The Search for a Good Basis

Gilbert Strang

Department of Mathematics

Massachusetts Institute of Technology

gs@math.mit.edu

http://www-math.mit.edu/~gs

Abstract

Good basis functions have many properties at once. For fast computations they should be local (involving banded matrices). They should give good approximation to piecewise smooth functions. For numerical stability the basis functions might be orthogonal. If they are also *shift-invariant*, at least in blocks, the calculations can use simple convolution filters.

To see quickly the key points, consider four rows of the infinite matrices H and H_b (Toeplitz and block Toeplitz):

$$H = \begin{bmatrix} d & c & b & a & & & \\ & d & c & b & a & & \\ & & d & c & b & a & \\ & & & d & c & b & a \end{bmatrix} \qquad H_b = \begin{bmatrix} d & c & b & a & & & \\ -a & b & -c & d & & & \\ & & d & c & b & a \\ & & -a & b & -c & d \end{bmatrix}$$

H represents one filter whereas *the block Toeplitz H_b represents a filter bank.* The rows of H cannot be orthogonal, especially rows 1 and 4. Worse than that, H cannot have a banded inverse. H_b achieves both properties if $ac + bd = 0$. That makes row 1 orthogonal to row 3 (it is automatically orthogonal to rows 2 and 4). The inverse is the transpose, still banded. This is the construction that led Daubechies to orthogonal compactly supported wavelets.

We discuss filter matrices and the wavelet basis. Our principal application is to compression of images. For that purpose a symmetric filter (with $a = d$ and $b = c$) would be better. But then orthogonality is lost; the construction must change to maintain a banded inverse. The algebra deals with the ring of 2×2 matrix polynomials, and the applications are remarkably widespread.

Introduction

This article is about compressing vectors—making them shorter. Those vectors typically come from images. Each component gives the gray scale of one pixel, and the vector might have $n = (512)^2$ components (triple for color). If you have waited for images on the Internet, you know why compression is desirable. If you are a radiologist, with megabytes of data from every MR and CT scan, compression is quite urgent (and also harder). A doctor can transform, but must not lose, the small piece of the vector that is clinically crucial.

One part of this problem is pure linear algebra: *To change to a new basis in* $\mathbf{R}^n$. The standard basis, matching components to pixels, is surely not the best. If a quarter of the image is completely white, we shouldn't use $n/4$ components to represent that part. It is an article of faith in signal processing that a Fourier basis is better. We can keep the largest Fourier coefficients (and destroy all the small ones). That type of compression is called hard thresholding. Smooth variations in intensity are well represented by the low frequency terms. But edges that were easy in the standard basis have become extremely expensive—because of the slow $1/k$ decay of Fourier coefficients, and the ripples from the Gibbs phenomenon when the series is truncated. These shadows near a discontinuity are called *ringing*.

A much-used alternative is to divide the image into 8×8 blocks. The discrete Fourier transform (or better, the discrete cosine transform) is applied to each block separately. This standard is heavily criticized because it produces "blocking" at a high compression ratio—the 8×8 pieces don't fit smoothly when 95% of the basis vectors are left out. *What we want is a basis of overlapping "local" vectors.* The change of basis will be fast when all matrices have a fixed number of nonzero entries in each row. The effect of an edge is controlled, and the overlapping reduces artificial blocking. Where a step function is disastrous in the Fourier basis, its influence is well concentrated in a local basis.

A second desirable property is shift-invariance or "stationarity" of the basis. If the input is shifted (in the $x - y$ plane for images and in time for audio signals), then the output would be equally shifted. For the change of basis matrix, this means that a shift to the right is identical to a shift up. All rows of the matrix H are translates of the zeroth row. The transform is represented by a *Toeplitz matrix* $H_{ij} = h(i - j)$, with a constant coefficient $h(k)$ down the kth diagonal. (Note that a two-dimensional convolution has double indices (i_1, i_2) and (j_1, j_2). A finite image stops at n pixels.) Our model will be one-dimensional, extended to the whole line: $i \in \mathbf{Z}$ and $j \in \mathbf{Z}$.

A Toeplitz matrix H is a convolution operator on $\ell^2(\mathbf{Z})$, known simply as a *filter*:

$$H = \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & & & \\ & h(3) & h(2) & h(1) & h(0) & & \\ & & h(3) & h(2) & h(1) & h(0) & \\ & & & \cdot & \cdot & \cdot & \cdot \end{bmatrix}$$

The rows overlap and are local (only four nonzeros in this example). In the Fourier basis, H is diagonal! Each pure exponential $v(n) = e^{in\omega}$ is formally an eigenvector of H, and the eigenvalue is the *frequency response* $H(\omega)$:

$$Hv = \left(h(0) + h(1)e^{-i\omega} + \cdots + h(N)e^{-iN\omega}\right) v = H(\omega)v\,. \tag{1}$$

In the Fourier basis, this changes $y = Hx$ into an ordinary multiplication $Y(\omega) = H(\omega)X(\omega)$:

$$\text{If } x(n) = \int_{-\pi}^{\pi} X(\omega)e^{in\omega}\,d\omega \quad \text{then } y(n) = \int_{-\pi}^{\pi} H(\omega)X(\omega)e^{in\omega}\,d\omega\,. \tag{2}$$

By choosing suitable numbers $h(0)$, $h(1)$, . . . , $h(N)$, we can design the filter to stop a band of frequencies. Figure 1 shows the frequency response of a *lowpass filter:* $H(\omega) \approx 0$ at high frequencies $|\omega| \approx \pi$. Eventually we will construct a bank of filters—not just one.

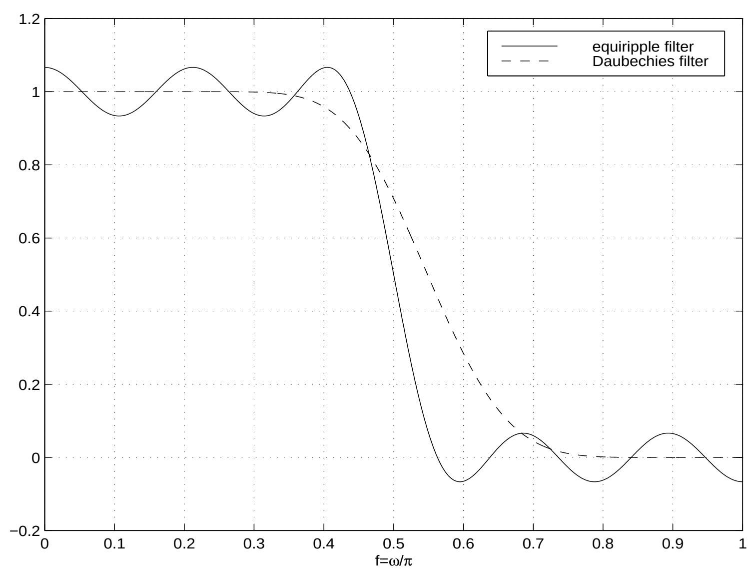


Figure 1: Two lowpass filters, with passband and stopband.

Note 1 We have expressed the filter as a matrix, believing that readers will see quickly how it operates. In the time domain the action of H is a discrete convolution:

$$y(n) = \sum_{k=0}^{N} h(k)x(n-k). \tag{3}$$

In the frequency domain this transforms to the all–important *convolution rule* $Y(\omega) = H(\omega)X(\omega)$. Each term $y(n)e^{-in\omega}$ comes from $h(k)e^{-ik\omega}$ times $x(n-k)e^{-i(n-k)\omega}$, summed on k, and this recovers (3). Even simpler is to use formal power series $Y(z) = H(z)X(z)$:

$$\sum y(n)z^{-n} = \left(\sum h(k)z^{-k}\right)\left(\sum x(\ell)z^{-\ell}\right).$$

The key point is $k+\ell = n$. That applies to powers of z and powers of $e^{i\omega}$. The indices in the convolution add to $k + (n-k) = n$. This representation of H by its transfer function $H(z)$ makes the algebra very convenient.

In all cases the crucial information is carried by the filter coefficients $h(0), \ldots, h(N)$. This vector h is the *"impulse response"*. It is the output (one column of the matrix H) when the input x is a unit impulse: if $x(n) = \delta(n)$ then $y(n) = h(n)$. In the z-domain $X(z) \equiv 1$ yields $Y(z) \equiv H(z)$. The design of suitable filters is totally fundamental to signal processing.

The matrix form begins to dominate when the signal length is finite (the image has boundaries). Toeplitz himself studied that problem; the true "Toeplitz matrices" are sections of our doubly-infinite H. But just beheading the matrix gives an unwelcome discontinuity, and boundary filters are needed in the boundary rows. References are in Chapter 8 of [S-N].

Filter Banks

The lowpass filter reduces the high frequency noise (good). It also blurs the edges (not good). An ideal filter would jump from 1 to 0 in its frequency response. That jump needs infinitely many coefficients $h(n)$; truncation of the series produces Gibbs ringing

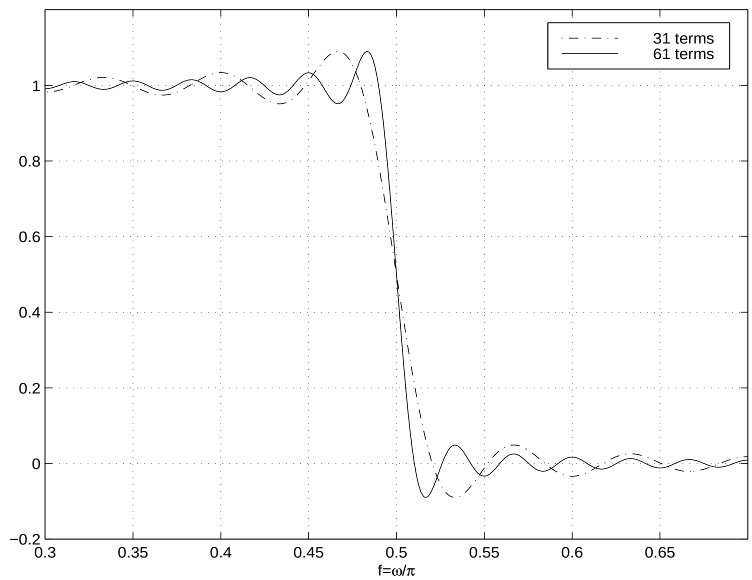
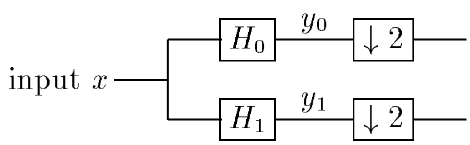


Figure 2: Gibbs phenomenon creates ringing around the edges.

in Figure 2, which nobody wants. The smoothing by a lowpass filter H is useful but the reconstruction by H^{-1} is terrible: $1/H(\omega)$ is not a polynomial and H^{-1} is not a banded matrix. In signal processing language, an FIR filter (a finite sequence $h(n)$) has an IIR inverse (infinitely long).

For fast computations, we want *a banded matrix with a banded inverse.* Then the change of basis will be fast both ways. The analysis step Hx is from old basis to new, and the synthesis step reconstructs $x = H^{-1}(Hx)$ from the basis vectors in the columns of H^{-1}. This goal of a banded inverse can be achieved, but one filter is not enough.

From all directions, we are being pushed toward two (or more) filters. H_0 will be lowpass and H_1 will be highpass. The outputs $y_0 = H_0x$ and $y_1 = H_1x$ will split the frequency spectrum into low and high, with some overlap. For an *invertible* filter bank, we only want half of each output (because two filters double the data). The neat idea of signal processing engineers is to downsample the two outputs by keeping only their even components $y_0(2n)$ and $y_1(2n)$. This produces a filter bank with two half-length outputs u_0 and u_1, from the filters H_0 and H_1:



$$u_0 = (\downarrow 2)H_0x = \sum h_0(k)x(2n-k)$$
$$u_1 = (\downarrow 2)H_1x = \sum h_1(k)x(2n-k)\,.$$

In each channel, we filter and downsample: $u(n) = y(2n)$. The product $(\downarrow 2)H$ is a Toeplitz matrix with every second row removed. (Now H represents H_0 or H_1.) There is a double shift between rows:

$$(\downarrow 2)H = \begin{bmatrix} \cdot & \cdot & & & & \\ h(3) & h(2) & h(1) & h(0) & & \\ & & h(3) & h(2) & h(1) & h(0) \\ & & & & \cdot & \cdot \end{bmatrix}.$$

This is the key operator in wavelet theory. After the convolution from H, downsampling replaces n by $2n$: $u(n) = \sum h(k)x(2n-k)$. Iteration will produce more time scales $2^j n$. In the limit of the iterations (suitably normalized), there will be a basis of wavelets at all scales $2^j t$. This step from discrete to continuous time is discussed below.

The downsampling operator $(\downarrow 2)$ produces the even part in the representation by power series (the z-domain):

$$U(z^2) = \frac{1}{2}\left(H(z)X(z) + H(-z)X(-z)\right).$$

That appearance of $-z$ signals *aliasing*. Frequencies ω and $\omega+\pi$ are mixed together $\left(-z \text{ is } e^{i(\omega+\pi)}\right)$. This aliasing will be cancelled in equation (4) by opposite aliasing in the highpass channel.

To see the complete filter bank, $(\downarrow 2)H_0$ joined with $(\downarrow 2)H_1$, interweave their rows:

$$
\text{Filter bank } H_b = \left[\begin{array}{cc|cc|cc}
\cdot & \cdot & & & & \\
\cdot & \cdot & & & & \\
h_0(3) & h_0(2) & h_0(1) & h_0(0) & & \\
h_1(3) & h_1(2) & h_1(1) & h_1(0) & & \\
\hline
 & & h_0(3) & h_0(2) & h_0(1) & h_0(0) \\
 & & h_1(3) & h_1(2) & h_1(1) & h_1(0) \\
 & & & & \cdot & \cdot \\
 & & & & \cdot & \cdot
\end{array}\right]
$$

That matrix H_b is *block* Toeplitz. Down its diagonals are 2×2 blocks, instead of scalars. This is the change of basis matrix. The transform $u = H_b x$ produces the two half-length vectors u_0 and u_1, interwoven. We fervently hope that many of their components are small (especially in the high pass output u_1). We have a better basis when x is well represented by a few basis vectors.

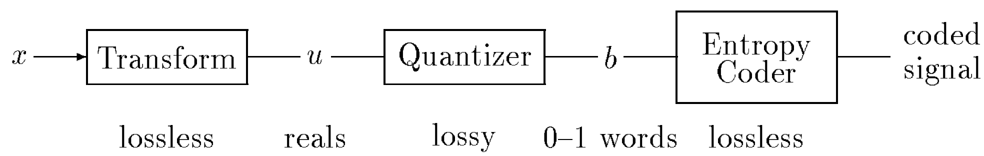
Note 2 A basis can be "good" for a specific class of inputs. If we have no information about x, we have no way to choose. In practice, a lot of signals are smooth. There is correlation between the sample $x(n)$ and its neighbor $x(n+1)$. The probability distribution for x will induce a probability distribution for its transform u—and this governs the compression.

I believe that an important problem, not truly solved, is to quantify the good properties of a basis. The problem becomes harder when we abandon the ℓ^2 norm (which is a poor measure of the visual errors that we really object to). The response of the human eye and ear at different frequencies, and the masking of nearby frequencies by a strong signal, lead to new error measures.

At present we mostly judge quality by looking at standard images. Expert eyes detect the effects of compression. We will mention at the end some undesirable artifacts that the professionals look for.

Image Coders

Transforming x to u (change of basis) is the first step in compressing an image. This is a lossless step; we can invert. Most of our paper will be about the construction of H_0 and H_1. But this is only part (the linear part) of the whole coder:



The quantizer takes real numbers $u(n)$ to binary numbers $b(n)$ of variable length. It is a sophisticated form of roundoff. Most engineers will say that *quantization is the most important step.* The allocation of bits to low and high channels governs the picture quality.

The final message to the receiver does not come one word (one component) at a time. Many components of b are zeros; they are literally rounded off! Instead of transmitting 100 consecutive zeros, the entropy coder might replace $00\ldots00$ by its runlength. Frequently repeated symbols are replaced by shorter symbols—the "Huffman Table" stores the meanings. (It is like a better Morse code, newly constructed for each message.) This step is lossless, and it reduces a typical message by 2:1 or 3:1. In UNIX you use instructions like *gzip* and *compress.*

For decoding, *gunzip* and *uncompress* recover the binary words $b(n)$. Your email file is reproduced exactly. Then for image compression the lossy quantization is approximately reversed: $\widehat{u}(n)$ is the center of the bin of numbers that are rounded off to $b(n)$. The basis vectors in the inverse matrix, times these $\widehat{u}(n)$, add up to the received message $\widehat{x}(n)$. After that synthesis step, the reconstructed image is displayed —and we judge its quality.

The Inverse Matrix and the Synthesis Filters

For one filter, the inverse matrix is not banded. The infinite coefficient sequence $f(0)$, $f(1), \ldots$ in the inverse matrix comes from $1/H(\omega)$:

$$FH = I \quad \text{when} \quad \left(f(0) + f(1)e^{-i\omega} + \cdots\right)\left(h(0) + \cdots + h(N)e^{-iN\omega}\right) = 1\,.$$

It is impossible for both factors to be nontrivial polynomials. But for filter banks and block Toeplitz matrices, a banded H *can* have a banded inverse. How can the product of two matrix polynomials be the identity matrix?

$$(Ae^{i\omega} + B)(C + De^{-i\omega}) = I.$$

The key is that $AC = 0$ and $BD = 0$ are possible with nonzero matrices. For scalars, A or C must be zero and B or D is zero. The scalar polynomials must reduce to trivial monomials. The matrix example below succeeds because

$$\begin{bmatrix} 1 & 1 \\ 3 & 3 \end{bmatrix}\begin{bmatrix} 3 & -1 \\ -3 & 1 \end{bmatrix} = \begin{bmatrix} 3 & -3 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} -1 & 3 \\ -1 & 3 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

The test for a polynomial inverse will be converted into equations (4) and (5).

We demonstrate with an example of F_b times H_b, or H^{-1} times H:

$$\begin{bmatrix} \cdot & \cdot & 1 & 1 & & & & \\ \cdot & \cdot & 3 & 3 & & & & \\ & & 3 & -3 & 1 & 1 & & \\ & & 1 & -1 & 3 & 3 & & \\ & & & & 3 & -3 & \cdot & \cdot \\ & & & & 1 & -1 & \cdot & \cdot \end{bmatrix}\begin{bmatrix} \cdot & \cdot & & & & & & \\ \cdot & \cdot & & & & & & \\ -1 & 3 & 3 & -1 & & & & \\ -1 & 3 & -3 & 1 & & & & \\ & & -1 & 3 & 3 & -1 & & \\ & & -1 & 3 & -3 & 1 & & \\ & & & & \cdot & \cdot & & \\ & & & & \cdot & \cdot & & \end{bmatrix} = 16I\,.$$

The new basis vectors are the columns on the left, from the synthesis matrix. The change of basis (this is a tricky lecture in teaching linear algebra) is based on $H^{-1}H = I$. To do it quickly, multiply $H^{-1}Hx = x$ as a sum of *columns times rows:*

$$\sum(\text{column } n \text{ of } H^{-1})(\text{row } n \text{ of } H)x = \sum(\text{new basis})(\text{new coefficient}) = x\,.$$

The new coefficients are in Hx (the analysis step). The reconstruction is $H^{-1}Hx$ (the synthesis step). These matrices are block Toeplitz and banded so both steps are fast.

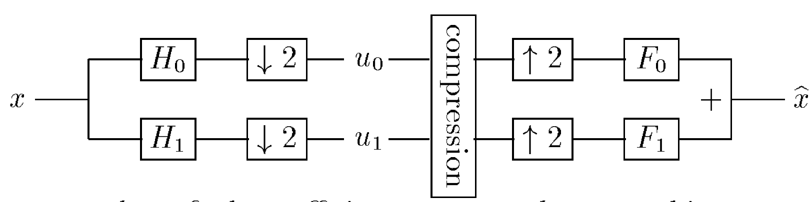
Instead of pure time-invariance (an ordinary Toeplitz matrix) we have block time-invariance. And the signal is separated into low and high frequencies, for better compression.

No self-respecting engineer would multiply those doubly infinite matrices! The one tool that is drilled into all filter designers is the Fourier transform. Instead of working with infinite block matrices, we can multiply the block frequency responses $F_b(\omega)$ and $H_b(\omega) = \sum (k\text{th block})\, e^{-ik\omega}$:

$$\begin{bmatrix} 3e^{-i\omega}+1 & -3e^{-i\omega}+1 \\ e^{-i\omega}+3 & -e^{-i\omega}+3 \end{bmatrix} \begin{bmatrix} -e^{-i\omega}+3 & 3e^{-i\omega}-1 \\ -e^{-i\omega}-3 & 3e^{-i\omega}+1 \end{bmatrix} = \begin{bmatrix} 16e^{-i\omega} & 0 \\ 0 & 16e^{-i\omega} \end{bmatrix}.$$

This illustrates the key point: *The determinant of $H_b(\omega)$ should be a monomial.* Here it is $16e^{-i\omega}$. That is the requirement for a polynomial inverse (since we divide cofactors by the determinant). Then H and H^{-1} are both banded as above.

To remove the 16, divide each entry in H_b and F_b by 4. The presence of $e^{-i\omega}$ indicates that the reconstructed $\hat{x} = F_b H_b x$ is a shift (a delay) of the input x. The delay becomes $\hat{x}(n) = x(n-3)$ when we line up the matrices correctly, and identify the synthesis bank:



analysis finds coefficients u_0, u_1 in the new basis. Those are compressed.

synthesis combines new basis times u_0 and u_1 (after compression).

Upsampling by $\uparrow 2$ inserts zeros between the components of u_0 and u_1. Then the full-length vectors are filtered by Toeplitz matrices F_0 and F_1. *Perfect reconstruction* means that the output (with no compression) is the input after ℓ delays: $\hat{x}(n) = x(n-\ell)$. One test is to multiply $F_b(\omega)H_b(\omega)$ as above. These are the *"polyphase matrices"* of multirate signal processing [V], with the columns reversed.

A direct test for perfect reconstruction follows the input through both channels, and adds. In the z-domain we want $\widehat{X}(z) = z^{-\ell}X(z)$. The operations of $(\downarrow 2)$ and $(\uparrow 2)$

produce the even part $\frac{1}{2}(H_i(z)X(z) + H_i(-z)X(-z))$. The synthesis filters multiply by $F_i(z)$. The alias term $X(-z)$ must disappear for perfect reconstruction:

$$F_0(z)H_0(-z) + F_1(z)H_1(-z) = 0 \qquad \text{(no aliasing)} \tag{4}$$

$$F_0(z)H_0(z) + F_1(z)H_1(z) = 2z^{-\ell} \qquad \text{(no distortion)}. \tag{5}$$

The algebra is in [V] and [S-N] and many earlier sources, including the Smith-Barnwell paper of 1976. Here we emphasize the conclusions:

1. The coefficients of F_0 and F_1 appear in the *columns* of F_b. They are $\frac{1}{4}(1,3,3,1)$ and $\frac{1}{4}(1,3,-3,-1)$.

2. Those columns are the new basis vectors, and $F_b(H_b x)$ expresses the output (the reconstruction $\widehat{x}$) as a combination of columns.

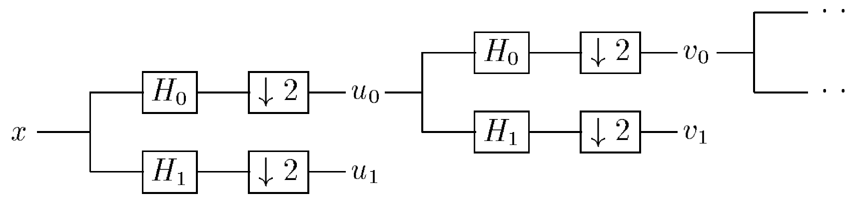
3. When u_0 and u_1 are quantized, $\widehat{x}$ uses a subset of the basis (with short binary coefficients). We hope this reconstruction is fast, and close to x.

The highpass synthesis filter F_1 comes from the lowpass analysis filter H_0. The rule is to alternate signs, so that $F_1(z) = -H_0(-z)$. The basis vector $1, 3, -3, -1$ comes from $-1, 3, 3, -1$.

Similarly, the lowpass $1, 3, 3, 1$ comes from the highpass H_1 by $f_0(n) = (-1)^n h_1(n)$. Then $F_0(z) = H_1(-z)$. These sign reversals mean that (4) is automatically satisfied (no aliasing). Now we have to see what properties make such a basis good or bad.

It will turn out that our $h_0 = \frac{1}{4}(-1, 3, 3, -1)$ is not so great. Its flaw appears when we iterate the basis change, aiming for more compression. *This iteration is the "wavelet idea"*—to replace u_0 by quarter-length vectors v_0 and v_1, and so on recursively. Four or five levels are typical. Our particular choice becomes unstable, and the components grow as the recursion continues. We need a test for stability.

A Tree of Filters: Approach to Wavelets



This tree displays *multiresolution.* The lowpass output u_0 is transformed again! After the second change of basis, the low-low part v_0 contains *averages of averages.* Ideally these have frequencies in the quarter-band $|\omega| \leq \frac{\pi}{4}$. A typical signal is mostly in v_0, partly in the coarse details v_1, and a little bit in the fine details u_1. The reconstruction goes back down the tree, using $(\uparrow 2)$ followed by synthesis filters f_0 and f_1. Those multiply coordinates v_0, v_1, u_1, by basis vectors to create signals $\widehat{v}_0$, $\widehat{v}_1$, $\widehat{u}_1$. Addition brings back $\widehat{x} = x$, with perfect reconstruction.

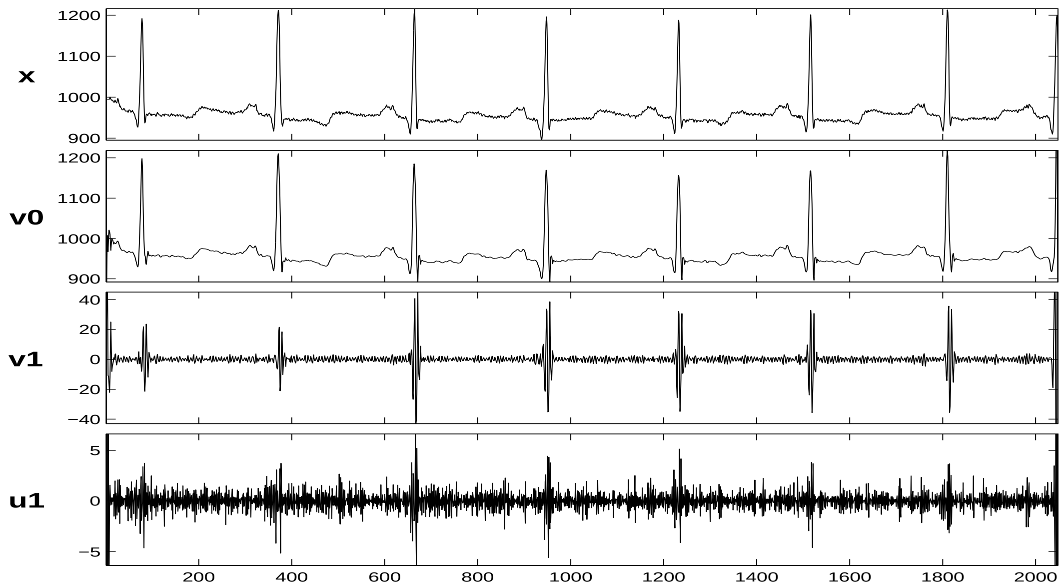


Figure 3: Two-level decomposition of a 1D-signal: note the scales.

This multiresolution pyramid (Figure 3 shows two steps) can be viewed in terms of *subspaces*. The even columns of F are double shifts of $f_0 = \frac{1}{4}(1,3,3,1)$ and they span a subspace V_0. The odd columns are double shifts of $f_1 = \frac{1}{4}(1,3,-3,-1)$ and they span a complementary "highpass" space W_0. The direct sum is $V_0 \oplus W_0 = \ell^2(\mathbf{Z})$. The components of u_0 and u_1 are the coefficients of x using these bases.

The second step of multiresolution splits V_0 into $V_1 \oplus W_1$. So the inverse pyramid, which reconstructs $\widehat{x}$, combines V_1 and W_1 and adds in the fine details from W_0. (We are using the Daubechies convention, with V_0 finer than V_1.) Because only *half* the basis is changed at each step, the whole multiresolution costs only $Nn(1+\frac{1}{2}+\frac{1}{4}+\cdots) = 2Nn$ operations. This is the "fast wavelet transform" in $O(n)$ steps.

To illustrate this multiresolution, follow two steps for the Haar filter bank. His lowpass and highpass coefficients are multiples of 1, 1 and 1, -1. The Haar filters take averages and differences. The basis vectors from the first step (notice the double shift!) are all the translates by two:

$$\text{In } V_0 : \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}, \ldots \quad \text{In } W_0 : \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ -1 \end{bmatrix}, \ldots$$

The second step changes only the basis for V_0. The sum and difference of those first two vectors yields

$$\text{In } V_1 : \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \ldots \quad \text{In } W_1 : \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \end{bmatrix}, \ldots$$

Now the dots represent all translates by four, and $V_1 \oplus W_1 \oplus W_0 = \ell^2(\mathbf{Z})$. This is an orthogonal sum for Haar; in general it is a direct sum. Haar's basis matrix F at this stage has 4×4 blocks:

$$F_{\text{Haar}} = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & -1 & 0 \\ 1 & -1 & 0 & 1 \\ 1 & -1 & 0 & -1 \end{bmatrix}.$$

Most early wavelets were orthogonal, $F = H^T$, but biorthogonality has become more common in practice: $F = H^{-1}$.

Note 3 For longer filters with more coefficients, there is a very useful factorization of the block Toeplitz matrices H_b and F_b. This is called *lifting*, and it breaks the steps into a sequence of shorter filters [Sw]. The calculations are done twice as fast and in place (possibly also in parallel).

The key is a factorization of the 2×2 polyphase matrices $H_p(z)$ and $F_p(z)$. Notice how the block Toeplitz matrix in our abstract can be represented by a matrix polynomial:

$$H_p(z) = \begin{bmatrix} a + cz^{-1} & b + dz^{-1} \\ d + bz^{-1} & -c - az^{-1} \end{bmatrix} = \begin{bmatrix} H_{0,\text{even}} & H_{0,\text{odd}} \\ H_{1,\text{even}} & H_{1,\text{odd}} \end{bmatrix}.$$

Lifting adds outputs from one channel (low = 0 or high = 1) to the other channel. The factors of H_p have short filters L and U:

$$\begin{bmatrix} 1 & 0 \\ L(z) & 1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 1 & u(z) \\ 0 & 1 \end{bmatrix}.$$

Image boundaries and irregular grids and integer-to-integer maps become much simpler for short filters. This lifting idea is developing rapidly.

Images are almost always processed by rows (with a one-dimensional filter bank) and then by columns (with the same filters). The Toeplitz matrices become tensor products. After downsampling we have *four quarter-length vectors*, u_{00} with most information and u_{01}, u_{10}, u_{11} with details. The low/low signal u_{00} goes back into the filter bank to produce four v_{ij}. We have a quad tree instead of a binary tree. The key to compression is *zero-tree coding* [Sh], which expects near-zero coefficients to appear in the same places at different scales.

Note 4 *Wavelet theory is usually about functions.* But the filters act on discrete coefficients, when a function is expanded in the old and new bases. Stéphane Mallat recognized that this action is by the *same filters H and F*. (Also Eero Simoncelli and Ingrid Daubechies and Yves Meyer saw this relation, that the fast pyramid also operates on the coefficients of functions.) The connection between functions and filters is in the *dilation equation* (refinement equation) and the *wavelet equation*:

$$\boxed{\phi(t) = \sum f_0(k)\phi(2t-k) \quad \textit{and} \quad w(t) = \sum f_1(k)\phi(2t-k)\,.} \tag{6}$$

The new basis contains the scaling functions $\phi(t-k)$ and the wavelets $w(t-k)$. The old basis is the translates of finer scale functions $\phi(2t-k)$. Equation (6) says that those bases are related *exactly* as they were for vectors, by the numbers $f_0(k)$ and

$f_1(k)$ in F_b. So the new coefficients must come by the same filters h_0, h_1 in the inverse matrix H_b. If $x(n)$ are the coefficients of $F(t)$ in one basis, then $u_0(n)$ and $u_1(n)$ give the coefficients in the new basis. The virtue of the new basis is to separate averages from details, for better compression.

To repeat: The continuous bases (old and new) are related by the same $f_0(k)$ and $f_1(k)$ as the discrete bases. So the expansion coefficients must be related by the same $h_0(k)$ and $h_1(k)$.

The $2t$ has the same source as $2n$ (downsampling). The functions ϕ and w are *local* (compact support) like the filters. The whole object of wavelet theory is to discover the properties of ϕ and w—which normally have no simple formulas. Those properties must be hidden in the filter coefficients. We choose the filters and they act on vectors of coefficients—allowing us to pretend that the input was really a function.

Note 5 It is not so easy to solve (6) for the basis function $\phi(t)$. Its Fourier transform is an infinite product from the infinite iteration (with rescaling):

$$\hat{\phi}(\omega) = \prod_{j=1}^{\infty} F_0\left(\omega/2^j\right).$$

This scaling function is central to the subject, but all computations are based on using the coefficients of the filters.

Exceptionally, $f_0 = \frac{1}{4}(1,3,3,1)$ does lead to a simple $\phi(t)$. It is the convolution $B * B * B$ of three box functions. The unit box on $[0,1]$ solves the dilation equation $B(t) = B(2t) + B(2t-1) =$ sum of two half-boxes. Its filter coefficients are $(1,1)$ and its frequency response is $F_0(\omega) = (1 + e^{-i\omega})$. Applying this Haar filter three times leads to $(1,3,3,1)$ in the filter. That produces $(1+e^{-i\omega})^3$ in the frequency response, and $B * B * B$ in the scaling function.

For any filter, each factor $1 + e^{-i\omega}$ means an additional zero at $\omega = \pi$ in the frequency response. The scaling function is convolved with the box $B(t)$, which makes it one order smoother. The number p of "zeros at π" in $F_0(\omega)$ is critical to the good properties of $\phi(t)$. The celebrated Daubechies filters have the largest possible p while retaining an *orthogonal basis*. They have $2p$ coefficients. The spline filters $(1,1)$ and $\frac{1}{2}(1,2,1)$ and $\frac{1}{4}(1,3,3,1)$ have the largest p without any restriction to orthogonality. F_b and H_b are orthogonal matrices for Daubechies, but not for $(1,3,3,1)$. Our basis functions $\phi(t-k)$ and $w(t-k)$ are piecewise polynomial splines of degree $p-1$, and not orthogonal, so $\widetilde{\phi}$ must be different from ϕ.

Note 6 We are describing the classical construction of wavelet bases. The more general idea of *wavelet packets* is to allow iteration of $(\downarrow 2)H_1$ as well as $(\downarrow 2)H_0$—the best basis [W] is adapted to the signal. And good bases come from many other sources too.

One neat construction by Malvar [M] and Coifman-Meyer [C-M] uses a window with cosines inside it. The basis contains $h(t)\cos k\pi t$ in the continuous case. With a careful choice of h, the translates of those modulated windows give an orthogonal basis for L^2. The windows $h(t)$ and $h(n)$ for audio signals are amazingly long, and overlap their translates by a lot. The windows can also have unequal lengths, escaping from the regular mesh that is associated with wavelets.

With no overlap (a window box $h=1$) we are back to the discrete cosine transform. Each basis function has M components $\cos\left((k+\frac{1}{2})(n+\frac{1}{2})\frac{\pi}{M}\right)$, and frequently $M=8$. These M basis vectors are the columns of the DCT matrix C_{kn}. The change of basis matrix is *block diagonal* (with C in each block). This is extremely fast, but compression produces a severe blocking artifact in the image.

Another part of this subject, different from compression, is signal *identification*: to find the important features of $x(n)$. The Fourier transform displays the dominant frequencies. The spectrum hidden in $x(n)$ becomes clear in $X(\omega)$. But discontinuities and "chirp" signals with time-varying frequencies are not so clear. A wavelet basis may be better. For identification and denoising, we refer to Mallat [Ma] and Donoho [Do] among many others. And the *enhancement* of an image is a further problem, needing a nonlinear operator. That theory goes beyond the basis changes of this paper.

Properties of Wavelets

We have moved from the practical problem (compressing images) to a mathematical problem. The properties of the basis functions $\phi(t-k)$ and $w(t-k)$ must be determined from the filter coefficients. We need to know this connection, in choosing a good basis. Here are desirable properties:

Symmetry: $\phi(t)$ is symmetric about $t = \frac{N}{2}$ when the filter is symmetric: $f_0(k) = f_0(N-k)$. A highly compressed image uses only a few basis vectors, and it looks a little strange without symmetry.

Orthogonality: The basis functions $\phi(t-k)$ and $w(t-k)$ are orthogonal when the 2×2 matrix $F(\omega)$ is unitary for every ω. Then F_b and H_b are orthogonal block Toeplitz matrices.

Regrettably, there is a conflict between orthogonality and symmetry. Suppose $a, b, b, a, 0, 0$ is orthogonal to its double shift $0, 0, a, b, b, a$, in the rows of H_b. Then $2ab = 0$ and either a or b must be zero. Symmetric orthogonal filters can have only *two* coefficients and we are back to Haar's simple box function. Most compression algorithms maintain symmetry and do without orthogonality. Or they use more filters.

Biorthogonality: This is achievable with symmetry, as the example shows. $F_b(\omega)$ and $H_b(\omega)$ are inverses; the rows of H_b are biorthogonal to the columns of F_b (to produce $HF = I$). The filter bank gives perfect reconstruction. Then the functions $\phi(t-k)$, $w(t-k)$ are biorthogonal to $\tilde{\phi}(t-k)$, $\tilde{w}(t-k)$. Those dual functions with tildes come from equation (6) with h in place of f.

Stability: Does the dilation equation (6) yield good functions $\tilde{\phi}(t-k)$? If so, they come from the infinite cascade $\tilde{\phi}^{i+1}(t) = \sum h_0(k)\tilde{\phi}^i(2t-k)$. This corresponds to an infinite filter tree and a complete multiresolution, iterating $(\downarrow 2)H$ and the change of basis forever.

If the iteration is stable, we get wavelets at all scales. This step to infinity was the key contribution in the 1980's. (The finite filter tree was invented ten years earlier by electrical engineers.) Condition E for stability applies to the double-shift matrix $T_{jk} = p(2j-k)$ coming from $P(\omega) = |H_0(\omega)|^2$, normalized by $P(0) = 2$. *All eigenvalues of T must have $|\lambda| < 1$ except for a simple eigenvalue $\lambda = 1$* (see [S-N]).

Smoothness: $\tilde{\phi}(t)$ and $\tilde{w}(t)$ have s derivatives in L_2 if and only if $|\lambda_{\max}| < 4^{-s}$. Here, $\lambda_{\max}$ is the largest eigenvalue of T after excluding the special eigenvalues $1, \frac{1}{2}, \ldots, \left(\frac{1}{2}\right)^{2p-1}$.

These tests for stability and smoothness are quick to apply. For $h_0 = (-1, 3, 3, -1)$ the entries in T are $p = (1, -6, 3, 20, 3, -6, 1)/8$, coming from $|H_0(\omega)|^2$. The crucial double-shift submatrix of T is

$$\frac{1}{8}\begin{bmatrix} -6 & 1 & & & \\ 20 & 3 & -6 & 1 & \\ -6 & 3 & 20 & 3 & -6 \\ & 1 & -6 & 3 & 20 \\ & & & 1 & -6 \end{bmatrix} \quad \text{with } \lambda = 1, \tfrac{1}{2}, -1, -.92 \text{ and } 2.17\,.$$

The special eigenvalues $\lambda = 1, \frac{1}{2}$ are produced by the zero of $H_0(\omega)$ at $\omega = \pi$ [S-N]. *That big eigenvalue $\lambda_{\max} = 2.17$ spells disaster.* Iterating this filter is unstable, although the other filter based on $(1, 3, 3, 1)$ and $(1 + e^{-i\omega})^3$ is very stable.

For a better pair, move one of these good factors $(1 + e^{-i\omega})$ to the bad filter. Then both filters have two zeros at $\omega = \pi$:

$$\begin{aligned} f_0 &= (1, 2, 1) && \text{corresponds to } F(\omega) = (1 + e^{-i\omega})^2 \text{ and} \\ h_0 &= (-1, 2, 6, 2, -1) && H(\omega) = (1 + e^{-i\omega})(-1 + 3e^{-i\omega} + 3e^{-2i\omega} - e^{-3i\omega}). \end{aligned}$$

Moving a zero adds one order of smoothness to $\tilde{\phi}(t)$ and subtracts one order from $\phi(t)$. It divides $\lambda_{\max}$ by 4, so the new $\lambda_{\max}$ is .54. This 5/3 pair is not bad. For more stability we can move another zero at π from $F(\omega)$ to $H(\omega)$:

$$\begin{aligned} f_0 &= (1, 1) && \text{corresponds to Haar's } 1 + e^{-i\omega} \text{ and} \\ h_0 &= (-1, 1, 8, 8, 1, -1) && (1 + e^{-i\omega})^2(-1 + 3e^{-i\omega} + 3e^{-2i\omega} - e^{-3i\omega}). \end{aligned}$$

With these simple binary coefficients, the multiplications Hx and Fx require only shifts (no roundoff errors). Nguyen is applying filters of this kind to medical images and legal documents, when loss of information is not allowed. The image in Figure 4 is somewhat crude, however, because $f_0 = (1,1)$ is so short and the box function $\phi(t) = B(t)$ is not smooth. By exchanging f_0 and f_1 with h_0 and h_1, we get a better and smoother basis from the 2/6 pair.

Figure 4: A crude image using Haar for synthesis (6/2 pair).

Artifacts from Compression

It is usual to measure the L^2 norm $\| x - \widehat{x} \|$, between input and output. This is easy to compute, but nobody believes in it! That error measure misses the features (like edges) that our eyes consider important.

One such feature is *masking*. In audio, a loud signal at frequency ω will drown a quiet signal at a nearby frequency. It is literally impossible to hear that second note. Similarly the human visual system, the eye and the brain, will govern what we see (or what we think we see). We close this paper with examples of undesirable artifacts from compression. Sometimes we know how to avoid a filter (a change of basis) that produces the artifact.

1. **Blocking:** The lowpass synthesis filter F_0 is too short. The scaling function $\phi(t)$ is rough, and we see its jumps much too clearly after compression (Figure 4).

2. **Ringing:** The highpass synthesis filter F_1 is too long (9 coefficients in Figure 5). Ringing is significant around sharp edges, when compression deletes the high frequencies.

3. **Checkerboarding:** The analysis filters H_0 and H_1 need to separate low and high frequencies. Especially we need $H_0(\pi) = 0$ and $H_1(0) = 0$, to avoid leakage of high and low frequencies into the wrong channel. This leakage produced the checkerboard artifact in Figure 6.

Overall, we want to preserve texture (high ω) by short functions. A smooth background (low ω) needs long functions. This is exactly what the wavelet transform promotes! The iteration of the lowpass filter automatically lengthens the basis vectors, while the highpass filter stops short (no iteration). In the language of waves, we need to observe a long wave for a long time. The time-frequency plane is (approximately) partitioned into the rectangles in Figure 7. Heisenberg's Uncertainty Principle forbids compact support in both time and frequency.

The competition between cosine transforms and wavelet transforms is certainly not over (and video is a battleground). For day-to-day production and low compression ratios, the Fourier methods are fast and practical. Wavelets can meet a more demanding challenge. The extensions and combinations such as Gen LOT (described in [S-N]) are giving image compression of remarkable quality.

Figure 5: Ringing around edges from a long synthesis filter.

Figure 6: Checkerboarding: The DC term leaks into the highpass channel.

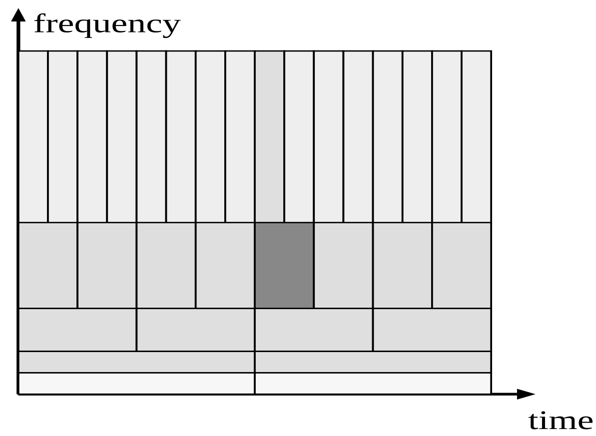


Figure 7: Wavelets yield time-frequency rectangles: Constant area $(\Delta t)(\Delta\omega)$.

References

[C-M] R. Coifman and Y. Meyer, Remarques sur l'analyse de Fourier à fenêtre, *C. R. Acad. Sci. Paris* **312** (1991), 259–261.

[D] Ingrid Daubechies, Ten Lectures on Wavelets, *SIAM* (1992).

[Do] D. Donoho, De-noising by soft thresholding, *IEEE Trans. Inf. Th.* **41** (1995), 613–627.

[M] H. S. Malvar, Signal Processing with Lapped Transforms, *Artech House* (1992).

[Ma] Stéphane Mallat, A theory for multiresolution signal decomposition: The wavelet representation, *IEEE Trans. PAMI II* (1989), 674-693 (see also *Trans. AMS* **315** (1989), 69–87).

[Me] Yves Meyer, Wavelets: Algorithms and Applications, *SIAM* (1993).

[R-V] Olivier Rioul and Martin Vetterli, Wavelets and signal processing, *IEEE Signal Proc. Mag.* (1991), 14–38.

[S-B] M. J. T. Smith and T. Barnwell, Exact reconstruction techniques for tree-structured subband coders, *IEEE Trans. ASSP* **34** (1986), 434–441.

[Sh] J. M. Shapiro, Embedded image coding using zerotrees of wavelet coefficients , *IEEE Trans. Signal Proc.* **41** (1993), 3445–3462.

[S-N] Gilbert Strang and Truong Nguyen, Wavelets and Filter Banks, *Wellesley-Cambridge Press* (1996).

[Sw] Wim Sweldens, The lifting scheme, *Appl. Comput. Harmonic Analysis* **3** (1996), 186–200; see also I. Daubechies and Wim Sweldens, Factoring wavelet transforms into lifting steps, to appear.

[U] M. Unser, Approximation power of biorthogonal wavelet expansions, *IEEE Trans. Signal Proc.* **44** (1996), 519–527; see also M. Unser and I. Daubechies, On the approximation power of convolution-based least squares versus interpolation, *IEEE Trans. Signal Proc.*, to appear.

[V] P. P. Vaidyanathan, Multirate Systems and Filter Banks, *Prentice-Hall* (1992).

[V-K] M. Vetterli and J. Kovacevic, Wavelets and Subband Coding, *Prentice-Hall* (1995).

[Wi] M. V. Wickerhauser, Adapted Wavelet Analysis from Theory to Software, *AK Peters* (1994).