^{j\omega})$ and $X(e^{j(\omega+\pi)})$. The first occasion is the most important right now, and we want to express the action of a filter both ways:

Convolution by h in time becomes multiplication by H in frequency:

$$\begin{aligned} Y(e^{j\omega}) &= H(e^{j\omega})\, X(e^{j\omega}) && \text{in signal processing notation} \\ Y(\omega) &= H(\omega)\, X(\omega) && \text{in reduced notation.} \end{aligned}$$

This is the transform of $y(n) = \sum h(k)\, x(n-k)$. Exercise 10 will ask you to verify this fundamental fact. It is the "convolution rule". In the z-domain it becomes $Y(z) = H(z)\, X(z)$. The only inputs to the proof are the definition of convolution and of the transform.

Decision on notation. Simplicity often wins. We keep the freedom to write $X(\omega)$ rather than $X(e^{j\omega})$. In reduced notation, the *frequency response* is $H(\omega)$:

$$H(\omega) = \sum h(n)\, e^{-in\omega}.$$

This is the response at frequency ω to a unit input at that frequency. When the input at *each* frequency is $X(\omega) = 1$, the output at each frequency is $H(\omega)$. Those inputs are coming from an impulse (all $X(\omega)$ are equal). ***Then the frequency response is the transform of the impulse response:***

When the input is a unit impulse $x(0) = 1$, the output is $y(n) = h(n)$.

When the input is a unit impulse $X(\omega) \equiv 1$, the output is $Y(\omega) = H(\omega)$.

Convolution by Hand

A good way to compute $y = h * x$ is to arrange it as an ordinary multiplication — but don't carry digits from one column to the next:

$$
\begin{array}{rrrrr}
 & & x(2) & x(1) & x(0) \\
 & & h(2) & h(1) & h(0) \\
\hline
 & & (2) & (1) & (0) \\
 & (3) & (2) & (1) & \\
(4) & (3) & (2) & & \\
\hline
y(4) & y(3) & y(2) & y(1) & y(0)
\end{array}
\qquad\qquad
\begin{array}{rrrrrl}
 & & 3 & 2 & 4 & = x \\
 & & 1 & 5 & 2 & = h \\
\hline
 & & 6 & 4 & 8 & \\
 & 15 & 10 & 20 & & \\
3 & 2 & 4 & & & \\
\hline
3 & 17 & 20 & 24 & 8 & = y
\end{array}
$$

The coefficients $x(n) = 4,\ 2,\ 3$ add to $X(0) = 9$. The sum of $h(n) = 2,\ 5,\ 1$ is $H(0) = 8$. Then notice that the sum for $y = h * x$ is $H(0)X(0) = 72$.

Another check is at $\omega = \pi$. The alternating sum $4 - 2 + 3$ gives $X(\pi) = 5$. Similarly $H(\pi) = -2$. Then necessarily $Y(\pi) = (5)(-2)$. This agrees with the alternating sum $3 - 17 + 20 - 24 + 8 = -10$.

Problem Set 1.1

1. Suppose the only nonzero components of x and h are $x(0) = 1, x(1) = 3$, and $h(0) = \frac{1}{2}$, $h(1) = \frac{1}{2}$. Compute the outputs $y(n)$. Verify in the frequency domain that $Y(\omega) = H(\omega)X(\omega)$.

2. What are the components $h(n)$ for the filter to become a simple advance? For any input vector $x(n)$, the output is $y(n) = x(n+1)$. Find $H(\omega)$ for this filter and verify that $Y(\omega) = H(\omega)X(\omega)$.

3. If the input filter vector x and the vector h are both causal, explain why the output y is also causal (meaning that $y(n) = 0$ for negative n). If h is causal and if $x(n) = 0$ for all $n < 8$, what can you conclude about $y(n)$?

4. If the output vector y is causal *whenever* the input x is causal, explain why the vector h must be causal.

5. If $\delta = (\ldots, 0, 0, 1, 0, 0, \ldots)$ is the unit impulse at time zero, show that convolution with any vector v leaves that vector unchanged. Translate this statement $v * \delta = v$ into the frequency domain.

6. What are the seven components of $h * h$, if $h(0) = h(1) = h(2) = h(3) = 1$ (all other $h(n) = 0$)? Use the long multiplication format in the text.

7. The long multiplication format corresponds to thinking of h and x as $h(0) + 10h(1) + 100h(2) + \cdots$ and $x(0) + 10x(1) + 100x(2) + \cdots$. The product begins with $h(0)x(0) + 10(\_\_\_) + 100(\_\_\_)$.

   This is the convolution rule $Y(z) = H(z)X(z)$ with $z = 10$.

8. Verify the convolution rule $Y = HX$ in the important special case when $h(1) = 1$ and all other $h(n) = 0$. Thus, $h = (\ldots, 0, 0, 0, 1, 0, \ldots)$.

   1. What is $y = h * x = h * (\ldots, x(-1), x(0), x(1), \ldots)$?
   2. What is $H\left(e^{j\omega}\right)$, also written $H(\omega)$?
   3. In this special case the filter H represents a ____.
   4. Verify that $\sum y(n)\, e^{-jn\omega}$ agrees with $H\left(e^{j\omega}\right) X\left(e^{j\omega}\right)$.

9. Repeating the previous exercise k times shows that $Y = HX$ is still correct when H is a k-step delay: $h(k) = 1$ and all other $h(n) = 0$. The frequency response for this delay is $H =$____.

   An arbitrary vector h is a linear combination of delays (and also advances!). By linearity, $Y = HX$ is true in general.

10. **(Important)** A direct approach to the convolution rule $Y = HX$. What is the coefficient of z^{-n} in $\left(\sum h(k)\, z^{-k}\right)\left(\sum x(\ell)\, z^{-\ell}\right)$? Show that your answer agrees with $\sum h(k)\, x(n-k)$.

    The exponent $-n$ appears in H times X when $k + l$ equals ____.

11. The *autocorrelation* $p = h * h^T$ is the convolution of h with its time reversal (or transpose): $h^T(n) = h(-n)$. Express the kth component $p(k)$ as a sum of terms.

1.2 Lowpass Filter = Moving Average

We go forward with this introduction by studying the simplest lowpass filter. Its output at time $t = n$ is the average of the input $x(n)$ at that time and the input $x(n-1)$ at the previous time:

$$y(n) = \tfrac{1}{2}x(n) + \tfrac{1}{2}x(n-1). \tag{1.4}$$

The filter coefficients are $h(0) = \frac{1}{2}$ and $h(1) = \frac{1}{2}$. Equation (1.4) fits the standard form $\sum h(k)x(n-k)$, with only two terms $k = 0$ and $k = 1$ in the sum. This is a convolution $y = h * x$. It is a *moving average*, because the output averages the current component $x(n)$ with the previous one. Old components drop away as the average moves forward with the time.

Suppose the input is the unit impulse $x = (\ldots, 0, 0, 1, 0, 0, \ldots)$. Then there are only two nonzero components in the output. The input vector has $x(0) = 1$; all other input components are zero. The output vector has $y(0) = \frac{1}{2}$, from equation (1.4) with $n = 0$. It also has $y(1) = \frac{1}{2}$, from equation (1.4) with $n = 1$. All other outputs, or moving averages, are zero. Thus the impulse response is the vector $y = \left(\ldots, 0, 0, \frac{1}{2}, \frac{1}{2}, 0, \ldots\right)$. Its components agree with the filter coefficients $h(n)$ as they should.

We want to see this filter as a linear time-invariant operator. It is a combination of two special operators, the "identity" which yields output = input and the "delay" whose output is the input one time earlier:

$$\text{averaging filter} = \tfrac{1}{2}(\text{identity}) + \tfrac{1}{2}(\text{delay}).$$

Every linear operator acting on the signal vector x can be represented by a matrix. Since the vectors are infinite, so are the matrices. Infinitely many components in x and y mean infinitely many entries in the filter matrix H. The matrix has a special structure which you see immediately in $y = Hx$:

$$\begin{bmatrix} \cdot \\ y(-1) \\ y(0) \\ y(1) \\ \cdot \end{bmatrix} = \begin{bmatrix} \cdot & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ & \frac{1}{2} & \frac{1}{2} & & \\ & & \frac{1}{2} & \frac{1}{2} & \\ & & & \cdot & \cdot \end{bmatrix} \begin{bmatrix} \cdot \\ x(-1) \\ x(0) \\ x(1) \\ \cdot \end{bmatrix}$$

The numbers $\frac{1}{2}$ on the main diagonal come from $\frac{1}{2}$ (identity). The numbers $\frac{1}{2}$ on the subdiagonal come from $\frac{1}{2}$ (delay). Substitute the unit impulse for x, with $x(0) = 1$ as its only nonzero component. Matrix multiplication produces the impulse response y. This vector has components $\frac{1}{2}$

and $\frac{1}{2}$. The response y is the filter vector h, in the middle column of the matrix, when x is the unit impulse.

This is a first occasion to see a filter as a ***constant-diagonal matrix***. The coefficient $h(0)$ appears constantly down the main diagonal. It represents $h(0)$ times the identity matrix and it yields $h(0)\,x(n)$. The coefficient $h(1)$ appears down the first subdiagonal, to represent $h(1)$ times a delay and to yield $h(1)\,x(n-1)$. If there is a coefficient $h(2)$ down the next diagonal, it multiplies a two-step delay and yields $h(2)\,x(n-2)$. The total output $y(n)$ is the sum of these special outputs:

$$\begin{aligned} y(n) &= h(0)x(n) + h(1)x(n-1) + h(2)x(n-2) + \cdots \\ &= \sum_k h(k)\,x(n-k). \end{aligned}$$

The reader notices that $h(-1)$ is not being allowed. We are dealing with ***causal filters***; the output cannot come earlier than the input. This makes $h(n) = 0$ for negative n, and it makes the filter matrix *lower triangular*.

Our example has only a finite number (two) of nonzero filter coefficients $h(n)$. The filter has a ***finite impulse response***. It is an "FIR filter". For large n, the coefficients $h(n)$ that give distant responses to the unit impulse are all zero.

To repeat: A causal FIR filter has $h(n) = 0$ for all negative n and for large positive n. The matrix is banded and lower triangular. Only a finite number of coefficients $h(0), h(1), \ldots, h(N)$ can be nonzero. The filter has $N + 1$ "taps". The matrix has bandwidth N; all other diagonals contain zeros. We concentrate almost exclusively on causal FIR filters.

Frequency Response

To find the frequency response to the filter, we change the input vector. Instead of an impulse, which combines all frequencies, the vector x will have pure frequency ω. Its components are

$$x(n) = e^{in\omega}. \tag{1.5}$$

[illegible]

This function is always periodic: $H(\omega+2\pi) = H(\omega)$. When we add 2π to the frequency ω, we add $2\pi n$ to the angle $n\omega$. The cosine, sine and complex exponential $e^{-in\omega} = \cos n\omega - i \sin n\omega$ are not changed.

Note that the response function $H(\omega)$ involves complex numbers. We strongly prefer $e^{i\omega}$ or $e^{j\omega}$ to its rectangular form $\cos \omega + i \sin \omega$. Separate formulas for the real and imaginary parts are much more complicated than a single formula for $H(\omega)$. But a single graph cannot so easily represent this complex function. One way to do it is to plot the magnitude $|H(\omega)|$ separately from the phase angle $\phi(\omega)$, recalling that

$$H(\omega) = |H(\omega)|\, e^{i\phi(\omega)}.$$

Our example has $H(\omega) = \frac{1}{2} + \frac{1}{2}e^{-i\omega}$. We factor out $e^{-i\omega/2}$ to leave a symmetric quantity $\frac{1}{2}\left(e^{i\omega/2} + e^{-i\omega/2}\right)$. This quantity is a perfect cosine:

$$H(\omega) = \left(\cos \frac{\omega}{2}\right) e^{-i\omega/2}. \tag{1.8}$$

This displays the magnitude and also the phase:

$$|H(\omega)| = \cos \frac{\omega}{2} \quad \text{and} \quad \phi(\omega) = -\frac{\omega}{2}. \tag{1.9}$$

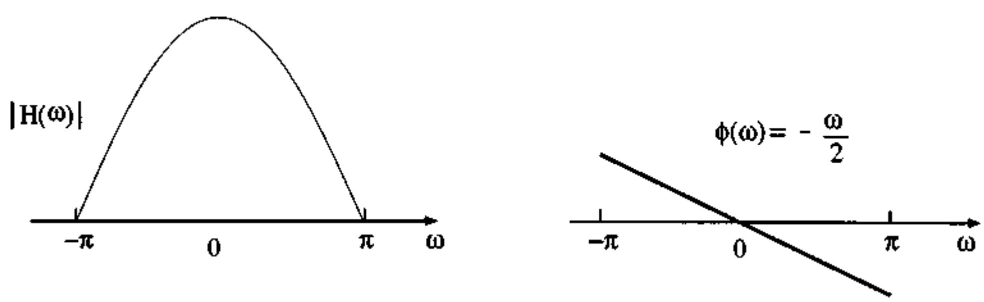


Figure 1.2: The magnitude $|H(\omega)| = \cos\left(\frac{\omega}{2}\right)$ and the phase of $H(\omega)$.

Figure 1.2 shows the plot of the magnitude $|H(\omega)|$ against the frequency ω. The cosine of $\frac{\omega}{2}$ drops to zero at $\omega = \pi$. This high frequency is wiped out, when the filter takes a moving average. In the time domain, $\omega = \pi$ corresponds to the input vector $x(n) = e^{i\pi n}$ with components $(-1)^n$. This input vector is

$$x = (\ldots, 1, -1, 1, -1, 1, \ldots).$$

The moving average of these components is constantly zero! This confirms $H(\pi) = 0$ as the correct response to the frequency $\omega = \pi$.

This is a lowpass filter. The lowest frequency $\omega = 0$, which is the DC term (direct current), is exactly preserved because $\cos 0 = 1$. The input vector $(\ldots, 1, 1, 1, \ldots)$ is equal to its moving average.

The phase $\phi(\omega)$ is the angle from the horizontal when the complex number $H(\omega)$ is plotted in the complex plane. For this particular response, those points $H(\omega) = \frac{1}{2} + \frac{1}{2}e^{-i\omega}$ lie along a circle. The constant term $\frac{1}{2}$ is the center. Then $\frac{1}{2}e^{-i\omega}$ produces a circle of radius $\frac{1}{2}$ around that center. Figure 1.3 shows this *"Nyquist diagram"*.

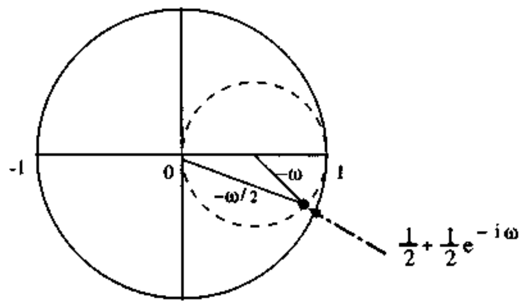


Figure 1.3: Nyquist diagram of the points $H(\omega)$ in the complex plane.

The graph of $\phi(\omega) = -\frac{\omega}{2}$ is a straight line. This is an example of "*linear phase*", an important property that some special filters possess. It reflects the fact that the filter coefficients $\frac{1}{2}$ and $\frac{1}{2}$ are symmetric. Reverse the order of coefficients and nothing changes.

Note that if the filter coefficients were symmetric *around zero*, so that $h(-1) = h(1)$ and $h(-2) = h(2)$, the frequency response would be *real*:

$$\begin{aligned} H(\omega) &= h(0) + h(1)\left(e^{i\omega} + e^{-i\omega}\right) + \cdots && \text{(symmetric coefficients)} \\ &= h(0) + h(1)\,(2\cos\omega) + \cdots && \text{(real response function)} \end{aligned}$$

In this case the phase angle is $\phi = 0$. The response has "zero phase". Similarly, coefficients that are *anti*symmetric around zero produce a pure imaginary $H(\omega)$. The phase is $\frac{\pi}{2}$ or $-\frac{\pi}{2}$. If $h(-n) = -h(n)$ then $h(0) = 0$:

$$\begin{aligned} H(\omega) &= h(1)\left(-e^{i\omega} + e^{-i\omega}\right) + h(2)\left(-e^{2i\omega} + e^{-2i\omega}\right) + \cdots && \text{(antisymmetry)} \\ &= -2i\,\left[h(1)\sin\omega + h(2)\sin 2\omega + \cdots\right] && \text{(imaginary } H(\omega)\text{)} \end{aligned}$$

Zero phase is ruled out for a causal filter. The coefficients can be symmetric or antisymmetric, but not around $n = 0$. Causal filters have *linear* phase when their coefficients are symmetric or antisymmetric around the central coefficient:

$$\textbf{Linear phase} \qquad \begin{aligned} h(k) &= h(N-k) && \text{(symmetry)} \\ h(k) &= -h(N-k) && \text{(antisymmetry)} \end{aligned}$$

Moving the center of the symmetry from 0 to $N/2$ produces a factor $e^{-iN\omega/2}$ in $H(\omega)$. This means a ***linear*** term $-N\omega/2$ in the phase. The magnitude $|H(\omega)|$ is an even (symmetric) function because $H(-\omega)$ is the complex conjugate of $H(\omega)$. The graph of $|H(\omega)|$ for $0 \le \omega \le \pi$ displays complete information.

The exercises ask you to compute $|H(\omega)|^2 = \frac{1}{2}(1 + \cos\omega)$. Then a trigonometric identity produces our special form $\cos\frac{\omega}{2}$:

$$|H(\omega)|^2 = \tfrac{1}{2}(1 + \cos\omega) = \cos^2\frac{\omega}{2} \quad \text{so that} \quad |H(\omega)| = \left|\cos\frac{\omega}{2}\right|.$$

For other filters, $|H(\omega)|^2$ is a cosine series but its square root $|H(\omega)|$ has no simple formula.

Problem Set 1.2

1. The magnitude squared is $H(\omega)$ times its complex conjugate $\overline{H(\omega)}$. Show that $|H(\omega)|^2 = \frac{1}{2}(1 + \cos\omega)$ for the moving average filter.

2. Obtain the same result from $|H(\omega)|^2 =$ (real part)^2+ (imaginary part)^2, with $H(\omega) = \frac{1}{2} + \frac{1}{2}(\cos\omega - i\sin\omega)$.

3. Why is the formula $|H(\omega)| = \cos\frac{\omega}{2}$ wrong beyond the frequency $\omega = \pi$? Draw the graph of $|H(\omega)|$ from -2π to 2π.

4. In the complex plane, draw $\tan\phi(\omega)$ as the ratio of the imaginary part $\operatorname{Im} H(\omega)$ to the real part $\operatorname{Re} H(\omega)$. Simplify this ratio to $-\tan\frac{\omega}{2}$.

   Solution. The phase $\phi(\omega)$ is the angle from the horizontal, so the tangent of $\phi(\omega)$ is the ratio of imaginary part to real part:

$$\tan\phi(\omega) = \frac{\operatorname{Im} H(\omega)}{\operatorname{Re} H(\omega)} = \frac{-\frac{1}{2}\sin\omega}{\frac{1}{2} + \frac{1}{2}\cos\omega}.$$

   Again trigonometric identities make this unusually simple:

$$\tan\phi(\omega) = \frac{-\sin\frac{\omega}{2}\cos\frac{\omega}{2}}{\cos^2\frac{\omega}{2}} = -\tan\frac{\omega}{2} \quad \text{and} \quad \phi(\omega) = -\frac{\omega}{2}.$$

5. Find the frequency response $H(\omega)$ and its magnitude and phase, for the 3-point moving average filter with $h(0) = h(1) = h(2) = \frac{1}{3}$. Does it have zero phase, constant phase, or linear phase?

6. What infinite matrix $\boldsymbol{H}$ represents the filter with $\boldsymbol{h}(0) = \boldsymbol{h}(1) = \boldsymbol{h}(2) = \frac{1}{3}$? Find two input vectors $\boldsymbol{x}(n)$ for which the output is $\boldsymbol{Hx} = \boldsymbol{0}$.

   Solution. $\boldsymbol{x} = (\ldots, 2, -1, -1, 2, -1, -1, \ldots)$ and $\boldsymbol{x}' =$ delay of $\boldsymbol{x}$.

7. What is the phase $\phi(\omega)$ of a symmetric filter with $\boldsymbol{h}(k) = \boldsymbol{h}(8-k)$?

8. What constant-diagonal matrix represents the anticausal filter $\boldsymbol{H}'$ with coefficients $\boldsymbol{h}'(0) = \frac{1}{2}$ and $\boldsymbol{h}'(-1) = \frac{1}{2}$? What matrix represents the symmetric filter $\boldsymbol{F} = \boldsymbol{H}'\boldsymbol{H}$ and what is the frequency response of $\boldsymbol{F}$?

9. Find causal filters whose magnitude responses are $|H(\omega)| = |\cos\omega|$ and $|H(\omega)| = (\cos\frac{\omega}{2})^2$. Are they unique?

10. Iterate the averaging filter four times to get $\boldsymbol{K} = \boldsymbol{H}^4$. What is $K(\omega)$ and what is the impulse response $\boldsymbol{k}(n)$?

11. Why is an antisymmetric filter, with $\boldsymbol{h}(k) = -\boldsymbol{h}(N-k)$, never lowpass?

12. Consider an averaging filter with four coefficients $\boldsymbol{h}(0) = \boldsymbol{h}(1) = \boldsymbol{h}(2) = \boldsymbol{h}(3) = \frac{1}{4}$ and the input $\boldsymbol{x}(n) = (-1)^n$. What is the output $\boldsymbol{y}(n) = \boldsymbol{x}(n) * \boldsymbol{h}(n)$? Without computing $H(\omega)$, explain why $\boldsymbol{h}(n)$ is lowpass.

13. Let $H(z)$ be a lowpass filter with $N+1$ coefficients and let $G(z) = z^{-N}H(z^{-1})$. Find $\boldsymbol{g}(n)$ in terms of $\boldsymbol{h}(n)$. What is the phase of $G(\omega)$ if H is symmetric? Is $G(z)$ lowpass or highpass?

1.3 Highpass Filter = Moving Difference

A lowpass filter takes "averages". It smoothes out the bumps in the signal. A bump is a high-frequency component, which the lowpass filter reduces or removes. The response is small or zero near the highest discrete-time frequency $\omega = \pi$.

A *highpass filter takes "differences"*. It picks out the bumps in the signal. The smooth parts are low-frequency components, which the highpass filter reduces or removes. Now the frequency response is small or zero for frequencies near $\omega = 0$ (which is direct current and has no bumps).

The lowpass filter that outputs the moving average $\frac{1}{2}(x(n) + x(n-1))$ has a twin (or mirror) filter. This is the highpass filter that computes *moving differences*:

$$\textit{Highpass:} \quad y(n) = \tfrac{1}{2}x(n) - \tfrac{1}{2}x(n-1). \tag{1.10}$$

The new filter coefficients are $\boldsymbol{h}(0) = \frac{1}{2}$ and $\boldsymbol{h}(1) = -\frac{1}{2}$. Equation (1.10) is a convolution $\boldsymbol{y} = \boldsymbol{h} * \boldsymbol{x}$, and the vector $\boldsymbol{h}$ for this highpass filter is

$$\boldsymbol{h} = \left(\ldots, 0, 0, \tfrac{1}{2}, -\tfrac{1}{2}, 0, \ldots\right). \tag{1.11}$$

This $\boldsymbol{h}$ is exactly the response to the impulse $\boldsymbol{x} = (\ldots, 0, 0, 1, 0, 0, \ldots)$. At time zero, the difference $\frac{1}{2}(x(0) - x(-1))$ is $\frac{1}{2}(1-0)$. The next difference $\frac{1}{2}(x(1) - x(0))$ is $\frac{1}{2}(0-1)$. Those numbers $\frac{1}{2}$ and $-\frac{1}{2}$ are the coefficients in $\boldsymbol{h}$.

This unit impulse at time zero we will denote by $\boldsymbol{\delta}$. In the language of convolution, we are saying that $\boldsymbol{\delta} = (\ldots, 0, 0, 1, 0, 0, \ldots)$ always yields

$$\boldsymbol{h} * \boldsymbol{\delta} = \boldsymbol{h}. \tag{1.12}$$

In the language of matrices, multiplying any matrix times the special column vector $\boldsymbol{\delta}$ always picks out the *zeroth column of the matrix*. The matrix for our highpass filter does have $\frac{1}{2}$ and $-\frac{1}{2}$ in its zeroth column:

$$\begin{bmatrix} \cdot \\ y(-1) \\ y(0) \\ y(1) \\ \cdot \end{bmatrix} = \begin{bmatrix} \cdot & \cdot & & & \\ -\frac{1}{2} & \frac{1}{2} & & & \\ & -\frac{1}{2} & \frac{1}{2} & & \\ & & -\frac{1}{2} & \frac{1}{2} & \\ & & & \cdot & \cdot \end{bmatrix} \begin{bmatrix} \cdot \\ x(-1) \\ x(0) \\ x(1) \\ \cdot \end{bmatrix}. \tag{1.13}$$

This is again a constant-diagonal matrix because the filter is again time-invariant. (We define and discuss time-invariance in the next chapter.) The main diagonal entries produce $\frac{1}{2}$(identity). The subdiagonal entries give $-\frac{1}{2}$(delay). The filter as a whole is

$$\text{highpass filter} = \tfrac{1}{2}\,(\text{identity}) - \tfrac{1}{2}\,(\text{delay}) = \tfrac{1}{2}\mathbf{I} - \tfrac{1}{2}\mathbf{S}. \tag{1.14}$$

This is another causal FIR filter with two taps, but its frequency response is completely different from the lowpass function $H_0(\omega) = \frac{1}{2}\left(1 + e^{-i\omega}\right)$.

Frequency Response

At frequency ω, the input vector is $x(n) = e^{in\omega}$. The highpass output is

$$\begin{aligned} y(n) &= \tfrac{1}{2}\, e^{in\omega} - \tfrac{1}{2}\, e^{i(n-1)\omega} \\ &= \left(\tfrac{1}{2} - \tfrac{1}{2}\, e^{-i\omega}\right) e^{in\omega} \\ &= H_1(\omega)\, e^{in\omega}. \end{aligned} \tag{1.15}$$

This quantity $H_1(\omega) = \frac{1}{2} - \frac{1}{2}\, e^{-i\omega}$ is the highpass response. We want to graph it and compare it with $H_0(\omega)$ — the lowpass response. *We are introducing the subscripts* 0 *and* 1 *for lowpass and highpass.*

As before, we take out a factor $e^{-i\omega/2}$. This leaves a sine not a cosine:

$$H_1(\omega) = \tfrac{1}{2}\left(e^{i\omega/2} - e^{-i\omega/2}\right) e^{-i\omega/2} = \sin\left(\frac{\omega}{2}\right) i\, e^{-i\omega/2}. \tag{1.16}$$

The magnitude is $|H_1(\omega)| = \left|\sin \frac{\omega}{2}\right|$. Since the sine function can be negative, we must take its absolute value.

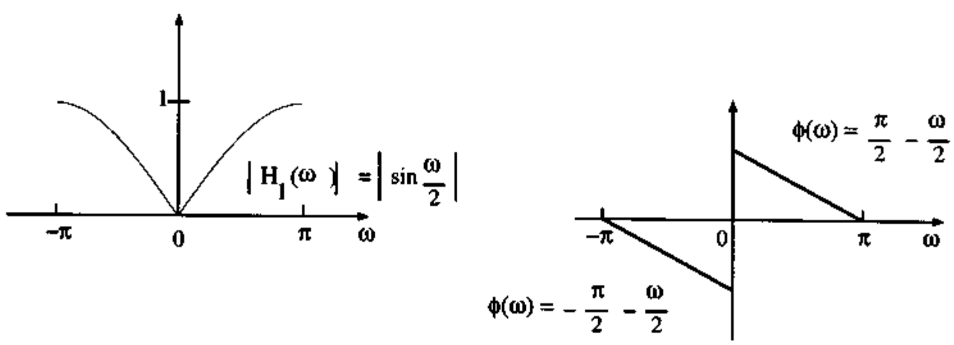


Figure 1.4: The magnitude and phase of the highpass filter $\frac{1}{2} - \frac{1}{2}\, e^{-i\omega}$.

The graph of $|H_1(\omega)|$ is in Figure 1.4a. It shows zero response to direct current, because $\sin 0 = 0$. It shows unit response at the highest frequency $\omega = \pi$, because $\sin \frac{\pi}{2} = 1$. In the time domain, these numbers come from taking differences of the lowest and highest frequency inputs:

$$x_{\text{low}} = (\ldots, 1, 1, 1, 1, 1, \ldots) \quad \text{and} \quad x_{\text{high}} = (\ldots, 1, -1, 1, -1, 1, \ldots).$$

The response to x_{low} is $0\, x_{\text{low}}$. The response to x_{high} is $1\, x_{\text{high}}$. The first vector has no bumps and the second vector is all bumps.

The phase factor in $H_1(\omega)$ is a little tricky. When the magnitude $\left|\sin \frac{\omega}{2}\right|$ is removed, we are left with a sign change at $\omega = 0$:

$$e^{i\phi(\omega)} = \begin{cases} -i\, e^{-i\omega/2} & \text{for} \quad -\pi < \omega < 0 \\ +i\, e^{-i\omega/2} & \text{for} \quad 0 < \omega < \pi. \end{cases}$$

Changing i to $e^{i\pi/2}$, we see a *discontinuity in the phase*. Figure 1.4b shows how $\phi(\omega)$ jumps from $-\frac{\pi}{2}$ to $+\frac{\pi}{2}$ at $\omega = 0$. At other points the graph is linear, so we turn a blind eye to this discontinuity and say that the filter is still *linear phase*. It is the zero at $\omega = 0$ that causes this discontinuity in phase.

Invertibility and Noninvertibility

In any reasonable sense, the averaging and differencing filters are *not invertible.* The constant signal $x = (\ldots, 1, 1, 1, 1, 1, \ldots)$ is wiped out by H_1. There cannot exist a filter H_1^{-1} that recovers this x from the zero vector. A linear operator cannot raise the dead — it only recovers $\mathbf{0}$ from $\mathbf{0}$. *If a filter H has an inverse*, the only vector in its nullspace is that zero vector:

$$\text{If } H \text{ is invertible and } Hx = \mathbf{0}, \text{ then } H^{-1}Hx = \mathbf{0} \text{ and thus } x = \mathbf{0}.$$

The frequency response of an invertible filter must have $H(\omega) \neq 0$ at all frequencies. Our filters are not invertible because $H_0(\pi) = 0$ and $H_1(0) = 0$.

The lowpass filter wipes out the alternating signal $x = (\ldots, 1, -1, 1, -1, \ldots)$. We cannot recover this x from zero output.

Note. When the inverse filter exists, it has frequency response $1/H(\omega)$. Multiplication will recover the input, because $H(\omega)/H(\omega) = 1$. We are safe as long as $H(\omega)$ is nonzero.

Suppose we attempt this inversion for the moving average. It is doomed to failure because $H_0(\pi) = 0$, but we can compute the filter coefficients that come from $1/H_0(\omega)$:

$$\frac{1}{\frac{1}{2} + \frac{1}{2}e^{-i\omega}} = 2\left(1 - e^{-i\omega} + e^{-2i\omega} - e^{-3i\omega} + \cdots\right). \tag{1.17}$$

Those coefficients $2, -2, 2, -2, \ldots$ go down the diagonals of the inverse filter matrix:

$$H^{-1}H = \begin{bmatrix} \cdot & & & & \\ -2 & 2 & & & \\ 2 & -2 & 2 & & \\ -2 & 2 & -2 & 2 & \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \begin{bmatrix} \cdot & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ & \frac{1}{2} & \frac{1}{2} & & \\ & & \frac{1}{2} & \frac{1}{2} & \\ & & & \frac{1}{2} & \frac{1}{2} \end{bmatrix} = I.$$

We seem to have found an inverse, but it is not a legal filter. *This H^{-1} is not stable.* The bounded input $y = (\ldots, 1, -1, 1, -1, 1, \ldots)$ would produce an unbounded output $H^{-1}y$. The series expansion in (1.17) breaks down completely at $\omega = \pi$, where $1 + e^{-i\pi} = 0$:

$$\frac{1}{0} = \frac{1}{\frac{1}{2} + \frac{1}{2}e^{-i\pi}} = 2(1 + 1 + 1 + 1 + \ldots).$$

But also note: If the input signal contains no frequencies near $\omega = \pi$, then we *could* reconstruct x from y. The output y would also have no high frequencies. Dividing by $H(\omega)$ would not involve dividing by zero.

The lowpass filter with coefficients $h(0) = \frac{2}{3}$ and $h(1) = \frac{1}{3}$ is safely invertible. Its frequency response $\frac{2}{3} + \frac{1}{3}e^{-i\omega}$ is never zero. The response at $\omega = 0$ is $\frac{2}{3} + \frac{1}{3} = 1$. The response at $\omega = \pi$ is $\frac{2}{3} - \frac{1}{3} = \frac{1}{3}$. The convolution with $h = (\ldots, 0, \frac{2}{3}, \frac{1}{3}, 0, \ldots)$ can be undone by *deconvolution*. To find the deconvolution coefficients, which enter the inverse filter, divide by $H(\omega)$:

$$\frac{1}{\frac{2}{3} + \frac{1}{3}e^{-i\omega}} = \frac{3}{2}\left(1 - \frac{1}{2}e^{-i\omega} + \frac{1}{4}e^{-2i\omega} - \frac{1}{8}e^{-3i\omega} + \ldots\right). \tag{1.18}$$

Those coefficients $\frac{3}{2}, -\frac{3}{4}, \frac{3}{8}, -\frac{3}{16}, \ldots$ go on the diagonals of the inverse filter matrix. It is an IIR filter, with infinitely many coefficients.

The main point is analysis in the frequency domain. Dividing by $\frac{2}{3} + \frac{1}{3} e^{-i\omega}$ inverts the filter (and the constant-diagonal infinite matrix) for deconvolution.

Final remark. The inverse filter is very rarely FIR, because $1/H(\omega)$ is not a polynomial. To invert a finite length filter with a finite length filter, we need to go to a *filter bank*. That comes next.

Problem Set 1.3

1. Can a symmetric filter, with $h(k) = h(N-k)$, be a highpass filter?
2. Draw the graph for $|\omega| \le \pi$ of "ideal" lowpass and highpass filters $H_0(\omega)$ and $H_1(\omega)$ with $H_0(\omega) + H_1(\omega) \equiv 1$. Why don't we use these filters exclusively?
3. Which of the following filters are invertible? Find the inverse filters:

$$\begin{array}{ll} (a) & h(0) = \frac{2}{3} \text{ and } h(1) = -\frac{1}{3} \\ (b) & h(0) = 2 \text{ and } h(2) = 1 \\ (c) & h(n) = \frac{1}{n!} \ (n = 0, 1, 2, \ldots) \end{array}$$

4. Invent a highpass filter K with three or four taps (coefficients) that is better than the moving difference H_1: the goal is

$$|K(\omega)| < |H_1(\omega)| \text{ for } 0 < |\omega| < \frac{\pi}{2}$$

and

$$|H_1(\omega)| < |K(\omega)| < 1 \text{ for } \frac{\pi}{2} < |\omega| < \pi.$$

5. Find all input signals $x(n)$ that are cut in half by the moving difference H_1, so that $H_1x(n) = \frac{1}{2}x(n)$. Answer in the frequency domain.
6. ***Important*** If $H_0(\omega)$ is the response of a lowpass filter, what is the response $H_1(\omega)$ of a corresponding highpass filter? If $h(0), \ldots, h(N)$ are the coefficients of H_0, what are the coefficients of your H_1?
7. Let $H(z)$ be symmetric lowpass with $2K+1$ coefficients. What is the phase of $G(z) = z^{-K} - H(z)$? Sketch the amplitude response if $H(\omega)$ is $\frac{1}{3}$-band (near 1 for $|\omega| \le \frac{\pi}{3}$). Is G lowpass or highpass?
8. A simple highpass filter design is $G(z) = H(-z)$. What is the relation between $g(n)$ and $h(n)$? With $H(\omega)$ as in Problem 7, sketch $G(\omega)$.
9. If $G(z) = H(-z^{-1})$ find $g(n)$ in terms of $h(n)$. If $H(z)$ is highpass show that $G(z)$ is lowpass.

1.4 Filter Bank = Lowpass and Highpass

Separately, the lowpass and highpass filters are not invertible. H_0 removes the highest frequency $\omega = \pi$, and H_1 removes the lowest frequency $\omega = 0$. Together, these filters do something very desirable. *They separate the signal into frequency bands.* The filtered output H_0x is weighted towards low frequencies, and H_1x is in some way the complement. The cutoff is not sharp, because these filters are so crude. But they make up the beginning of a *filter bank.*

One difficulty: the signal length has doubled. If the input x is nonzero over a time T, so are the outputs from both filters. If the first nonzero input is $x(0)$ and the last is $x(T)$, then H_0x and H_1x end at time $T+1$ (because of the final average and difference). One extra component is no problem, but doubling the storage by keeping two full-length outputs is unacceptable.

The solution is to downsample (or decimate).

Downsampling

We can keep *half* of H_0x and H_1x, and still recover x. This is an essential part of the filter bank, to *downsample* the outputs from the separate filters. We shall save only the *even-numbered components* of the two outputs. The odd-numbered components are removed:

$$(\downarrow 2)y = (\ldots, y(-4), y(-2), y(0), y(2), y(4), \ldots). \tag{1.19}$$

The symbol $\downarrow 2$ indicates downsampling or decimation. This is a linear operation (of course). But it is not time-invariant and it is not invertible. We study it closely in Chapter 3, and already this first filter bank indicates how it works in practice.

One note about normalization. To compensate for losing half the components in $(\downarrow 2)$, we multiply the surviving $y(2n)$ by $\sqrt{2}$. This normalizing factor (explained below) is usually included with the filter bank, so that

$$\begin{array}{lll} \textit{lowpass:} & H_0(\omega) \text{ changes to } & C(\omega) = \sqrt{2}\, H_0(\omega) \\ \textit{highpass:} & H_1(\omega) \text{ changes to } & D(\omega) = \sqrt{2}\, H_1(\omega). \end{array}$$

In the averaging filter, the coefficients increase to $\frac{\sqrt{2}}{2}$. We keep the notation $h(0)$, $h(1)$ for the original coefficients, and we introduce $c(0)$, $c(1)$ for the new (renormalized) coefficients of C:

$$c(0) = c(1) = \frac{\sqrt{2}}{2} \quad \text{(which is } \frac{1}{\sqrt{2}}\text{)}.$$

Similarly the highpass coefficients $\frac{1}{2}$ and $-\frac{1}{2}$ are multiplied by $\sqrt{2}$, in D:

$$d(0) = \frac{\sqrt{2}}{2} = \frac{1}{\sqrt{2}} \quad \text{and} \quad d(1) = -\frac{\sqrt{2}}{2} = -\frac{1}{\sqrt{2}}.$$

Systematically in this book, we will write C and D for $\sqrt{2}H_0$ and $\sqrt{2}H_1$. The response at frequency $\omega = 0$ is $C = \sqrt{2}$ rather than $H_0 = 1$.

Decimation Filters in the Time Domain

Downsampling follows the filter C, in operating on x. These two steps, filtering and decimation, can be done with *one matrix* L. Decimation removes the odd-numbered components. To obtain $(\downarrow 2)\,Cx$ in one step, *remove the odd-numbered rows of the filter matrix* C. The combination of filtering by C and decimation by $(\downarrow 2)$ is represented by a *rectangular matrix* L that no longer has constant diagonals. It has 1×2 blocks:

$$L = (\downarrow 2)\,C = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & & & & \\ & & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & & \\ & & & & \cdot & \cdot \end{bmatrix}.$$

The entries are $c(0)$ and $c(1)$ but half the rows of C have disappeared. Similarly the decimated highpass filter is represented by a rectangular matrix $B = (\downarrow 2)D$. Removing half the rows of D leaves the matrix B with a ***double-shift***:

$$B = (\downarrow 2)\,D = \begin{bmatrix} -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & & & & \\ & & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & & \\ & & & & \cdot & \cdot \end{bmatrix}.$$

When the lowpass L and the highpass B go into one matrix, you will see why the normalization by $\sqrt{2}$ is desirable. The rectangular L and B fit into a square matrix:

$$\begin{bmatrix} (\downarrow 2)\,C \\ (\downarrow 2)\,D \end{bmatrix} = \begin{bmatrix} L \\ B \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 & & & \\ & & 1 & 1 & \\ & & & & \cdot\;\cdot \\ -1 & 1 & & & \\ & & -1 & 1 & \\ & & & & \cdot\;\cdot \end{bmatrix}.$$

This matrix represents the whole analysis bank. It executes the lowpass channel and the highpass channel (both decimated). All rows are unit vectors (because of the division by $\sqrt{2}$). Those row vectors are mutually orthogonal. At the same time, the columns are also *orthogonal unit vectors.*

The combined square matrix is invertible. ***The inverse is the transpose***:

$$\begin{bmatrix} L \\ B \end{bmatrix}^{-1} = \begin{bmatrix} L^T & B^T \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & & -1 & \\ 1 & & 1 & \\ & 1 & & -1 \\ & 1 & & 1 \\ & \cdot & & \cdot \\ & \cdot & & \cdot \end{bmatrix}.$$

Multiplying in either order yields the identity matrix. *The second matrix* $\begin{bmatrix} L^T & B^T \end{bmatrix}$ *represents the synthesis bank.* This is an ***orthogonal filter bank***, because *inverse = transpose*. We pause to summarize what you have seen in this example.

The channels $L = (\downarrow 2)\,C$ *and* $B = (\downarrow 2)\,D$ *of an orthogonal filter bank are represented in the time domain by a combined orthogonal matrix:*

$$\begin{bmatrix} L^T & B^T \end{bmatrix} \begin{bmatrix} L \\ B \end{bmatrix} = L^T L + B^T B = I.$$

The synthesis bank is the transpose of the analysis bank. When one follows the other we have perfect reconstruction. For causality we add a delay.

That summary is a foretaste of later chapters. We will construct longer and better filters, but the underlying problem will be close to this one. There is an analysis bank and a synthesis bank. When they are transposes as well as inverses, the whole filter bank is called ***orthogonal***. When they are inverses but not necessarily transposes, the filter bank is ***biorthogonal***.

In the biorthogonal case, the coefficients in the synthesis bank are different from $c(n)$ and $d(n)$ in the analysis bank. Our ***Haar filter bank*** has orthogonal filters. This chapter pursues it further, all the way to Haar wavelets.

Block Form of a Filter Bank

The best way to represent a two-channel filter bank is by a block diagram (Figure 1.5). The input is a vector x. The blocks are linear operators and the output is two half-length vectors: $Lx = (\downarrow 2)\,Cx$ and $Bx = (\downarrow 2)\,Dx$.

For this special filter bank, we want to display all vectors in detail. Later we could supply only the essential information: the coefficients $c(n)$ and $d(n)$. There will be several ways to display those coefficients — the filter bank form, the modulation form, and the polyphase form. We

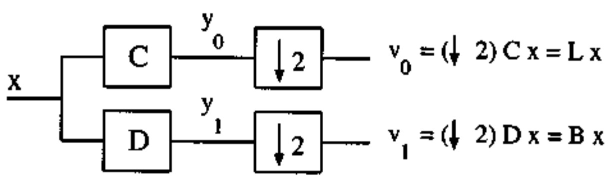


Figure 1.5: Schematic of the analysis half of a two-channel filter bank.

always need the coefficients! Here we use the direct filter bank form, and write the components of each output vector:

$$Lx = \frac{1}{\sqrt{2}}\begin{bmatrix} & \cdot & \\ x(0) & + & x(-1) \\ x(2) & + & x(1) \\ x(4) & + & x(3) \end{bmatrix} \qquad Bx = \frac{1}{\sqrt{2}}\begin{bmatrix} & \cdot & \\ x(0) & - & x(-1) \\ x(2) & - & x(1) \\ x(4) & - & x(3) \end{bmatrix}.$$

The odd-numbered components of Cx involved $x(1)+x(0)$ and $x(3)+x(2)$. Those are gone in Lx. Similarly the odd-numbered components of Dx are gone in Bx. We are left with two half-length signals. Nevertheless we have enough information to recover the full-length input x — and you see how.

To recover the zeroth component, add $x(0)+x(-1)$ to the difference $x(0)-x(-1)$. That gives $2x(0)$. Since our sums and differences are already divided by $\sqrt{2}$ in the analysis bank, we need another $\sqrt{2}$ in synthesis:

$$x(0) = \frac{1}{\sqrt{2}}\left(\frac{x(0)+x(-1)}{\sqrt{2}} + \frac{x(0)-x(-1)}{\sqrt{2}}\right). \tag{1.20}$$

For $x(-1)$, use the same components of Lx and Bx, and *subtract*:

$$x(-1) \quad = \quad \frac{1}{\sqrt{2}}\left(\frac{x(0)+x(-1)}{\sqrt{2}} - \frac{x(0)-x(-1)}{\sqrt{2}}\right).$$

Addition and subtraction are simple for this example, but the synthesis steps must be organized in a way that extends to other examples. At the end of the reconstruction, we want to reach these two vectors w_0 and w_1:

$$w_0 = \frac{1}{2}\begin{bmatrix} & \cdot & \\ x(0) & + & x(-1) \\ x(0) & + & x(-1) \\ x(2) & + & x(1) \\ x(2) & + & x(1) \\ & \cdot & \end{bmatrix} \text{ and } w_1 = \frac{1}{2}\begin{bmatrix} & \cdot & \\ -x(0) & + & x(-1) \\ x(0) & - & x(-1) \\ -x(2) & + & x(1) \\ x(2) & - & x(1) \\ & \cdot & \end{bmatrix}. \tag{1.21}$$

Notice how the signs in w_1 are adjusted so that $w_0 + w_1$ recovers the input vector. Actually we are getting $x(n-1)$ instead of $x(n)$. *The total effect of the whole filter bank is a delay.* The input is $x(n)$ and the output is the delayed $x(n-1)$. The sum $w_0 + w_1$ is almost, but not quite, the original x.

A delay is built in because all filters are *causal*. We analyzed x into low and high frequencies. Now we synthesize to reach w_0 and w_1 and recover x.

The Synthesis Bank

A well-organized synthesis bank is the inverse of the analysis bank. The analysis bank had two steps, *filtering* and *downsampling*. The synthesis bank also has two steps, ***upsampling and filtering***. Notice how the order is reversed — as it always is for inverses.

The first step is to bring back full-length vectors. The downsampling operation $(\downarrow 2)$ is not invertible, but *upsampling* is as close as we can come. The odd-numbered components are *returned as zeros by upsampling*. Applied to a half-length vector $\boldsymbol{v}$, upsampling inserts zeros:

$$\textbf{\textit{Upsampling}} \qquad (\uparrow 2)\begin{bmatrix} \cdot \\ v(0) \\ v(1) \\ v(2) \\ \cdot \end{bmatrix} = \begin{bmatrix} \cdot \\ v(0) \\ 0 \\ v(1) \\ 0 \\ v(2) \\ \cdot \end{bmatrix}. \tag{1.22}$$

Upsampling is denoted by $(\uparrow 2)$. To understand it, look at the result of downsampling to get $\boldsymbol{v} = (\downarrow 2)\boldsymbol{y}$ and upsampling to get $\boldsymbol{u} = (\uparrow 2)(\downarrow 2)\boldsymbol{y}$:

$$\boldsymbol{y} = \begin{bmatrix} \cdot \\ y(0) \\ y(1) \\ y(2) \\ y(3) \\ y(4) \\ \cdot \end{bmatrix} \quad (\downarrow 2)\boldsymbol{y} = \begin{bmatrix} \cdot \\ y(0) \\ y(2) \\ y(4) \\ \cdot \end{bmatrix} \quad (\uparrow 2)(\downarrow 2)\boldsymbol{y} = \begin{bmatrix} \cdot \\ y(0) \\ 0 \\ y(2) \\ 0 \\ y(4) \\ \cdot \end{bmatrix}. \tag{1.23}$$

The odd-numbered components of $\boldsymbol{y}$ are replaced by zeros. We will see that $(\uparrow 2)$ is the *transpose* of $(\downarrow 2)$. Fortunately, transposes come in reverse order exactly as inverses do. So synthesis can be the *transpose* of analysis — apart from our ever-present delay.

Small note: Also $(\uparrow 2)$ is a *right-inverse* of $(\downarrow 2)$. If we put in zeros and remove them, we recover $\boldsymbol{y}$. Thus $(\downarrow 2)(\uparrow 2) = \boldsymbol{I}$. The order $(\uparrow 2)(\downarrow 2)$ that we actually use is displayed above, and it inserts zeros. Properly speaking, $(\uparrow 2)$ is the "pseudoinverse" of $(\downarrow 2)$ — which has no inverse.

The vectors reached by upsampling have zeros in their odd components:

$$\mathbf{u}_0 = \frac{1}{\sqrt{2}}\begin{bmatrix} x(0) + x(-1) \\ 0 \\ x(2) + x(1) \\ 0 \\ x(4) + x(3) \\ \cdot \end{bmatrix} \quad \text{and} \quad \mathbf{u}_1 = \frac{1}{\sqrt{2}}\begin{bmatrix} x(0) - x(-1) \\ 0 \\ x(2) - x(1) \\ 0 \\ x(4) - x(3) \\ \cdot \end{bmatrix}. \tag{1.24}$$

The second step in the synthesis bank, after upsampling, is *filtering*. The two vectors $\mathbf{u}_0$ and $\mathbf{u}_1$ are the inputs to the two filters. The vectors $\boldsymbol{w}_0$ and $\boldsymbol{w}_1$ are the desired outputs. Schematically, the structure of the synthesis bank is in Figure 1.6.

Normally we would construct the synthesis filters F and G based on the analysis filters C and D. That will be our procedure in the rest of the book. For this example, when we know the

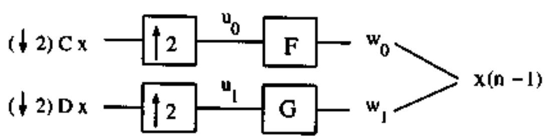


Figure 1.6: The synthesis half of a filter bank: upsample, filter, and add.

desired outputs, we proceed more directly. The filter F that produces w_0 from u_0 is an addition filter:

$$F \text{ filters } \frac{1}{\sqrt{2}}\begin{bmatrix} x(0)+x(-1) \\ 0 \\ x(2)+x(1) \\ 0 \\ \cdot \end{bmatrix} \text{ to give } \frac{1}{2}\begin{bmatrix} x(0)+x(-1) \\ x(0)+x(-1) \\ x(2)+x(1) \\ x(2)+x(1) \\ \cdot \end{bmatrix} = w_0.$$

This is the output we want. It comes from a time-invariant causal filter F. There is no separate treatment of even and odd components! When the input to F has components $u(n)$, the output has components

$$Fu(n) = \frac{1}{\sqrt{2}}(u(n) + u(n-1)). \tag{1.25}$$

The filter coefficients are $G(0) = f(1) = \frac{1}{\sqrt{2}}$.

The second synthesis filter G is a subtraction filter. Its coefficients are $\frac{-1}{\sqrt{2}}$ and $\frac{1}{\sqrt{2}}$. For an arbitrary input vector u, the output has components $\frac{1}{\sqrt{2}}(-u(n) + u(n-1))$. Notice especially how G acts on $u_1 = (\uparrow 2)(\downarrow 2)Dx$:

$$G \text{ filters } \frac{1}{\sqrt{2}}\begin{bmatrix} x(0)-x(-1) \\ 0 \\ x(2)-x(1) \\ 0 \\ \cdot \end{bmatrix} \text{ to give } \frac{1}{2}\begin{bmatrix} -x(0) & + & x(-1) \\ x(0) & - & x(-1) \\ -x(2) & + & x(1) \\ x(2) & - & x(1) \\ & \cdot & \end{bmatrix} = w_1.$$

The filter gives the right result, again without treating even and odd components differently. We caution that the highpass coefficients are in the order $-\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}$. When you transpose D, the order of coefficients is reversed. Then a delay makes G a causal filter (and the same for F).

The only other caution concerns the factors $\frac{1}{\sqrt{2}}$. They are present in all four filters. A less perfect symmetry would have $\frac{1}{2}$ in one bank and 1 in the other bank. This is exactly like the two-point discrete Fourier transform, where we often allow $\frac{1}{2}$ and 1 in the matrix and its inverse. The orthogonal matrix has $\frac{1}{\sqrt{2}}$ in both:

$$\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}^{-1} = \frac{1}{2}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \qquad \text{inverse matrices with 1 and } \tfrac{1}{2}$$

$$\begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}^{-1} = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \text{ orthogonal matrix with } 1/\sqrt{2}.$$

An orthogonal (or unitary) matrix has orthogonal rows and orthogonal columns normalized to be unit vectors. *The inverse is the transpose* (or conjugate transpose). It is an accident for this example that the matrix is real and symmetric. You can't see that it was transposed and conjugated.

Important. The connection between this Haar filter bank and the 2-point DFT is no accident. The simplest M-band filter bank comes from an M-point DFT. It is called a *uniform DFT filter bank.* We are seeing the case $M = 2$, written as an ordinary filter bank (and made causal by a delay).

We summarize this section with a schematic of the whole filter bank (Figure 1.7).

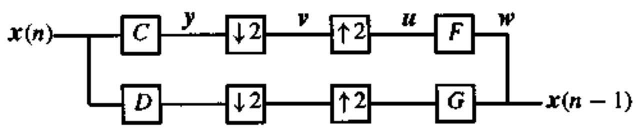


Figure 1.7: The analysis bank followed by the synthesis bank.

This allows us to indicate the symbols $\boldsymbol{y}$, $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{w}$ for the outputs at the four stages. For a two-channel bank, we mark those vectors in the lowpass and highpass channels by subscripts 0 and 1. When there are M channels, we need subscripts $0, 1, \ldots, M-1$.

From the filter bank, the next step is to wavelets.

Problem Set 1.4

1. Write down the matrix $(\downarrow 2)$ that executes downsampling: $(\downarrow 2)\,x(n) = x(2n)$.
2. Write down the transpose matrix $(\uparrow 2) = (\downarrow 2)^T$. Multiply the matrices $(\uparrow 2)\,(\downarrow 2)$ and $(\downarrow 2)\,(\uparrow 2)$. Describe the output from $(\uparrow 2)\,y(n)$.
3. Describe the output from $(\downarrow 2)^2\,x(n)$ and $(\uparrow 2)^2\,y(n)$.
4. Send the signal with $x(0) = x(1) = x(2) = 1$ through the whole filter bank, and give the output at every step.
5. Put each row $[\,-1 \quad 1\,]/\sqrt{2}$ of $B = (\downarrow 2)\,D$ after the row $[\,1 \quad 1\,]/\sqrt{2}$ of $L = (\downarrow 2)\,C$. In this order we see a *block transform*. What is the inverse transform?
6. Show how to delay an anticausal (= upper triangular) filter matrix so it becomes causal. If the coefficients $h(0), \ldots, h(N)$ are on the diagonals of the anticausal matrix, what are the diagonals after the delay? Give the two frequency responses, anticausal and causal.
7. The 4-channel bank analogous to Haar is based on the 4-point DFT matrix F_4. Find F_4 and the four analysis filters and the outputs after downsampling by $(\downarrow 4)$.
8. Suppose a matrix has the property that $Q^TQ = I$. Show that the columns of Q are mutually orthogonal unit vectors. Does it follow that $QQ^T = I$?
9. In a transmultiplexer, the synthesis bank comes *before* the analysis bank. Compute LL^T and LB^T and BB^T to verify that the Haar transmultiplexer still gives perfect reconstruction:
$$\begin{bmatrix} L \\ B \end{bmatrix}\begin{bmatrix} L^T & B^T \end{bmatrix} = \begin{bmatrix} LL^T & LB^T \\ BL^T & BB^T \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}.$$
10. Compute the subband outputs $v_0(n)$ and $v_1(n)$ for the average-difference filter bank with input $x(n) = 0,\ 1,\ -1,\ 2,\ 5,\ 1,\ 7,\ 0$. Reconstruct the signal by feeding $v_k(n)$ into the synthesis bank in Figure 1.6. Verify that the output is $x(n-1)$.
11. If $H_0(z) = 1$ and $H_1(z) = z^{-1}$ (no filtering) write the entries of $\begin{bmatrix} L \\ B \end{bmatrix}\begin{bmatrix} L^T & B^T \end{bmatrix} = I$.

1.5 Scaling Function and Wavelets

Corresponding to the lowpass filter, with $h(0) = \frac{1}{2}$ and $h(1) = \frac{1}{2}$, there is a continuous-time scaling function $\phi(t)$. Corresponding to the highpass filter, with coefficients $\frac{1}{2}$ and $-\frac{1}{2}$, there is a wavelet $w(t)$. We now describe the *dilation equation* that produces $\phi(t)$ and the *wavelet equation* for $w(t)$.

You will see how the filter coefficients (the c's and d's) enter these equations. Two time scales also appear. This joint appearance of t and $2t$ is the novelty of the dilation equation. It is also the source of difficulty! We have a "two-scale difference equation". Later we develop the background of these equations and a general method for solving them. Here we quickly recognize the box function as $\phi(t)$. Then we construct the wavelet $w(t)$ and use it.

The dilation equation for the scaling function $\phi(t)$ is

$$\phi(t) = \sqrt{2}\sum_{k=0}^{N} c(k)\,\phi(2t-k). \tag{1.26}$$

In terms of the original lowpass coefficients $h(k)$, the extra factor is 2:

$$\phi(t) = 2\sum_{k=0}^{N} h(k)\,\phi(2t-k). \tag{1.27}$$

This involves a function $\phi(t)$ in continuous time, and a set of coefficients $c(k)$ or $h(k)$ from discrete time. The presence of t and $2t$ is the key. Without the 2 we would have an ordinary constant-coefficient equation (look for exponential solutions $e^{\lambda t}$). With two time scales, there are major changes:

1. There may or may not be a solution $\phi(t)$.
2. The solution is zero outside the interval $0 \le t < N$.
3. The solution seldom has an elementary formula.
4. The solution is not likely to be a smooth function.

Formally, we can find an expression for the Fourier transform of $\phi(t)$. It is an infinite product (Section 6.4). The inverse Fourier transform yields $\phi(t)$ as a "distribution"—not necessarily continuous, possibly involving delta functions. (Those are impulses. Their integrals are jumps. More cautious people call them steps.) In our Haar example, the solution $\phi(t)$ lies just outside the class of continuous functions—it has a jump.

For the coefficients $2h(0) = 1$ and $2h(1) = 1$, the dilation equation is

$$\phi(t) = \phi(2t) + \phi(2t-1). \tag{1.28}$$

The graph of $\phi(t)$ is *compressed by* 2, to give the graph of $\phi(2t)$. When that is *shifted to the right by* $\frac{1}{2}$, it becomes the graph of $\phi(2t-1)$. We ask the two compressed graphs to combine into the original graph. Figure 1.8 shows that this occurs when $\phi(t)$ is the *box function*:

$$\phi(t) = \begin{cases} 1 & \text{for } 0 \le t < 1 \\ 0 & \text{otherwise.} \end{cases}$$

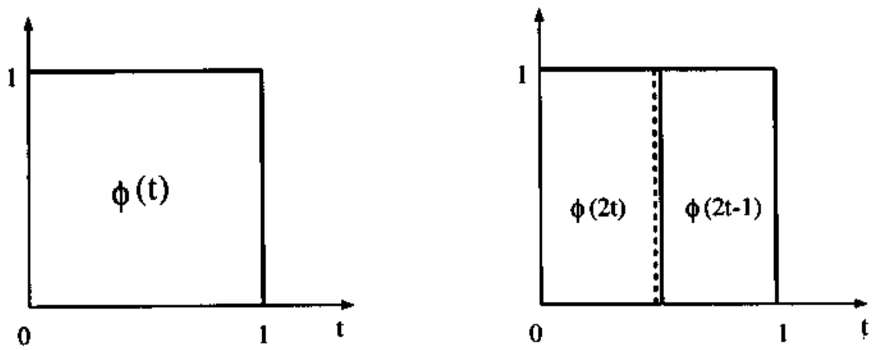


Figure 1.8: The box function with dilation and translation.

The graphs of $\phi(2t)$ and $\phi(2t-1)$ are half-size boxes. Their sum is the full-size box $\phi(t)$.

As planned, we wrote down the solution rather than deriving it. The dilation equation is linear, so any multiple of the box function is also a solution. It is convenient to normalize so that *the integral of $\phi(t)$ from $-\infty$ to ∞ equals one.* Note that a solution $\phi(t)$ covering unit area is only possible when the coefficients in the dilation equation add up to 2.

Theorem 1.1 *If* $\int_{-\infty}^{\infty} \phi(t)\,dt = 1$ *then* $2h(0) + 2h(1) + \cdots + 2h(N) = 2.$

Proof: The graph of $\phi(2t)$ and every $\phi(2t-k)$ is compressed to area $\frac{1}{2}$:

$$2\int_{-\infty}^{\infty} \phi(2t-k)\,dt = \int_{-\infty}^{\infty} \phi(u)\,du = 1 \qquad (\text{set } u = 2t-k). \tag{1.29}$$

So integrating both sides of the dilation equation $\phi(t) = 2\sum h(k)\phi(2t-k)$ gives $1 = h(0) + h(1) + \cdots + h(N)$. This is our lowpass filter convention.

Important. For the filter, then the filter bank, and finally the dilation equation, *the normalization is different.* This is clear from the actual numbers in our lowpass filter. The sum of coefficients is 1 or $\sqrt{2}$ or 2:

h: $\frac{1}{2}$ and $\frac{1}{2}$ in the single filter, adding to 1
c: $\frac{1}{\sqrt{2}}$ and $\frac{1}{\sqrt{2}}$ in the filter bank, adding to $\sqrt{2}$
$2h$: 1 and 1 in the dilation equation, adding to 2.

The single lowpass filter has sum $H(0) = 1$. That preserves the zero frequency DC term: output $Y(0) =$ input $X(0)$. The filter bank has a factor $\sqrt{2}$ to account for the downsampling step. There are only half as many components and half as many terms in the energy, when $\sum (y(n))^2$ is replaced by $\sum (y(2n))^2$. To compensate we must multiply $y(2n)$ by $\sqrt{2}$. That gives the normalization $C(0) = \sqrt{2}$ in place of $H(0) = 1$.

For the dilation equation, we have just seen why rescaling the time requires renormalizing the coefficients by 2 —to preserve area.

There are certainly filters in which $H(0)$ is not exactly 1. If $H(\omega)$ stays very near 1 over an interval around $\omega = 0$, this still deserves the name "lowpass filter". *Such filters do not lead to* *wavelets!* The requirements for wavelets are very strict, at $\omega = 0$ and $\omega = \pi$. Only a subset of special filters can pass the test, which starts with $H(0) = 1$ and $H(\pi) = 0$.

The box function is like the averaging filter — it smoothes the input. Convolution with the box function gives a moving average in continuous time, just as the filter coefficients $\boldsymbol{h} = \frac{1}{2}, \frac{1}{2}$ did in discrete time:

$$\boldsymbol{h} * (\ldots, x(0), x(1), \ldots) = \left(\ldots, \frac{x(0) + x(-1)}{2}, \frac{x(1) + x(0)}{2}, \ldots\right)$$

$$\phi(t) * x(t) = \int_{t-1}^{t} x(s)\, ds = \text{average over moving interval.} \tag{1.30}$$

There is a similar convolution with $w(t)$. Instead of picking up the smooth low frequency part of the function, the wavelet will lead to the high-frequency details. The coefficients for the Haar wavelet are 1 and -1.

The Wavelet Equation

The equation for the wavelet involves the highpass coefficients $\boldsymbol{d}(k)$. It is a direct equation that gives $w(t)$ immediately and explicitly from $\phi(t)$:

$$w(t) = \sqrt{2} \sum \boldsymbol{d}(k)\, \phi(2t - k). \tag{1.31}$$

In terms of the original coefficients $\boldsymbol{h}_1(k)$, the factor $\sqrt{2}$ becomes 2:

$$\textbf{\textit{Wavelet equation}} \qquad w(t) = 2 \sum \boldsymbol{h}_1(k)\, \phi(2t - k). \tag{1.32}$$

In our example, $\phi(t)$ is a box function and its dilations $\phi(2t-k)$ are half-boxes. Then the wavelet is a *difference of half-boxes*:

$$w(t) = \phi(2t) - \phi(2t - 1). \tag{1.33}$$

Explicitly, $w(t) = 1$ for $0 \le t < \frac{1}{2}$ and $w(t) = -1$ for $\frac{1}{2} \le t < 1$. This is the ***Haar wavelet***. Its graph is in Figure 1.9 along with the graphs of $w(2t)$ and $w(2t - 1)$. Those are wavelets at scale $2t$; their graphs are compressed and shifted. They join the original $w(t)$, and all its other dilations and translations, in the *wavelet basis*.

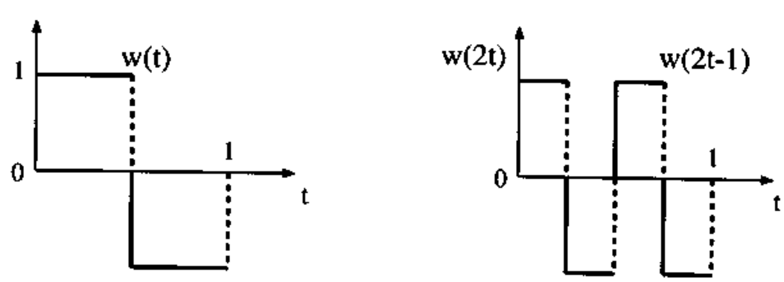


Figure 1.9: The Haar wavelet $w(t)$; the rescaled wavelets $w(2t)$ and $w(2t - 1)$.

It goes without saying that Alfred Haar did not call his function a "***wavelet***". He was writing in 1910 about this particular function; all other wavelets came later. That name emerged from the literature on geophysics, by a route through France. The word *onde* led to *ondelette*. In translation, the word wave led to wavelet. ***A wavelet is a small wave.*** In Haar's case it happened to be a square wave.

Comment. The square wave $w(t)$ has compact support. It comes from an FIR filter, with finite length. We use the word "support" for the closed interval in continuous time, here $[0, 1]$, outside which $w(t)$ is zero. When we close the set where it is nonzero, to include the jump location $t = \frac{1}{2}$ as well as the endpoints, we have found the support of the function $w(t)$. The words "compact support" mean that this closed set is bounded. The wavelet is zero outside a bounded interval: ***compact support corresponds to FIR.***

Wavelets need not have compact support! They can come from IIR filters instead of FIR filters. Historically, since the connection of wavelets to filters was not immediately recognized, the wavelets after Haar were constructed in other ways (with considerable difficulty). They oscillated above and below zero along the whole line $\infty < t < \infty$, decaying as $|t| \to \infty$. This still qualifies as a wavelet. It is a localized pulse that decreases to zero and *has integral zero.* Ingrid Daubechies showed in 1988 that compact support was possible for other wavelets than Haar's.

We will tell that story more completely in Chapter 6. From Haar onwards, one property that most designers hoped for was **orthogonality**. This means: $w(t)$ is orthogonal to all its dilations and translations. The wavelet basis, containing all these functions $w(2^j t - k)$, is an *orthogonal basis*. For the first few Haar wavelets that is easy to verify:

$$\begin{aligned} \text{inner product} &= \int_{-\infty}^{\infty} w(t)w(2t)\,dt &= 0 \\ \text{inner product} &= \int_{-\infty}^{\infty} w(t)w(2t-1)\,dt &= 0 \\ \text{inner product} &= \int_{-\infty}^{\infty} w(2t)w(2t-1)\,dt &= 0. \end{aligned}$$

In the first integral, $w(t) = 1$ in Figure 1.9 where $w(2t)$ is positive and then negative. The integral is zero. Similarly $w(t) = -1$ on the second half-interval where $w(2t-1)$ is plus and minus. The second integral is therefore zero. The third integral vanishes for a different reason — the functions $w(2t)$ and $w(2t-1)$ do not overlap. One is zero where the other is nonzero. So the product $w(2t)\,w(2t-1)$ is zero everywhere.

The pattern continues for all translations by k and dilations by 2^j. Haar wavelets at the same scaling level (same j) do not overlap. They are orthogonal in the strictest way. When Haar wavelets at different levels j and J do overlap, the coarse one is constant where the fine one goes up and down. All integrals are zero, giving an orthogonal basis $w_{jk}(t) = w\left(2^j t - k\right)$:

$$\text{inner product} = \int_{-\infty}^{\infty} w\left(2^j t - k\right) w\left(2^J t - K\right) dt = 0. \tag{1.34}$$

A perfect basis is not only orthogonal but orthonormal. The functions have length 1. Like a unit vector, the inner product $< w(t), w(t) >$ is normalized to 1:

$$\text{length squared} = \int_{-\infty}^{\infty} (w(t))^2\, dt = 1.$$

This is true for the Haar wavelet. To make it true for the dilations of that wavelet, we multiply by $2^{j/2}$ — otherwise the compressed graphs cover less area. The same factor will apply to other wavelets, and we record it now:

Theorem 1.2 *The rescaled Haar wavelets $w_{jk}(t) = 2^{j/2}w\left(2^j t - k\right)$ form an orthonormal basis:*

$$\int_{-\infty}^{\infty} w_{jk}(t)\, w_{JK}(t)\, dt = \delta(j - J)\, \delta(k - K). \tag{1.35}$$

This Kronecker delta symbol equals zero except when $j = J$ and $k = K$. In that case the integral of $(w_{JK}(t))^2$ equals one. We need two indices j and k because there are two operations (dilation and translation) in constructing the basis.

Important note. For these Haar wavelets, orthogonality was verified by direct integration. For future wavelets, this integration is not desirable and not possible. We will not have elementary formulas for $\phi(t)$ and $w(t)$. Instead, we will know the coefficients $c(k)$ in the dilation equation and $d(k)$ in the wavelet equation. ***All information about $\phi(t)$ and $w(t)$ — their support interval, their orthogonality, their smoothness, and their vanishing moments — will be determined by and from the c's and d's.***

Second note. We have not said ***which space of functions*** has the wavelets $w_{jk}(t)$ as a basis. Actually there are many choices. The space starts with *finite* combinations $g(t) = \sum b_{jk} w_{jk}(t)$. Those functions are piecewise constant on binary intervals — length $1/2^j$ and endpoints $m/2^j$. Other functions $f(t)$ are in the space *if they are limits of these piecewise constant* g's:

$$\|f(t) - g_n(t)\| \to 0 \quad \text{for some sequence } g_n(t).$$

The choice of function space is decided by the choice of the norm $\|f - g_n\|$. A function $f(t)$ might be a limit of piecewise constants in the maximum norm but not in an integral norm, or vice versa. The most frequent choice is the L^2 norm, where the superscript 2 signals that we integrate the *square*:

$$\|f(t) - g(t)\| = \left(\int_{-\infty}^{\infty} |f(t) - g(t)|^2\, dt\right)^{1/2}.$$

This choice of the L^2 norm is popular for four major reasons:

1. The norm is directly connected to the inner product:

By definition	$\|f(t)\|^2$	$=$	$< f(t), f(t) >$
By the Schwarz inequality	$\|f(t)\|\, \|g(t)\|$	$\geq$	$\| < f(t), g(t) > \|$.

2. Minimizing the L^2 norm leads to linear equations. This is familiar from ordinary least squares problems. The word "squares" introduces the L^2 norm. When the functions $g(t)$ are restricted to a subspace, the closest one to $f(t)$ is $g(t) =$ *projection of $f(t)$ onto the subspace.* Projection leads to right angles and linear equations.

3. The Fourier transform *preserves the L^2 norm and the inner product* $< f, g >$. This is the Parseval identity:

$$\|f\|^2 = \int_{-\infty}^{\infty} |f(t)|^2\, dt \quad = \quad \int_{-\infty}^{\infty} |\widehat{f}(\omega)|^2\, d\omega = \|\widehat{f}\|^2 \tag{1.36}$$

$$< f, g > = \int_{-\infty}^{\infty} f(t)\, \overline{g(t)}\, dt \quad = \quad \int_{-\infty}^{\infty} \widehat{f}(\omega)\, \overline{\widehat{g}(\omega)}\, d\omega = < \widehat{f}, \widehat{g} > . \tag{1.37}$$

The reader will know that irrelevant but necessary factors of 2π should enter these equations. They depend on how we define the Fourier transforms. Since $\widehat{f}(\omega)$ and $\widehat{g}(\omega)$ can be complex, we introduced absolute values in (1.36) and complex conjugates in (1.37). Of course (1.37) becomes (1.36) if $g(t) = f(t)$.

4. For an orthonormal basis (like the Haar wavelet basis), the L^2 norm $\left\|\sum b_{jk} w_{jk}(t)\right\|^2$ equals the sum of squares of the coefficients b_{jk}:

$$\begin{aligned}\int \left(\sum b_{jk}\, w_{jk}(t)\right)^2 dt &= \sum\sum b_{jk}\, b_{JK} \int w_{jk}(t)\, w_{JK}(t)\, dt \\ &= \sum \left(b_{jk}\right)^2 .\end{aligned}$$

For all these reasons, and more, the L^2 norm is our choice. But we must mention that wavelets provide an unusually convenient basis for other norms and other function spaces. This makes them popular in functional analysis and harmonic analysis, which deal with the properties of functions. If we change the exponent from 2 to p, the space L^p contains all functions for which $\left(\int |f(t)|^p\right)^{1/p}$ is finite. This norm $\|f\|_p$ is not quite equal to $C\left(\sum \left|b_{jk}\right|^p\right)^{1/p}$, as it was for $p = 2$. But if the b_{jk} are wavelet coefficients, rather than Fourier coefficients, this sum lies between fixed bounds $A_p\, \|f\|_p$ and $B_p\, \|f\|_p$. Thus the absolute values $|b_{jk}|$ still indicate which functions have finite norm and belong to L^p.

In shorthand: The wavelets are an ***unconditional basis*** for L^p (no condition on the signs of the b_{jk}). The complex exponentials are conditional; phase information is needed on the b_{jk}.

We recognize that these last comments are "pure mathematics". The reader is invited to start learning that language too — if desired and not already achieved. It is rewarding and not difficult. The *wavelet transform* that connects $f(t)$ to its wavelet coefficients b_{jk} is absolutely central — to theory and also to applications.

We turn to the practical problem: *How to compute the coefficients b_{jk} quickly?* This has a very good answer for wavelet transforms. There is a recursive Fast Wavelet Transform comparable in speed and stability to the FFT for Fourier transforms.

Problem Set 1.5

1. If $w(t)$ has unit norm, so that $\int (w(t))^2 dt = 1$, show that the function $w_{jk}(t) = 2^{j/2} w(2^j t - k)$ also has unit norm.
2. What combination of the half-boxes $\phi(2t)$ and $\phi(2t-1)$ is closest in L^2 (least squares) to $f(t) = t^2$ for $0 \le t \le 1$?
3. The box function $\phi_s(t)$, shifted to the interval [3, 4], is what combination of the functions $\phi_s(2t-k)$?
4. Show that the convolution of the box function with a continuous-time signal is the continuous average in (1.30). The convolution formula is

$$\phi(t) * x(t) = \int_{-\infty}^{\infty} \phi(t-s)\, x(s)\, ds.$$

5. Given a combination $a_0\phi(2t) + a_1\phi(2t-1)$, express it as $A_0\phi(t) + B_0 w(t)$. Then invert to find a_0 and a_1 from A_0 and B_0.
6. What scaling function satisfies the three-scale equation $\phi(t) = \phi(3t) + \phi(3t-1) + \phi(3t-2)$? Find ***two wavelets*** that are combinations of $\phi(3t-k)$, orthogonal to $\phi(t)$ and to each other.

1.6 Wavelet Transforms by Multiresolution

The wavelet transform operates in continuous time (on functions) and in discrete time (on vectors). The input is $f(t)$ or $x(n)$. The output is the set of coefficients b_{jk}, which express the input in the wavelet basis. For functions and infinite signals, this basis is necessarily infinite. For finite length vectors with L components, there will be L basis vectors and L coefficients. The discrete wavelet transform, from L components of the signal to L wavelet coefficients, is expressed by an L by L matrix.

To begin, we derive formulas for the coefficients b_{jk}. In continuous time they involve integrals of $f(t)$ times $w_{jk}(t)$. In discrete time we are solving a linear system. The inverse transform involves the inverse matrix.

Then we show how the b_{jk} can be found recursively. Levels j and $j-1$ are connected. This reorganizes the matrix multiplication. The discrete wavelet transform (DWT) becomes the *fast* wavelet transform (FWT). The central idea in this fast recursion is *multiresolution.*

First come the two directions, synthesis and analysis, for an orthonormal basis:

$$\textbf{Synthesis}\text{ of a function:}\quad f(t)=\sum_{j,k} b_{jk}\,w_{jk}(t)$$

$$\textbf{Analysis}\text{ of a function:}\quad b_{JK}=\int_{-\infty}^{\infty} f(t)\,w_{JK}(t)\,dt. \tag{1.38}$$

In the matrix case, the wavelets are ordinary vectors. They go into the columns of the ***wavelet matrix*** S. To maintain the parallel with the continuous case, we use a double index jk for the column number. A single index would go from 1 to L, but the double index is more natural. Each wavelet vector has a position in time given by k and a position in frequency (better to say, in *scale*) given by j. The columns of the L by L matrix S are the discrete wavelets:

$$\textbf{Synthesis}\text{ in discrete time: } \boldsymbol{x}=S\boldsymbol{b}.$$

The rows of the L by L matrix A contain the "analyzing" wavelets:

$$\textbf{Analysis}\text{ in discrete time: } \boldsymbol{b}=A\boldsymbol{x}.$$

For Haar and all orthonormal wavelets, the columns of S are the same as the rows of A. Analysis and synthesis are related by $A=S^T$. In general they are related by $A=S^{-1}$.

The synthesis equation $\boldsymbol{x}=S\boldsymbol{b}$ multiplies each coefficient b_{jk} by the basis vector in column jk of S, and adds. This is just matrix multiplication: $S\boldsymbol{b}$ is a combination of the columns of S.

The analysis equation $A\boldsymbol{x}=AS\boldsymbol{b}=\boldsymbol{b}$ is completely parallel to the continuous formula $\int w_{jk}(t)f(t)\,dt=b_{jk}$. The left side is the inner product of $\boldsymbol{x}$ with each analyzing vector. The wavelet basis consists of *unit vectors*, so all formulas are simple. The basis is orthonormal. In the continuous case, $\int \left(w_{jk}(t)\right)^2 dt=1$. In the discrete case $S^TS=I$. Then $\boldsymbol{b}=S^T\boldsymbol{x}$ and no division by length squared is required.

Analysis is the ***inverse*** of synthesis. Why did we multiply by the *transpose matrix*, when we should have introduced the *inverse matrix*? For an orthonormal basis, the reason is fundamental: The inverse is the same as the transpose!

$$\text{Orthonormal columns: } S^TS=I \text{ means } S^{-1}=S^T.$$

When the basis is only orthogonal, and not normalized to unit vectors, S^TS is diagonal. By using the word *basis*, we ensured that all these matrices are square. In the rectangular case S would not have an inverse. There are too many columns to be independent; instead of a basis we have a *frame* (Section 2.6). Our formulas would give the "pseudoinverse" S^+ instead of S^{-1}.

Without orthogonality, the rows of $A = S^{-1}$ are *biorthogonal* to the columns of S:

$$(\text{row } i \text{ of } A) \cdot (\text{column } j \text{ of } S) = \delta(i - j).$$

Each row of S^{-1} is orthogonal to $L - 1$ columns of S. This is ***biorthogonality***. The columns of S are the *synthesis basis*, and the rows of $A = S^{-1}$ are the *analysis basis*.

Tree-Structured Filter Bank

We move from transforms toward *fast* transforms. The wavelet basis and the Fourier basis have special properties, beyond orthogonality. The scales j and $j - 1$ are closely related, just as the frequencies ω and $\omega/2$ are closely related. By taking advantage of these relations, the multiplications by S and S^T can be reorganized. We will explain the special properties, and then derive fast transforms (starting in this section with Haar wavelets).

In the Fourier case, the matrices become F and F^T —or actually $\overline{F}^T$, since the Fourier vectors are complex exponentials. These are the L-point DFT matrix and its inverse. The Fast Fourier Transform is summarized in Section 8.1 (on periodic problems). You will see that the FWT is asymptotically faster than the FFT, requiring only $O(L)$ steps instead of $O(L \ln L)$. (The Walsh basis, which is not local, brings back $L \ln L$. The components are ± 1, with no zeros.) All these transforms are so important and successful that we do not overemphasize the comparison. Our goal is to understand wavelets and Fourier both.

The recursive nature of wavelets is clearest when we construct a tree of filter banks (Figure 1.10). The highpass filter D computes differences of the input. The downsampling step symbolized by $(\downarrow 2)$ keeps the even-numbered differences $(x(2k) - x(2k-1))/\sqrt{2}$. These are final outputs because they are not transformed again. They are at the end of their branch, in this "logarithmic tree". These outputs b_{jk} are at the fine-mesh level. The factor $r = 1/\sqrt{2}$ is included to produce unit vectors in the matrix S, below.

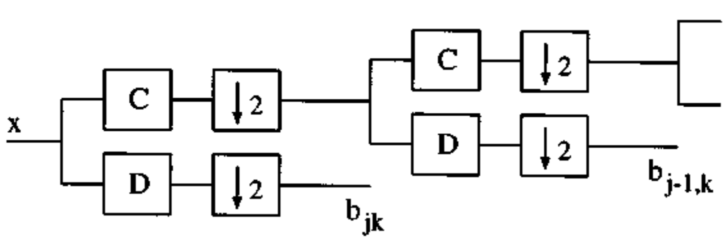


Figure 1.10: The logarithmic tree that leads to wavelets.

The lowpass filter C computes averages. Again the $(\downarrow 2)$ step keeps the even samples. These averages are not final outputs, because they will be filtered again by D and C. The averages and differences of all levels follow the fundamental recursion:

$$\begin{aligned} &\textit{Averages (lowpass filter)} \quad && a_{j-1,k} = \tfrac{1}{\sqrt{2}}\left(a_{j,2k} + a_{j,2k+1}\right) \\ &\textit{Differences (highpass filter)} \quad && b_{j-1,k} = \tfrac{1}{\sqrt{2}}\left(a_{j,2k} - a_{j,2k+1}\right). \end{aligned} \tag{1.39}$$

These filters are anticausal. ***There is a time-reversal here.*** That requirement is built in to convolutions and inner products. We will discuss it again below, for functions in continous time. The essential thing is to see *filters with downsampling* in equation (1.39). Each step takes us from a finer level j to a coarser level $j-1$, with half as many outputs.

You can see this logarithmic tree as a ***pyramid*** of averages and differences. The averages are sent up the pyramid, to be averaged again (and also differenced). Whenever a difference is computed, it is final. The sum of $\frac{1}{2}+\frac{1}{4}+\frac{1}{8}+\ldots$ gives 1, representing 100% of the output in the limit.

Figure 1.11 shows a finite pyramid. The input vector $\boldsymbol{x}$ at the bottom has length $L=2^J$. It is at level J, where we find 2^{J-1} differences and averages. The averages are the inputs at the next level $J-1$. Eventually we reach level 0 with an overall average and an overall difference (second half average − first half average). Keep this overall average as the final component — you might say the zeroth component — of the wavelet transform. Starting at level $J=3$, where the input $\boldsymbol{x}$ has $L=2^J=8$ components, the count of wavelet coefficients is

$$4 \text{ differences} + 2 \text{ differences} + 1 \text{ difference} + \text{overall average} = 8.$$

The seven differences are wavelet coefficients b_{jk}. The overall average can be denoted for convenience by a_{00}. In the model case of infinite length, the iteration of the lowpass filter can go on forever.

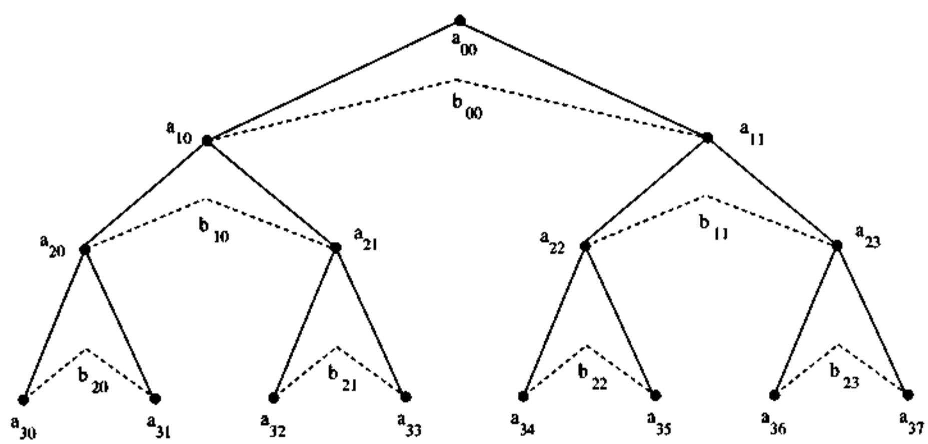


Figure 1.11: Averages a_{jk} go up the pyramid. Differences b_{jk} stop.

You must see the finite case in terms of matrices. With length $L=4$, there are two fine differences, one coarse difference, and the overall average. The columns of S are the basis vectors for the Haar wavelets:

$$\begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & -1 & 0 \\ 1 & -1 & 0 & 1 \\ 1 & -1 & 0 & -1 \end{bmatrix} \text{ is scaled by } r=\frac{1}{\sqrt{2}} \text{ to } \boldsymbol{S}=\begin{bmatrix} r^2 & r^2 & r & 0 \\ r^2 & r^2 & -r & 0 \\ r^2 & -r^2 & 0 & r \\ r^2 & -r^2 & 0 & -r \end{bmatrix}. \qquad (1.40)$$

The scaling gives unit vectors in the columns and rows, because $2r^2=1$ and $4r^4=1$. The inverse matrix appears in analysis, as we create the tree of averages and differences:

$$A = S^{-1} = S^T = \begin{bmatrix} r^2 & r^2 & r^2 & r^2 \\ r^2 & r^2 & -r^2 & -r^2 \\ r & -r & 0 & 0 \\ 0 & 0 & r & -r \end{bmatrix}. \tag{1.41}$$

The first row gives the overall average a_{00}. The second row gives b_{00}. The third and fourth rows give the finer differences, coming earlier in the tree of filters. The transform $\boldsymbol{b} = \boldsymbol{A}\boldsymbol{x}$ contains these four numbers.

Fast Wavelet Transform (FWT)

The tree of filters is the fast wavelet transform!! The FWT expresses the analysis matrix $\boldsymbol{A}$ as a product of simple average-difference matrices, coming from the downsampled filters in the tree:

$$A = \begin{bmatrix} r & r & & \\ r & -r & & \\ & & 1 & \\ & & & 1 \end{bmatrix} \begin{bmatrix} r & r & & \\ & & r & r \\ r & -r & & \\ & & r & -r \end{bmatrix} \tag{1.42}$$

The matrix on the right is first in the tree. The coefficients r, r are in the lowpass filter, and downsampling leaves a matrix $\boldsymbol{L}$ in the top two rows. The rows of $\boldsymbol{L}$ have a *double shift*. The coefficients $r, -r$ come from the highpass (or bandpass) filter, downsampled to leave $\boldsymbol{B}$. ***There is again a time-reversal from*** $-\boldsymbol{r}, \boldsymbol{r}$. The pattern that you see in the 4 by 4 Haar matrix is the product in (1.42):

$$\textit{Analysis tree:} \quad A = \left[\begin{array}{c|c} \begin{matrix} L \\ B \end{matrix} & \\ \hline & I \end{array}\right] \begin{bmatrix} L \\ B \end{bmatrix}.$$

This pattern applies to transforms of all lengths $L = 2^J$. ***It will apply to all wavelets!*** The fast transform expresses $\boldsymbol{A}$ (and later S) using *matrices with many zeros*. It is the matrix form of the *pyramid algorithm*.

For length $L = 2^J$, there will be J levels in the tree and J matrices in $\boldsymbol{A}$. The matrix on the right, from the start of the tree, has two nonzeros in each row. Filters with T coefficients will produce T nonzeros in each row. Then the finest factor has TL nonzero entries.

The next factor has $TL/2$ coefficients in the top half, processing a shorter input. (The identity matrix in the lower right costs nothing. It leaves the differences alone.) The third stage of the tree has $TL/4$ coefficients. The total for the fast wavelet transform (= factored form of W^{-1}) comes from J factors:

$$TL\left(1 + \frac{1}{2} + \frac{1}{4} + \cdots + \frac{1}{2^{J-1}}\right) < 2TL. \tag{1.43}$$

Theorem 1.3 *The fast wavelet transform computes the L coefficients $\boldsymbol{b} = \boldsymbol{A}\boldsymbol{x}$ in less than $2TL$ multiplications.*

The same count applies to the synthesis step $\boldsymbol{x} = S\boldsymbol{b}$. The factors of S give the inverse of equation (1.42). The synthesis tree has the inverse matrices in opposite order:

$$S = \begin{bmatrix} r & & r & \\ r & & -r & \\ & r & & r \\ & r & & -r \end{bmatrix} \begin{bmatrix} r & r & & \\ r & -r & & \\ & & 1 & \\ & & & 1 \end{bmatrix}. \tag{1.44}$$

The matrix on the right inverts the last analysis filter (still with time-reversal). It is the first step in the synthesis tree. The product of J matrices is the whole synthesis bank, which reconstructs x in Figure 1.12.

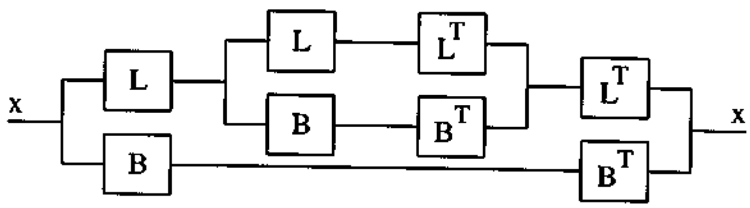


Figure 1.12: Analysis bank followed by synthesis bank: $\boldsymbol{SAx} = \boldsymbol{x}$. The matrices are factored by the tree, producing the FWT.

Haar Wavelets and Recursion

We move now to continuous time. The input $f(t)$ is a function instead of a vector. The output is the set of coefficients b_{jk} that multiply the wavelet basis functions $w_{jk}(t)$. These coefficients are inner products of $f(t)$ with $w_{jk}(t)$. The basis functions are normalized to unit length by the factor $2^{j/2}$:

$$b_{jk} = < f, w_{jk} > = \int_{-\infty}^{\infty} f(t)\, 2^{j/2}\, w\left(2^j t - k\right)\, dt. \tag{1.45}$$

For Haar, the wavelets are piecewise constant. The original wavelet $w(t) = w_{00}(t)$ is $+1$ on the interval $[0, \frac{1}{2})$ and -1 on the interval $[\frac{1}{2}, 1)$. The basis function w_{jk} is $2^{j/2}$ on a subinterval of length $\frac{1}{2}2^{-j}$ and $-2^{j/2}$ on the next subinterval. There are four wavelets at level $j = 2$, when we start on the unit interval $[0, 1)$. There are 2^j wavelets at level j.

To compute all the inner product integrals at levels $j = 0$, $j = 1$, and $j = 2$, we can integrate $f(t)$ over all eight subintervals of length $\frac{1}{8}$. This gives eight numbers. *How do those numbers produce b_{2k} and b_{1k} and b_{00} and a_{00}?*

The answer is beautiful. Those eight numbers are exactly the level 3 averages $a_{30}, a_{31}, a_{32}, \ldots, a_{37}$. We act on them exactly as in discrete time: filter and downsample! *The numbers at level $j-1$ come directly from the numbers at level j.* This is because the functions at level $j-1$ come from the functions at level j. The box function is a sum of half-boxes and the wavelet is a difference of half-boxes:

$$\phi(t) = \phi(2t) + \phi(2t-1) \text{ gives } \phi_{j-1,k}(t) = \frac{1}{\sqrt{2}}\left[\phi_{j,2k}(t) + \phi_{j,2k+1}(t)\right] \tag{1.46}$$

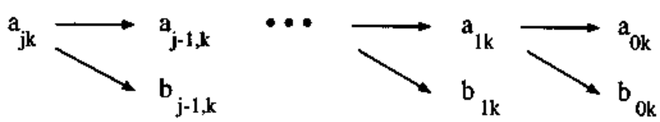
$$w(t) = \phi(2t) - \phi(2t-1) \text{ gives } w_{j-1,k}(t) = \frac{1}{\sqrt{2}}\left[\phi_{j,2k}(t) - \phi_{j,2k+1}(t)\right]. \tag{1.47}$$

Please look at these equations. They are the dilation equation and wavelet equation. The scaling function $\phi_{jk}(t) = 2^{j/2}\phi(2^j t - k)$ is constant on the interval of length 2^{-j} starting at $t = k2^{-j}$. When we take its inner product with $f(t)$, we are integrating $f(t)$ over this subinterval. Multiply the two equations by $f(t)$ and integrate:

$$\begin{array}{ll} \textit{Scaling coefficient:} & a_{j-1,k} = \frac{1}{\sqrt{2}}(a_{j,2k} + a_{j,2k+1}) \\ \textit{Wavelet coefficient:} & b_{j-1,k} = \frac{1}{\sqrt{2}}(a_{j,2k} - a_{j,2k+1}). \end{array} \tag{1.48}$$

The coefficients follow the pyramid algorithm. Notice again the time-reversal in which $2k+1$ appears. Our coefficients $a_{jk} = < f, \phi_{jk}(t) >$ and $b_{jk} = < f, w_{jk}(t) >$ are inner products, and a filter (a convolution $\sum h(k)\, x(n-k)$) has this reversal.

The main point is the pyramid. This beautiful connection between wavelets and filter banks was discovered by Stéphane Mallat. The pyramid algorithm is also called the Mallat algorithm (don't pronounce the 't'). It is a tree of butterflies in the Preface, it is a tree of filters in Figure 1.11, and a third form of the tree shows the trunk of averages a_{jk} and the branches of differences b_{jk}:



Wavelet coefficients at level $j+1$ are differences of scaling coefficients at level j.

This pyramid is also an equality of functions. A function at fine resolution j is equal to a combination of "average plus detail" at coarse resolution $j-1$:

$$\sum_k a_{jk}\, \phi_{jk}(t) = \sum_k a_{j-1,k}\, \phi_{j-1,k}(t) + \sum_k b_{j-1,k}\, w_{j-1,k}(t). \tag{1.49}$$

Multiresolution in Continuous Time

The equality of functions in (1.49) is also an equality of *function spaces*. On the left side is a combination of ϕ's at level j. Let V_j denote the space of all such combinations. On the right side is a combination of ϕ's at level $j-1$. This is a function in the scaling space V_{j-1}. Also on the right is a combination of w's at level $j-1$. This is a function in the wavelet space W_{j-1}. The key statement of multiresolution is

$$V_j = V_{j-1} \oplus W_{j-1} \tag{1.50}$$

The symbol $+$ for vector spaces means that every function in V_j is a sum of functions in V_{j-1} and W_{j-1}, as in equation (1.49). The symbol $\oplus$ for "direct sum" means that those smaller spaces meet only in the zero function. This is guaranteed when ***the two subspaces are orthogonal***. In that case the direct sum $\oplus$ becomes an "orthogonal sum". For emphasis we restate the definition of the subspaces:

$$V_j = \text{all combinations } \sum_k a_{jk}\phi_{jk}(t) \text{ of scaling functions at level } j$$
$$W_j = \text{all combinations } \sum_k b_{jk} w_{jk}(t) \text{ of wavelets at level } j.$$

V_j is spanned by translates of $\phi\left(2^j t\right)$ and W_j is spanned by translates of $w\left(2^j t\right)$. The time scale is 2^{-j}. At each level, all inner products are zero. We have two orthogonal bases for V_j, either the ϕ's at level j or the ϕ's and w's at level $j-1$.

Notice how this multiresolution grows to three levels or more:

$$V_3 = V_2 \oplus W_2 = V_1 \oplus W_1 \oplus W_2 = V_0 \oplus W_0 \oplus W_1 \oplus W_2. \tag{1.51}$$

On the left side are all piecewise constant functions on intervals of length $\frac{1}{8}$. On the right side is the same space of functions, differently expressed. The functions in V_0 are constant on $[0, 1)$. The functions in W_0, W_1, W_2 are combinations of wavelets. The function $f(t)$ in V_3 has a piece $f_j(t)$ in each wavelet subspace W_j (plus V_0):

$$f(t) = \sum_k a_{0k}\phi_{0k}(t) + \sum_k b_{0k}w_{0k}(t) + \sum_k b_{1k}w_{1k}(t) + \sum_k b_{2k}w_{2k}(t). \tag{1.52}$$

Note to the reader. The parallels between a filter tree in discrete time and multiresolution in continuous time are almost perfect. The filter bank separates lower and lower frequencies, as we iterate. Multiresolution uses longer and longer wavelets, as we climb the pyramid. We are speaking of the analysis half, where inputs are separated by *scale*.

Discrete time	***Continuous time***
filter bank tree	multiresolution
downsampling $\omega \to \frac{\omega}{2}$	rescaling $t \to 2t$
lowpass filter	averaging with $\phi(t)$
highpass filter	detailing with $w(t)$
orthogonal matrices	orthogonal bases
analysis bank output	wavelet coefficients
synthesis bank output	sum of wavelet series
product of filter matrices	fast wavelet transform

The reader will understand that in writing about Haar wavelets, we are writing about all wavelets. The pattern is fundamental, the pieces in the pattern can change. The actual $\boldsymbol{c}(k)$ and $\boldsymbol{d}(k)$ and $\phi(t)$ and $w(t)$ are at our disposal. The next chapters move to *filter design*, where we make choices. Those choices determine the wavelet design. Some filters and wavelets are better than others. We will not allow ourselves to forget the pattern that makes all of them succeed.

Caution. The Haar wavelets are orthogonal. Thus they are biorthogonal *to themselves*. We are not seeing a clear difference between analysis and synthesis filters, $\boldsymbol{C}$ and $\boldsymbol{D}$ versus $\boldsymbol{F}$ and $\boldsymbol{G}$. For the same reason we are not seeing the dual functions $\widetilde{\phi}(t)$ and $\widetilde{w}(t)$. A clue to this shadow world (or tilde world) is in the time-reversals, which involve the *next* sample $2k + 1$ instead of the previous sample $2k - 1$. This suggests filters $\boldsymbol{C}^T$ and $\boldsymbol{D}^T$ rather than $\boldsymbol{C}$ and $\boldsymbol{D}$.

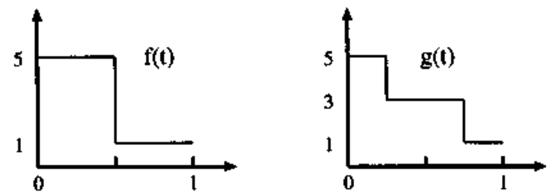
The biorthogonal case comes in Section 6.5. It has *two* multiresolutions, $\widetilde{V}_{j+1} = \widetilde{V}_j \oplus \widetilde{W}_j$ in parallel with $V_{j+1} = V_j \oplus W_j$. The pyramid and the fast wavelet transform go one way in analysis (with tilde). The inverse transform goes the other way in synthesis (without tilde).

Problem Set 1.6

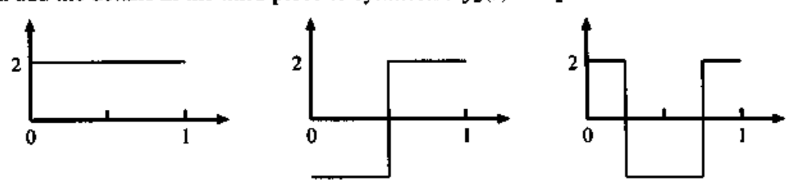
1. (a) Show that the exact sum in equation (1.43) is $2T(N-1)$.

   (b) How many matches are needed to decide the winner in a knockout tournament with N players? The average is the winner that goes to the next round (next filter). The difference is the loser that stops.

2. Write out the factorization of Haar's A for $N = 8$ (following 1.42).
3. For $N = 8$, write out the factorization of S corresponding to (1.44).
4. Draw the synthesis tree, the reverse of Mallat's analysis tree. A similar pyramid algorithm was proposed by Burt and Adelson before wavelets were named.
5. Split the function $f(t)$ into a scaling function plus a wavelet.



6. Split the function $g(t)$ into its pieces in V_0, W_0, and W_1 (box plus up-down coarse wavelet plus two fine wavelets).
7. The function $g(t)$ is in V_2 (its scale is $\frac{1}{4}$). The pyramid splits it first into a function in V_1 (two half-size boxes) plus a function in W_1 (two half-size wavelets). Find those pieces. Then split the piece in V_1 into a full-size box plus a full-size wavelet.
8. These three pieces are in V_0 and W_0 and W_1. Synthesize $f_1(t)$ in V_1 from the first two pieces. Then add the details in the third piece to synthesize $f_2(t)$ in V_2.



9. Suppose $H(t - \frac{1}{3})$ is the unit step function with jump at $t = \frac{1}{3}$. Its inner products a_{jk} with the boxes $\phi_{jk}(t)$ will be nonzero for about two-thirds of the 2^j boxes on $[0, 1]$. How many inner products b_{jk} with the wavelets $w_{jk}(t)$ will be nonzero?

   This example shows the compression of step functions by Haar wavelets.