# Chapter 8

# Finite Length Signals

## 8.1 Circular Convolution and the DFT

Our signals have so far had infinite length. We expected $x(n)$ and $f(t)$ to be defined forever — for all integers $n$ and all real numbers $t$. The data streams for audio signals are so long that this model is very reasonable. Other applications (like image processing) have *data streams of a definite length* $L$. Then the finiteness of the signal $x(0), \ldots, x(L-1)$ must be taken into account.

The immediate problem is filtering a finite signal. The computation of $\sum h(k)\,x(n-k)$ may ask for $x(-1)$ which is not defined. The purpose of extension is to define it. The difficulty is not in the middle of the signal (unless the filter is IIR). The problem is near the ends. One possibility is to *change to a one-sided boundary filter.* The end values of $x$ are processed without crossing the boundary. The other possibility is to *extend the signal beyond the boundary.*

We concentrate first on extending the signal, to define $x(n)$ for every $n$. Filtering this extension is equivalent to one particular set of boundary filters. Section 8.5 discusses boundary filters in general.

A finite-length filter bank uses $L$ inputs to produce $L$ outputs. An $L$ by $L$ matrix must be present. It is conceptually easiest to extend to an infinite signal, followed by ordinary time-invariant filtering. The question is how to extend, and there are many possibilities:

1. Extend by zeros *(zero-padding)* *(not studied)*
2. Extend by periodicity *(wraparound)* *(this section)*
3. Extend by reflection *(symmetric extension)* *(next section).*

These three methods are applied in Figure 8.1. You notice immediately a very important point:

Zero-padding and wraparound generally introduce a *jump in the function.*

Reflection generally introduces a *jump in the first derivative.*

Section 10.1 shows the three methods applied to a filtered image. The sky at the top is degraded by zero-padding. Wraparound is better but a thin error layer is visible. Symmetric extension is the best.

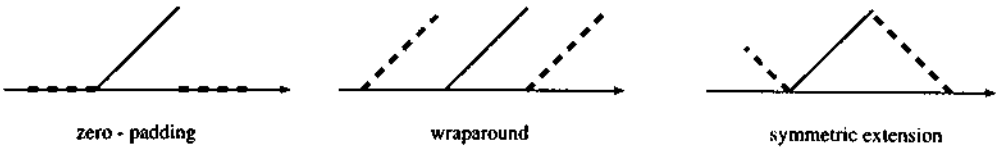



**Figure 8.1:** Three extension methods: zero-padding, wraparound, reflection.

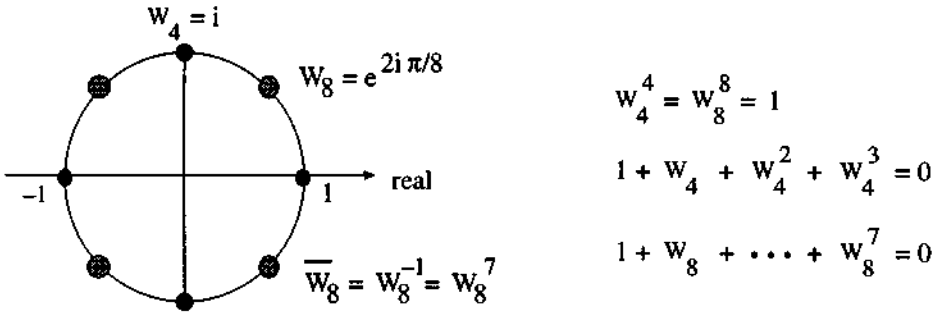



**Figure 8.2:** The fourth roots of 1 and the eighth roots of 1. They add to zero.

We do not pursue zero-padding in the text. It chops off the infinite Toeplitz matrix $\boldsymbol{H}$, with no adjustment at the boundary. Wraparound is good if the original problem is genuinely periodic or close to it. The filter matrix becomes a *circulant*. All indices are interpreted "*modulo* $L$". The $k$th diagonal of the matrix contains $\boldsymbol{h}(k)$, and it wraps around to continue as the $(k - L)$th diagonal.

*All circulant matrices are diagonalized by the DFT.* Therefore multiplications and inversions are very fast. Convolution $\boldsymbol{h} * \boldsymbol{x}$ is still multiplication of transforms $H(z)X(z)$ (with $z^L = 1$). In frequency this is the component-by-component multiplication $\widehat{\boldsymbol{h}}(k)\widehat{\boldsymbol{x}}(k)$. The inverse matrix exists if all $\widehat{\boldsymbol{h}}(k) \neq 0$, and it is also a circulant.

These circulant matrices are so simple and useful that we take time out to explain them (and also the DFT). Together they are a perfect match. Circulants have constant diagonals in the time domain. They are built out of (circular) shift matrices. The transform to frequency domain is executed at high speed by the Fast Fourier Transform, and we will put the FFT in a matrix form. The Fourier matrix is a product of very sparse matrices.

The next section returns to the critical question of symmetric extension. *Do we want period* $2(L - 1)$ *or* $2L$? The answer depends on the length of the signal and the length of the filter! We hope those details will be useful.

**Note 1** Other extensions are possible and interesting. Zero-padding could change to constant-padding (almost as simple and more accurate). A more careful extension could have constant slope. Section 8.5 shows how to fit a polynomial and extrapolate. In all cases we only need to provide the filter with a few values (at most $N$ values) beyond the endpoints 0 and $L - 1$. Symmetric extension finds those values entirely by data addressing, without extra computations.

When it helps the understanding, we are free to assume that the periodic signals exist at all

times $-\infty < n < \infty$. The essential problem is to implement a filter bank on a finite length signal, *without expanding that length.* Circular extension allows a probable jump at the endpoints.

### Properties of the Circular Shift

The building blocks for time-invariant filters are the shift matrices $S$ and $S^{-1}$. All components of a vector are delayed one step (by $S$) and advanced one step (by $S^{-1}$). The building blocks for circulants (*cyclic filters*) are cyclic shifts. Every component is delayed by $S_L$, except the last component which comes around to be the first component.

The 4 by 4 delay takes components 0, 1, 2, 3 *forward* into positions 1, 2, 3, 4 – except that 4 becomes 0. Algebraically, we work "modulo 4". Intuitively, our problem has period 4, so that $x(4)$ is the same as $x(0)$. Here is $S_4$ multiplying the vector $x$:

$$S_4 x = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix} = \begin{bmatrix} x(3) \\ x(0) \\ x(1) \\ x(2) \end{bmatrix}. \tag{8.1}$$

$S_4$ is a *cyclic permutation matrix.* It has the same rows as $I$, in a different order. Those rows are mutually orthogonal and they are unit vectors – so $S_4$ is an orthogonal matrix. The shift matrices have very special properties:

1. The inverse equals the transpose. The $L$th power of $S_L$ is $I$.

2. The eigenvalues of $S_L$ are the $L$ roots of 1 in the complex plane (on the unit circle in Figure 8.2). They are the powers of the number $W = e^{2\pi i/L}$. They are also the powers of $\overline{W} = W^{-1} = e^{-2\pi i/L}$.

   It is convenient to list the eigenvalues in the clockwise order $1, \overline{W}, \overline{W}^2, \ldots$, because $\overline{W}^k = e^{-2\pi ik/L}$ fits into the Discrete Fourier Transform.

3. Corresponding to $\lambda = \overline{W}^k$ is the eigenvector $x = (1, W^k, W^{2k}, \ldots)$. These eigenvectors are the columns of the $L$ by $L$ ***Fourier matrix* = *DFT matrix*** $= F_L$. The entries of $F_L$ are $W^{nk} = e^{2\pi ink/L}$. This matrix diagonalizes the shift (and all circulants):

$$F_L^{-1} S_L F_L = D_L = \text{diag}\,(1, \overline{W}, \overline{W}^2, \ldots). \tag{8.2}$$

We can quickly establish Properties **1–3**, using $S_4$ as the example to work with:

$$S_4^T = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \quad \text{times} \quad S_4 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad \text{is} \quad S_4^T S_4 = I.$$

An advance times a delay is the identity. This is true and useful on the whole line, with times $-\infty < n < \infty$. It is true here *on the circle.* The times are $0, \ldots, L-1$, and they repeat. On an infinite half-line, $S_+^T S_+ = I$ is true but $S_+ S_+^T = I$ is not true! An advance $S_+^T$ will wipe out the first component $x(0)$, and a delay $S_+$ cannot bring it back. Real boundaries are distinctly harder than periodic boundaries.

The fourth power of $S_4$ is the identity matrix. When we shift forward four times, the cycle completes – we come back to the start. Always $(S_L)^L = I$. Here are $S_4$, $S_4^2$, $S_4^3 = S_4^{-1}$, and $S_4^4 = I$:

$$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \quad \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}, \quad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Properties 2 and 3 are about eigenvalues and eigenvectors. To find the eigenvalues, we normally solve $\det(S - \lambda I) = 0$. For the $L$ by $L$ shift matrix, this determinant (see Problem 1) yields the equation $\lambda^L = 1$. The eigenvalues are roots of unity, in agreement with the fact that $S^L = I$.

$$\textit{The determinant of} \begin{bmatrix} -\lambda & 0 & 0 & 1 \\ 1 & -\lambda & 0 & 0 \\ 0 & 1 & -\lambda & 0 \\ 0 & 0 & 1 & -\lambda \end{bmatrix} \textit{ is } \lambda^4 - 1.$$

The four eigenvalues are $1, -i, -1, i$. Those fourth roots of 1 are equally spaced around the unit circle, at angles $\frac{8\pi}{4}, \frac{6\pi}{4}, \frac{4\pi}{4}, \frac{2\pi}{4}$. In general the $L$th roots are at multiples of the angle $\frac{2\pi}{L}$. Going clockwise around the circle, we have the $L$ powers of $\overline{W} = e^{-2\pi i/L}$ in Figure 8.2.

For the circular shift, we can announce the eigenvectors at the same time. The eigenvector for $\lambda = 1$ is just $x = (1, 1, \ldots, 1)$. The shift leaves it unchanged: $Sx = 1x$. The second eigenvalue $\overline{W}$ has eigenvector $x = (1, W, W^2, \ldots, W^{L-1})$. Shifting brings $W^{L-1} = W^{-1}$ to the top:

$$\textit{Circular shift of} \begin{bmatrix} 1 \\ W \\ W^2 \\ \vdots \\ W^{L-1} \end{bmatrix} \text{ is } \begin{bmatrix} W^{L-1} \\ 1 \\ W \\ \vdots \\ W^{L-2} \end{bmatrix} = W^{-1}x = \overline{W}x.$$

Thus $Sx = \overline{W}x$ as required. Similarly the $k$th eigenvalue $\lambda = W^{-k} = \overline{W}^k$ has the eigenvector $x_k = (1, W^k, W^{2k}, \ldots, W^{(L-1)k})$. Here is $Sx = \lambda x$:

$$\textit{Circular shift of} \begin{bmatrix} 1 \\ W^k \\ W^{2k} \\ \vdots \end{bmatrix} \text{ is } \begin{bmatrix} W^{(L-1)k} \\ 1 \\ W^k \\ \vdots \end{bmatrix} = W^{-k}x = \overline{W}^k x.$$

We are making liberal use of the fact that $W^L = 1$. Now summarize:

**Diagonalization of $S$ by $F$.** The eigenvectors of the shift $S$ are the columns of the Fourier matrix $F$. The matrix $F^{-1}SF$ is diagonal:

$$F^{-1}SF = D \quad \text{and} \quad SF = FD \quad \text{and} \quad S = FDF^{-1}. \tag{8.3}$$

$D$ is the diagonal matrix of eigenvalues $1, \overline{W}, \overline{W}^2, \ldots$. For $L = 4$ these matrices are displayed

in the multiplication $S_4F_4 = F_4D_4$, whose columns are $Sx = \lambda x$:

$$\underset{S}{\begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}} \underset{F}{\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & i^2 & i^3 \\ 1 & i^2 & i^4 & i^6 \\ 1 & i^3 & i^6 & i^9 \end{bmatrix}} = \underset{F}{\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & i^2 & i^3 \\ 1 & i^2 & i^4 & i^6 \\ 1 & i^3 & i^6 & i^9 \end{bmatrix}} \underset{D}{\begin{bmatrix} 1 & & & \\ & \bar{i} & & \\ & & \bar{i}^2 & \\ & & & \bar{i}^3 \end{bmatrix}} \tag{8.4}$$

The columns of $F$ are ***orthogonal eigenvectors*** of $S$, with length $\sqrt{L}$.

Why are the eigenvectors orthogonal? The answer from linear algebra is: every real matrix with the property $SS^T = S^TS$ has orthogonal eigenvectors. These are the "normal matrices." In our case $SS^T = S^TS = I$.

A second answer is to verify orthogonality directly. The eigenvectors of $S$ (the columns of $F$) are complex even though $S$ is real. Permutations like $S$ generally have complex eigenvalues and eigenvectors. Take the inner product:

$$\bar{x}_k^T x_\ell = \begin{bmatrix} 1 & \overline{W}^k & \overline{W}^{2k} & \cdots \end{bmatrix} \begin{bmatrix} 1 \\ W^\ell \\ W^{2\ell} \\ \vdots \end{bmatrix} = 1 + z + z^2 + \cdots + z^{L-1} = 0.$$

Here $z = \overline{W}^k W^\ell$. The sum is $(1 - z^L)/(1 - z)$. This is zero because $z^L = 1$ (remember that $W^L = 1$). So the eigenvectors of $S$ are orthogonal to each other: $\overline{F}F = LI$. We are ready to move forward to circulants, via the DFT.

### Discrete Fourier Transform and Circulants

An ordinary filter $H$ is a combination of powers of the (infinite) shift matrix $S$. A circular filter $H_L$ is a combination of powers (with $N < L$) of the finite shift matrix $S_L$:

Ordinary filter $H = \sum_{-\infty}^{\infty} h(k)S^k$ Circular filter $H_L = \sum_{n=0}^{N} h(n)(S_L)^n$.

$H$ is a Toeplitz matrix and $H_L$ is a ***circulant matrix***. Multiplication by $H$ is convolution. Multiplication by $H_L$ is circular convolution.

For $L = 4$, the four powers of $S_4$ were displayed earlier. They combine into a typical $4 \times 4$ circulant matrix $H_4 = h(0)I + h(1)S_4 + h(2)S_4^2 + h(3)S_4^3$:

$$H_4 = \begin{bmatrix} h(0) & h(3) & h(2) & h(1) \\ h(1) & h(0) & h(3) & h(2) \\ h(2) & h(1) & h(0) & h(3) \\ h(3) & h(2) & h(1) & h(0) \end{bmatrix}. \tag{8.5}$$

Notice how the diagonals wrap around. We still have $h(n)$ along diagonal $n$. In this circular world 3 is the same as $-1$, so $h(3)$ is also on diagonal $-1$. With periodicity we work modulo $L$. Two numbers are the same ($m \equiv n \bmod L$) when $m - n$ is a multiple of $L$.

Usually $N$ is much smaller than $L$. We have a band of nonzeros below the diagonal, wrapping around to the upper right corner (as in $S$). Otherwise $H_L$ is zero. We give the properties of

$\boldsymbol{H}_L$ and then the applications. The powers of $\boldsymbol{S}$ all have the same eigenvectors as $\boldsymbol{S}$, so the key to a circulant is its *eigenvectors* and *eigenvalues*.

**Diagonalization of $\boldsymbol{H}_L$ by $\boldsymbol{F}_L$.** The eigenvectors of $\boldsymbol{H}_L$ are the columns of the Fourier matrix $\boldsymbol{F}_L$. The eigenvalues appear in the discrete Fourier transform $(\widehat{h}(0), \widehat{h}(1), \widehat{h}(2), \ldots)$ of the coefficients $(h(0), h(1), h(2), \ldots)$:

$$\boldsymbol{F}_L^{-1}\boldsymbol{H}_L\boldsymbol{F}_L = \widehat{\boldsymbol{H}}_L = \operatorname{diag}\big(\widehat{h}(0), \widehat{h}(1), \ldots, \widehat{h}(L-1)\big). \tag{8.6}$$

**Proof.** Every time we multiply $\boldsymbol{x} = (1, W^k, W^{2k}, \ldots)$ by the shift matrix $\boldsymbol{S}$, we get one more power of $\lambda = \overline{W}^k$. A combination of powers $\boldsymbol{S}^n$ produces the same combination of powers $\overline{W}^{kn}$:

$$\boldsymbol{H}_L\boldsymbol{x} = \Big(\sum_0^N h(n)\boldsymbol{S}^n\Big)\boldsymbol{x} = \Big(\sum_0^N h(n)\overline{W}^{kn}\Big)\boldsymbol{x} = \widehat{h}(k)\boldsymbol{x}. \tag{8.7}$$

The eigenvalues of $\boldsymbol{H}_L$ are exactly the components $\widehat{h}(k)$ of the DFT:

| | | | |
|---|---|---|---|
| The eigenvalue for | $\boldsymbol{x}_0 = (1, 1, \ldots)$ | is | $h(0) + h(1) + \cdots = \widehat{h}(0)$ |
| The eigenvalue for | $\boldsymbol{x}_k = (1, W^k, \ldots)$ | is | $h(0) + h(1)\overline{W}^k + \cdots = \widehat{h}(k)$. |

The matrix form $\boldsymbol{H}\boldsymbol{F} = \boldsymbol{F}\widehat{\boldsymbol{H}}$ just gives these $L$ equations all at once.

**Example 8.1.** Choose $h(0) = 4$, $h(1) = 1$ and $h(2) = 1$. Take $L = 3$:

$$\boldsymbol{H} = \begin{bmatrix} 4 & 1 & 1 \\ 1 & 4 & 1 \\ 1 & 1 & 4 \end{bmatrix} \quad \text{has eigenvalues} \quad \begin{aligned} \widehat{h}(0) &= 4 + 1 + 1 = 6 \\ \widehat{h}(1) &= 4 + \overline{W} + \overline{W}^2 = 3 \\ \widehat{h}(2) &= 4 + \overline{W}^2 + \overline{W}^4 = 3. \end{aligned}$$

The eigenvalues are real because $\boldsymbol{H}$ is symmetric. The eigenvectors are orthogonal (of course). The first eigenvector is $(1, 1, 1)$. The other eigenvectors are $(1, e^{2\pi i/3}, e^{4\pi i/3})$ and its complex conjugate. We could change to real eigenvectors $(1, \cos\frac{2\pi}{3}, \cos\frac{4\pi}{3})$ and $(0, \sin\frac{2\pi}{3}, \sin\frac{4\pi}{3})$ if we wanted to (both with eigenvalue 3). This is the option in the real case of using sines and cosines. Everywhere we use $W^3 = 1$ and $1 + W + W^2 = 0$.

**Example 8.2.** Choose $h(0) = 4$, $h(1) = 1$, $h(2) = 0$, $h(3) = 1$ and $L = 4$:

$$\boldsymbol{H} = \begin{bmatrix} 4 & 1 & 0 & 1 \\ 1 & 4 & 1 & 0 \\ 0 & 1 & 4 & 1 \\ 1 & 0 & 1 & 4 \end{bmatrix} \quad \text{has eigenvalues} \quad \begin{aligned} \widehat{h}(0) &= 4 + 1 + 0 + 1 = 6 \\ \widehat{h}(1) &= 4 + i^3 + 0 + i = 4 \\ \widehat{h}(2) &= 4 + i^2 + 0 + i^6 = 2 \\ \widehat{h}(3) &= 4 + i + 0 + i^3 = 4. \end{aligned}$$

Again $\boldsymbol{H}$ is symmetric and the eigenvalues are real. Note the eigenvector $(1, -1, 1, -1)$ corresponding to $\widehat{h}(2) = 2$. The DFT matrix diagonalizes $\boldsymbol{H}_L$:

| | | | |
|---|---|---|---|
| For infinite signals | $\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x}$ | becomes | $Y(z) = H(z)X(z)$ |
| For length $L$ signals | $\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x}$ | becomes | $\widehat{\boldsymbol{y}} = \widehat{\boldsymbol{H}}\widehat{\boldsymbol{x}}$. |

$\boldsymbol{H}\boldsymbol{x}$ is a matrix-vector multiplication. $\widehat{\boldsymbol{H}}\widehat{\boldsymbol{x}}$ is a pointwise multiplication.

We now explain the finite Fourier transform $\widehat{\boldsymbol{x}}$ and circular convolution. The vector $\boldsymbol{x}$ has length $L$ and so does $\widehat{\boldsymbol{x}}$:

**Definition 8.1** *The Discrete Fourier Transform of $x$ is $\widehat{x} = \overline{F}_L x$. Its components are*

$$\widehat{x}(k) = \sum_{n=0}^{L-1} x(n)\overline{W}^{nk} = \sum_{n=0}^{L-1} x(n)e^{-2\pi ink/L}. \tag{8.8}$$

The index $n$ corresponds to time and $k$ corresponds to frequency:

$$\text{DFT}[x(n)] = X(k) = \widehat{x}(k) \qquad \text{IDFT}[\widehat{x}(k)] = \text{IDFT}[X(k)] = x(n).$$

The inverse transform is $x = \overline{F}_L^{-1}\widehat{x} = \frac{1}{L}F_L x$. Its components are

$$x(n) = \frac{1}{L}\sum_{k=0}^{L-1} \widehat{x}(k)e^{2\pi ink/L}. \tag{8.9}$$

The DFT analyzes the signal into pure frequencies. The IDFT synthesizes it again. The exponent is $-2\pi ink/L$ in (8.8) and $2\pi ink/L$ in (8.9), just as the Fourier integral in continuous time has $e^{-i\omega t}$ and $e^{i\omega t}$. Do not forget why the complex conjugate gives the inverse! It is because the columns of $F_L = F_L^T$ are orthogonal. The inner products give a multiple of the identity matrix:

$$\overline{F}_L F_L = LI \quad \text{and} \quad F_L\overline{F}_L = LI \quad \text{and} \quad \overline{F}_L^{-1} = \tfrac{1}{L}F_L. \tag{8.10}$$

Dividing by $L$ gives the inverse, like dividing by $2\pi$ in continuous time. The transform preserves energy (apart from this factor $L$). This is the discrete version of Parseval's theorem $\int |x(t)|^2 dt = \frac{1}{2\pi}\int |X(\omega)|^2 d\omega$ :

$$\textit{Discrete Parseval theorem:} \quad \|x\|^2 = \tfrac{1}{L}\|\widehat{x}\|^2. \tag{8.11}$$

This is $\sum |x(n)|^2 = \frac{1}{L}\sum |\widehat{x}(k)|^2$. It is based on orthogonality of the columns:

$$\langle x, x\rangle = \langle \tfrac{1}{L}F_L\widehat{x}, \tfrac{1}{L}F_L\widehat{x}\rangle = \tfrac{1}{L}\langle \widehat{x}, \widehat{x}\rangle \quad \text{because} \quad \overline{F}_L^T F_L = LI.$$

## The Circular Convolution Rule

*Diagonalization by the Fourier matrix* is true for all circulants. Therefore any multiplication $Hx$ can be done two ways. A direct matrix multiplication is a "circular convolution". The indirect way is multiplication by $F_L^{-1}$, then the diagonal matrix $D_L$, then $F_L$. This is a transform to the frequency domain, where $H$ is diagonalized. This ***convolution rule*** underlies the whole theory of filters: convolution in time $\Leftrightarrow$ multiplication after DFT.

The circular filter $y = Hx$ becomes $\widehat{y} = \widehat{H}\widehat{x}$ in frequency. This comes from the diagonalization $F^{-1}HF = \widehat{H}$ and the fact that $F\overline{F} = LI$:

$$\widehat{y} = \widehat{H}\widehat{x} \quad \text{is} \quad \overline{F}y = (F^{-1}HF)\overline{F}x \quad \text{which is} \quad y = Hx.$$

In words, the diagonalization is a change of basis from time to frequency. In the frequency domain, the filter multiplies each $\widehat{x}(k)$ by $\widehat{h}(k)$. We have agreement between a convolution $Hx$ and a pointwise multiplication $\widehat{H}\widehat{x}$:

$$\begin{bmatrix} 4 & 1 & 1 \\ 1 & 4 & 1 \\ 1 & 1 & 4 \end{bmatrix}\begin{bmatrix} x(0) \\ x(1) \\ x(2) \end{bmatrix} \quad \text{transforms to} \quad \begin{bmatrix} 6 & & \\ & 3 & \\ & & 3 \end{bmatrix}\begin{bmatrix} \widehat{x}(0) \\ \widehat{x}(1) \\ \widehat{x}(2) \end{bmatrix}.$$

For a specific example take

$$x = \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix} \quad \text{and} \quad \widehat{x} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & \overline{W} & \overline{W}^2 \\ 1 & \overline{W}^2 & \overline{W}^4 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 1 \\ 1 \end{bmatrix}.$$

Parseval's identity says that $2^2 + 1^2 + 1^2$ should be $\frac{1}{L}$ times $4^2 + 1^2 + 1^2$. (6 is $\frac{1}{3}$ of 18.) The matrix multiplication $\boldsymbol{Hx}$ and the pointwise $\widehat{\boldsymbol{H}}\widehat{\boldsymbol{x}}$ give

$$y = Hx = \begin{bmatrix} 10 \\ 7 \\ 7 \end{bmatrix} \quad \text{and} \quad \widehat{y} = \widehat{H}\widehat{x} = \begin{bmatrix} 6 \cdot 4 \\ 3 \cdot 1 \\ 3 \cdot 1 \end{bmatrix} = \begin{bmatrix} 24 \\ 3 \\ 3 \end{bmatrix}.$$

Again Parseval's identity checks: $10^2 + 7^2 + 7^2 = 198$ is $\frac{1}{3}$ of $24^2 + 3^2 + 3^2$.

Finally we display $\boldsymbol{Hx}$ as a ***circular convolution***. One way is to extend $x$ periodically. Then ordinary convolution has periodic output 10, 7, 7 (repeated):

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Periodic signal | 2 | 1 | 1 | 2 | 1 | 1 | 2 | 1 | 1 | 2 | 1 | 1 |
| Normal filter | | 1 | 1 | 4 | | 1 | 1 | 4 | | 1 | 1 | 4 |
| | | | 10 | | | | 7 | | | | 7 | |

Another way is to think of convolution as multiplication $H(z)X(z)$. But in this circular case $z^3 = 1$ and $z^4 = z$. Therefore $H(z)$ times $X(z)$ is

$$\begin{aligned} (4 + z + z^2)(2 + z + z^2) &= 8 + 6z + 7z^2 + 2z^3 + z^4 \qquad \text{(ordinary)} \\ &= 10 + 7z + 7z^2 \qquad \text{(wraparound).} \end{aligned}$$

This is $Y(z)$. Therefore $y = (10, 7, 7)$. We summarize the essential points:

The circular convolution of $\boldsymbol{h}$ with $\boldsymbol{x} = (x(0), x(1), \ldots)$ is $\boldsymbol{h} \circledast \boldsymbol{x} = \boldsymbol{Hx}$.

All vectors have length $L$. The transform of $\boldsymbol{y} = \boldsymbol{h} \circledast \boldsymbol{x}$ is $\widehat{y}(k) = \widehat{h}(k)\widehat{x}(k)$.

This convolution rule replaces the matrix multiplication $\boldsymbol{Hx}$ by the pointwise $\widehat{\boldsymbol{H}}\widehat{\boldsymbol{x}}$. We have to transform $\boldsymbol{h}$ and $\boldsymbol{x}$ to the Fourier domain! And then transform $\widehat{y}$ back to $y$. So three DFT's and the $L$ multiplications $\widehat{h}(k)\widehat{x}(k)$ replace *one* circular convolution — which is multiplication by $\boldsymbol{H}$:

$$h = \begin{bmatrix} 4 \\ 1 \\ 1 \end{bmatrix} \to \widehat{h} = \begin{bmatrix} 6 \\ 3 \\ 3 \end{bmatrix} \quad \searrow$$

$$x = \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix} \to \widehat{x} = \begin{bmatrix} 4 \\ 1 \\ 1 \end{bmatrix} \quad \nearrow \qquad \widehat{y} = \widehat{h}\widehat{x} = \begin{bmatrix} 24 \\ 3 \\ 3 \end{bmatrix} \to y = \begin{bmatrix} 10 \\ 7 \\ 7 \end{bmatrix}$$

In the time domain $\boldsymbol{Hx}$ needs about $NL$ multiplications. The $N$ filter coefficients are used $L$ times each. Compare with three DFT steps, which are executed by the ***Fast Fourier Transform***. Each FFT requires only $\frac{1}{2}L \log_2 L$ individual steps. Transforming is worthwhile when $N > \frac{3}{2} \log_2 L$.

In matrix notation, $\boldsymbol{F}$ is the product of $\log_2 L$ very sparse matrices, involving only $L/2$ multiplications each. This FFT recursion connects the $L$-point transform to two copies of the $\frac{L}{2}$-point

transform. It produces a matrix that is *half zero* and two very simple matrices around it. Here is the key to the FFT when $L = 1024$:

$$\textbf{\textit{FFT factors}}: \quad F_{1024} = \begin{bmatrix} I_{512} & D_{512} \\ I_{512} & -D_{512} \end{bmatrix} \begin{bmatrix} F_{512} & \\ & F_{512} \end{bmatrix} \begin{bmatrix} \text{even} \\ \text{odd} \end{bmatrix}. \tag{8.12}$$

$I_{512}$ is the identity matrix. $D_{512}$ is the diagonal matrix with entries $(1, W, \ldots, W^{511})$. The key is the two copies of $F_{512}$ (which use $W^2$, the 512th root of 1). The permutation at the end separates the incoming vector into its even part $(\downarrow 2)x = (x(0), x(2), \ldots)$ and its odd part $(x(1), x(3), \ldots)$.

This reduction from $F_L$ to two copies of $F_{L/2}$ almost cuts the work in half. We only need an extra 512 multiplications for $D_{512}$, plus additions and permutations. The full FFT algorithm keeps going, recursively. Each Fourier matrix $F_{512}$ is replaced by two copies of $F_{256}$, with diagonal matrices $D_{256}$ and a permutation that starts with $(\downarrow 4)$. Since 1024 is $2^{10}$, it takes only 10 steps to go from $F_{1024}$ to $F_1$! Maybe we don't go that far, but we could:

$F_{1024}$ is the product of 10 matrices with $I$'s and $D$'s and a permutation.

*The* $10 = \log_2 L$ *matrices each require* $512 = \frac{1}{2}L$ *multiplications for the* $D$*'s. So the total count for the FFT is* $\frac{1}{2}L \log_2 L$.

This simple idea changes $(1024)^2$ multiplications for the full $F_{1024}$ into 5(1024) multiplications for its ten sparse factors. That is a savings ratio of 200. Very rarely, perhaps never, has a single algorithm produced such a revolution as the FFT.

### Problem Set 8.1

1. Find the determinant of $S_4 - \lambda I$ and the eigenvalues of $S_4$:

$$\det(S_4 - \lambda I) = \det \begin{bmatrix} -\lambda & 0 & 0 & 1 \\ 1 & -\lambda & 0 & 0 \\ 0 & 1 & -\lambda & 0 \\ 0 & 0 & 1 & -\lambda \end{bmatrix}.$$

2. Each term in an $L$ by $L$ determinant has one factor from each row and column. Why are $(-\lambda)^L$ and $1^L$ the only possibilities in the determinant of $S_L - \lambda I$? Why not products of $-\lambda$ with 1?

3. If $h(0), \ldots, h(3)$ is the first column of the circulant matrix $H$ and $g(0), \ldots, g(3)$ is the first column of the circulant matrix $G$, show that the product $GH$ is a circulant matrix with first column $g \circledast h$. One approach is to diagonalize $H = F\widehat{H}F^{-1}$ and $G = F\widehat{G}F^{-1}$.

4. Find the eigenvalues and eigenvectors of

$$H_3 = \begin{bmatrix} 6 & -1 & 1 \\ 1 & 6 & -1 \\ -1 & 1 & 6 \end{bmatrix} \quad \text{and} \quad H_4 = \begin{bmatrix} 0 & 1 & 2 & 1 \\ 1 & 0 & 1 & 2 \\ 2 & 1 & 0 & 1 \\ 1 & 2 & 1 & 0 \end{bmatrix}.$$

5. If $x = (1, 1, 1)$ then $H_3x = (6, 6, 6)$. Find that answer from the convolution rule using $\widehat{h}(0)\widehat{x}(0)$ and $\widehat{h}(1)\widehat{x}(1)$ and $\widehat{h}(2)\widehat{x}(2)$.

6. Suppose the decimation operator $(\downarrow M)$ is applied to a periodized signal of length $L$. How many independent samples do we get if $M$ divides $L$? If this occurs in each of $M$ channels, find the total number of subsamples. Why is $M$ required to divide $L$?

7. On the interval $0 \le t \le 1$, draw the four Haar wavelets and the four (periodized) Daubechies $D_4$ wavelets at scale level $\frac{1}{4}$. These are the basis functions for the 4-dimensional space $W_2$.
8. When is a circulant matrix invertible? The entries in its first column are $h(0), \ldots, h(L-1)$. The transform of that column vector is $(\widehat{h}(0), \ldots, \widehat{h}(L-1))$. The answer is in the transform, since $H$ becomes a multiplication.
9. Write out explicitly the factorization (8.12) for the Fourier matrix $F_4$.

## 8.2 Symmetric Extension for Symmetric Filters

Now we come to the heart of the finite length problem. A signal of length $L$ will be extended in a symmetric way. The extended signal is $\boldsymbol{Ex}$ and the filter produces $\boldsymbol{HEx}$. If the filter is symmetric (or antisymmetric), then the output $\boldsymbol{HEx}$ will be symmetric (or antisymmetric). Downsampling yields $(\downarrow 2)\boldsymbol{HEx}$. ***The extension must be chosen so that this subsample is symmetric.*** Then we restrict attention to a piece of that subband signal, of length approximately $L/2$. (The exact integer will be crucial.) This restriction $\boldsymbol{R}$ yields the output from that channel:

$$\boldsymbol{R}(\downarrow 2)\boldsymbol{HEx} = (\textit{restriction})(\downarrow 2)(\textit{filter})(\textit{extension of } \boldsymbol{x}).$$

*To repeat:* A linear phase signal convolved with a linear phase filter has a linear phase output. You can see this in the time domain by simple algebra. In the frequency domain, the linear exponents in the phase add and stay linear. The key question is whether the *subsample* has linear phase, and what phase it has. We need to know the *point of symmetry* at every step of $\boldsymbol{R}(\downarrow 2)\boldsymbol{HEx}$.

The advantage of symmetric extension is this. *We may introduce a corner but we don't introduce a jump.* Wraparound creates a jump because generally $x(L)$ does not equal $x(0)$. With the extension, $x(2L)$ *does* equal $x(0)$. So you should extend before wraparound.

The Discrete Cosine Transform (DCT) is an example of symmetric extension. But there is no filter. It extends a block at a time, period. The DFT of the double block has only cosine terms, by symmetry. The DCT of Type II is the restriction back to the original block.

The DCT of Type IV extends even further, to period $4L$ instead of $2L$. After computing the DFT of a full period, we always restrict back to $L$ coefficients. (More would be redundant, since we started with $L$ samples.) Section 8.3 will describe both types of DCT. The present section is about extension and restriction of ordinary signals, before and after ordinary filtering.

### Symmetric Extension and Symmetric Subsamples

One decision becomes crucial for multirate filter banks. Do we repeat the first and last samples, or do we not repeat? The results after the sampling operator $(\downarrow 2)$ are very different. We face a question that does not arise for the DCT. *Is the subsampled signal still symmetric?* That requirement will govern our extension of the input signal. We will take up in order the key decisions and the reasons behind them:

1. When to repeat the first and last samples (depending on $N$).
2. Where to start the subsampling (depending on $L$).
3. How many subsamples to keep from each channel.

The input is a set of $L$ numbers $x(0), \ldots, x(L-1)$. The output is to be a total of $L$ samples from the lowpass and highpass channels. We must choose a method that gives exactly $L$ nonredundant outputs. Then we have a nonexpansive transform (a square matrix).

### Repeating (H Extension) or Not Repeating (W Extension)

There are two symmetric ways to extend a signal that starts with $x(0)$. Everything depends on the choice of $x(-1)$. We may take $x(-1) = x(1)$, and then continue with $x(-2) = x(2)$. The "point of symmetry" is $t = 0$. When we do this at both ends, $t = 0$ and $t = L-1$, the period is $2L-2$. This is called ***whole-point symmetry***, and it is indicated by the letter W — not in italic.

This W extension corresponds to $x(-t) = x(t)$ in continuous time. In discrete time, it means that the number $x(0)$ is not repeated. At the other end, $x(L-1)$ also appears only once. All other samples $x(n)$ reappear as $x(-n)$.

The other possibility is to choose $x(-1) = x(0)$. The value at $t = 0$ is repeated at $t = -1$. The point of symmetry is halfway between, at $t = -\frac{1}{2}$. The extension continues with $x(-2) = x(1)$. Figure 8.3 shows how this *half-point symmetry* produces a signal that has period $2L$. It is sometimes referred to as a (2, 2) extension, to indicate the repetition at both ends. The extension is symmetric about $t = -\frac{1}{2}$ and $t = L-\frac{1}{2}$ and $t = 2L-\frac{1}{2}$. This half-point symmetry is indicated by the non-italic letter H.

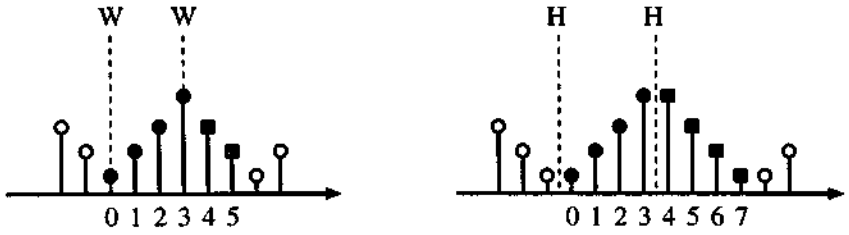



**Figure 8.3:** Whole-point symmetry (W) and half-point symmetry (H). The input has $L = 4$. The period is $2(L-1) = 6$ or $2L = 8$.

*Which extension to choose,* **W** *or* **H***?* The answer depends on the filter length. *The right choice depends on whether the filter itself is* **W** *or* **H**. A symmetric filter with an odd number of coefficients, say $h(0)$, $h(1)$, $h(0)$, is a W filter. The point of symmetry is the center point 1. A symmetric filter with an even number of coefficients, say $h(0)$, $h(1)$, $h(1)$, $h(0)$, is an H filter. It is symmetric about the half-point $\frac{3}{2}$. The H filter has repetition, $h(1) = h(2)$, and the W filter has whole-point symmetry with no repetition of the center coefficient.

The goal is to have a symmetric extension after downsampling. Then restriction step will succeed. The subsampled signal $(\downarrow 2)y = (\downarrow 2)\boldsymbol{HEx}$ is always periodic, but it is symmetric only if we follow the rule:

*Use a* **W** (no repeat) *extension for a* **W** (odd length) *filter.*

*Use an* **H** (repeat) *extension for an* **H** (even length) *filter.*

*Those rules both give a* **W** *extension for* $\boldsymbol{HEx}$. Downsampling preserves symmetry.

A mixture of **H** and **W** would give an **H** extension for $y = \boldsymbol{HEx}$. Downsampling an **H** extension goes wrong! Try it for $y(2)$, $y(1)$, $y(0)$, $y(0)$, $y(1)$, $y(2)$.

The clearest way to justify this rule is to try to break it. Figure 8.4 shows the filtered outputs $y = \boldsymbol{HEx}$ when the extension and filter are of opposite types (one is W, the other is H). Downsampling $y$ will not produce a symmetric signal. Then the restriction of $(\downarrow 2)y$ will fail. We show only failure here because there are many variations of success still to consider. The signal length $L$ is going to enter soon (and $L = 4$ does not show all variations of failure either).

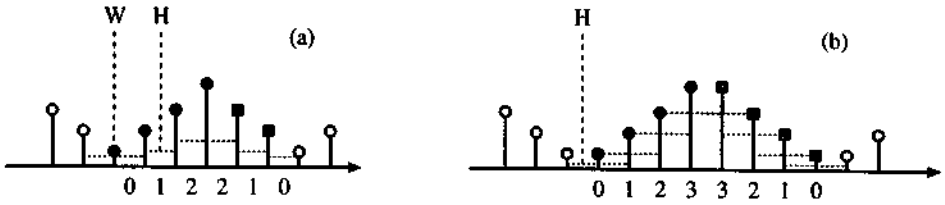



**Figure 8.4:** H and W do not mix. (a) W extension and H filter (two taps) give H extension. Downsampling yields 0, 2, 1 repeated. (b) H extension and W filter (three taps) give H extension. Downsampling yields 0, 2, 3, 1 repeated. *These subsamples are not symmetric and restriction fails.*

### Length of Signal and Center of Filter

W extension will be chosen for W filters (odd length, symmetric or antisymmetric). H extension will be used for H filters. The next question is where to "center" the filter, and that depends on the signal length $L$. The goal is a total of $L$ nonredundant samples from the two channels.

We do not study a (1, 2) or (2, 1) extension, with one endpoint repeated, because the period is then odd. This cannot be used with $(\downarrow 2)$. For an $M$-channel filter bank with odd $M$, new possibilities appear: W can mix with H. A complete table of admissible extensions is in the valuable paper by [Brislawn], which has been our guide. Earlier references are [SmEd, KiYaIw].

Start with the W extension (no repeat of end values). The filter length is odd, and both analysis filters (lowpass and highpass) are symmetric. The extended signal $\boldsymbol{Ex}$ has period $2(L-1)$. Figure 8.5 shows $L = 4$ on the left with an extension of period 6. The filtered values $y = \boldsymbol{HEx}$ also have period 6. Those $y$'s display a W extension (no repeats). Downsampling gives a symmetric extension. (It has to be (1, 2) because the half-period $L - 1$ is odd.) Then restriction produces $L/2 = 2$ independent samples from each channel.

On the right side of Figure 8.5 is an odd-length signal ($L = 5$). The filter has odd length 3. So a W extension of $x$ leads to a W extension for $y$, after filtering. There are 5 independent $y$'s. By correct centering (in other words, correct phase) we get $\frac{1}{2}(L+1)$ subsamples from the lowpass channel and $\frac{1}{2}(L-1)$ subsamples from the highpass channel. The analysis bank is nonexpansive, producing $L$ outputs.

In the time domain this analysis bank is expressed by an $L \times L$ matrix. Then the synthesis bank (from biorthogonal filters) is expressed by the inverse matrix.

Now we go to an H extension (with repeat) for an H filter (even length). The extension has period $2L$. The lowpass filter will be symmetric and the highpass filter will be antisymmetric. For simplicity we use the average – difference pair with coefficients $\frac{1}{2}$, $\frac{1}{2}$ and $\frac{1}{2}$, $-\frac{1}{2}$. We still aim for $L$ nonredundant subband samples.

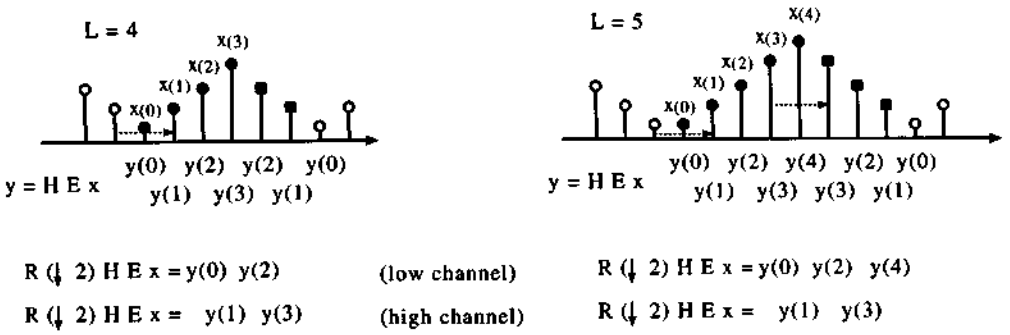



**Figure 8.5:** W extension and W filter yield $y = \boldsymbol{HEx}$ with a W extension. The samples $(\downarrow 2)y$ are symmetric and the restriction $\boldsymbol{R}$ succeeds.

On the left of Figure 8.6 is the case of even length $L = 4$ (four solid bullets). The extended period is 8. The lowpass filter produces $y$'s and the antisymmetric highpass filter produces $z$'s. (This combination is required for biorthogonality with even length.) Now the centering makes a difference. We can get subsamples of the original size from downsampling:

$$\boldsymbol{R}(\downarrow 2)\boldsymbol{HEx} = y(0), y(2), y(4) \text{ and } z(2) \qquad (\text{total } L = 4)$$

Or, as we prefer, a change of center gives equal size samples from the two channels:

$$\boldsymbol{R}(\downarrow 2)\boldsymbol{HEx} = y(1), y(3) \text{ and } z(1), z(3) \qquad (\text{total } L = 4)$$

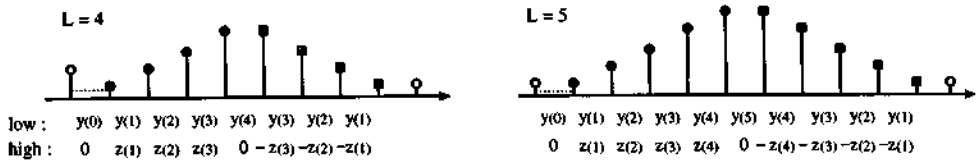



**Figure 8.6:** Even-length **H** filters with **H** extension. The filtered and subsampled signals have $L$ independent samples: $L/2$ from each channel on the left, $(L+1)/2$ and $(L-1)/2$ on the right.

On the right of Figure 8.6 is a signal with odd length $L = 5$. The lowpass filter gives 6 independent $y$'s. The highpass filter gives 4 independent $z$'s. There will be 3 and 2 good subsamples. It is natural to keep the same centering for the two channels. We may select $y(0)$, $y(2)$, $y(4)$ and $z(2)$, $z(4)$. Or select $y(1)$, $y(3)$, $y(5)$ and $z(1)$, $z(3)$. In all cases $L$ independent subsamples are returned by the analysis bank.

### Problem Set 8.2

1. Show that the symmetric extension of $f(t)$ from $[0, \pi]$ to $[-\pi, 0]$ gives a function $Ef(t)$ that has no sine terms: $\int_{-\pi}^{\pi} Ef(t) \sin kt \, dt = 0$. Give a formula for $Ef(t)$.
2. Extend a function $x(t)$ to have period $L$ (wraparound instead of symmetric extension). Draw the output from the centered filter $h(-1) = h(1) = \frac{1}{2}$. Rescale to $2t$ (period $L/2$) and draw the output $v(t)$.

3. Summarize the reason for choosing **H** extensions for **H** filters and **W** extensions for **W** filters: *We want the filtered values* $\mathbf{y} = \mathbf{HEx}$ *to be* **W**. If the filtered values are **H**, why does symmetry fail for $(\downarrow 2)\mathbf{y}$?
4. What is the difference in the $z$-domain between a **W**-extended signal and an **H**-extended signal? One of the extensions has $X(z) = X(z^{-1})$ and the other doesn't.
5. Draw the **W** and **H** extensions of a signal $\mathbf{x}$ of length $L = 6$. Apply a four-coefficient filter to each, and indicate the full period of the filtered signal $\mathbf{y}$. Which yields a satisfactory subsample $(\downarrow 2)\mathbf{y}$?
6. Draw the **W** and **H** extensions of a signal $\mathbf{x}$ of length $L = 7$. Apply a symmetric 5-coefficient filter and indicate a full period of the filtered signal $y$. Which subsamples would you choose?
7. Suppose the decimation operator is $(\downarrow 3)$ instead of $(\downarrow 2)$. How many independent samples if $\mathbf{x}$ has length $L$, with **W** extension and then with **H** extension? The main requirement is that the period ($2L$ or $2L-2$ or even $2L-1$) should be divisible by $M$: ***Why?***
8. Show that $Ex(t) = \sum_{-\infty}^{\infty} [f(t-2n) + f(2n-t)]$ gives the symmetric extension with period 2 of a function $x(t)$ defined on $[0, 1]$.

## 8.3 Cosine Bases and the DCT

The Discrete Cosine Transform (DCT) improves on the DFT for the same reason that symmetric extension improves on periodic extension. *The symmetric extension is continuous.* The periodic extension generally has a jump. This section develops the DCT as an extension followed by the DFT of size $2L$ followed by a restriction back to length $L$. There are *four types of* DCT coming from four types of extension.

We start in continuous time, by extending $f(t)$ from $[0, \pi]$ to $[-\pi, \pi]$. The symmetric extension is $Ef(t) = f(-t)$ on $[-\pi, 0]$ and then $2\pi$-periodic. This even function $Ef(t)$ is a combination of cosines:

$$Ef(t) = \sum_{0}^{\infty} a_k \cos kt.$$

The discrete analogue is the DCT of Type I or II. There is a choice between **W** extension and **H** extension! We may define $x(-1)$ to equal $x(1)$ or to equal $x(0)$. This choice doesn't arise in continuous time, but the previous section showed its importance in discrete time.

There is another extension in continuous time. It is surprising and very important. We call it $E^{\text{IV}}$ because it leads to the DCT of Type IV. Start again with $f(t)$ on $[0, \pi]$. The extension is *even* across the left endpoint $t = 0$, as before, but it is *odd* across the right endpoint $t = \pi$. The extended (unfolded) function $E^{\text{IV}} f(t)$ is drawn in Figure 8.7. *It has period* $4\pi$. There is a jump at $t = \pi$ where the extension is odd. From the left side the graph approaches $f(\pi)$, from the right side the graph approaches $-f(\pi)$. This even-odd extension $E^{\text{IV}}$ is remarkably useful.

Notice that $E^{\text{IV}} f(t)$ involves cosines of period $4\pi$ (they are $\cos \frac{\ell t}{2}$). But it requires only *half of those cosines*, because the even frequencies $\ell = 2k$ have zero coefficients. These even frequencies yield $\cos kt$, which is an even function around $t = \pi$. The extension is an *odd* function around $t = \pi$, so the integral of $\cos kt$ times $E^{\text{IV}} f(t)$ over a period is zero. We are left with the odd frequencies $\ell = 2k + 1$ and the basis functions $\cos \frac{\ell t}{2} = \cos\left(k + \frac{1}{2}\right) t$:

$$E^{\text{IV}} f(t) = \sum_{0}^{\infty} a_k \cos\left(k + \tfrac{1}{2}\right) t. \tag{8.13}$$

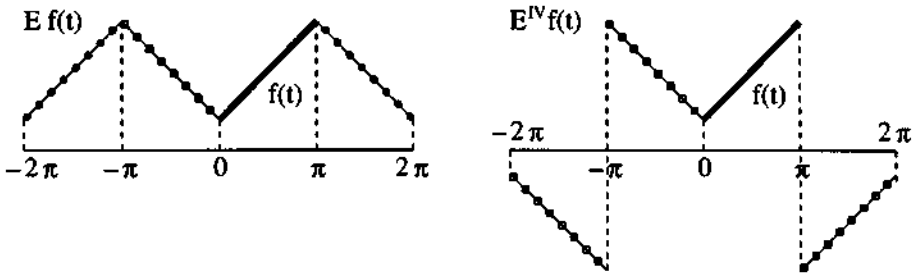



**Figure 8.7:** The symmetric extension $Ef(t)$ is continuous with period $2\pi$. The even-odd extension $E^{IV}f(t)$ is discontinuous with period $4\pi$.

### Discrete Extension and the DCT-II

The function $f(t)$ on $[0, \pi]$ is now replaced by a vector $(x(0), \ldots, x(L-1))$. We will extend that vector and apply the DFT (the Fourier matrix $F_{2L}$) of *doubled length* $2L$. ***The whole-sample* W *extension is not satisfactory***. By symmetry it creates $x(-n) = x(n)$, extending the definition to $|n| \leq L-1$. This is $2L-1$ samples, and we do not want an odd number. It is better to use the half-sample **H** extension, which repeats $x(0)$ as $x(-1)$. The point of symmetry is $-\frac{1}{2}$, but this inconvenience is small. The symmetric **H** extension has period $2L$, and we call it $E^{II}$:

$$E^{II}x(-n) = x(n-1) \quad \text{for} \quad n = 1, \ldots, L. \tag{8.14}$$

The Fourier expansion of $E^{II}x$ uses an ordinary DFT of length $2L$. That has $2L$ basis functions indexed by frequency $k$. The $n$th component of the $k$th basis vector is normally $e^{2\pi ikn/2L}$. But we shift $n$ to $n+\frac{1}{2}$ in order to move the point of symmetry. The shifted signal is even around zero, and can be expressed by $L$ cosines. The shift brings out a factor $e^{i\pi k/2L}$, and the expansion of the $2L$-periodic signal is

$$E^{II}x(n) \quad = \quad \sum_{k=0}^{L-1} e^{i\pi k/2L}\widehat{x}^{II}(k)\cos\frac{\pi k\left(n+\frac{1}{2}\right)}{L}. \tag{8.15}$$

Orthogonality of the exponentials leads to orthogonality of the cosines (not trivial to see). The modulation factors $e^{i\pi k/2L}$ of absolute value 1 just multiply the separate basis functions (indexed by $k$).

The $L \times L$ DCT-II matrix in this cosine expansion has entries

$$C^{II}(k,n) = b(k)\sqrt{\frac{2}{L}}\cos\frac{\pi k\left(n+\frac{1}{2}\right)}{L}.$$

The square root of $\frac{2}{L}$ is included to make the rows of $C^{II}$ into unit vectors. So is the factor

$$b(k) = \begin{cases} 1/\sqrt{2} & \text{if } k = 0 \\ 1 & \text{if } k = 1, \ldots, L-1. \end{cases}$$

This is the familiar factor for the zero frequency $k = 0$ when the cosine stays at 1.

The key points about $C^{II}$ are its *orthogonality* and *fast execution*. Those are proved by the factorization (8.18) below. They follow from the connection to the DFT (and therefore the FFT).

### Even–Odd Extension and the DCT-IV

The even-odd extension $E^{IV}$ was applied to $f(t)$ in continuous time. Its discrete analogue leads to the DCT matrix of Type IV. This is the $L$ by $L$ matrix $\boldsymbol{E}^{IV}$, with remarkable properties.

Again the original signal has $L$ components $x(n)$. This is extended evenly across $n = -\frac{1}{2}$, with H symmetry, to a signal with $2L$ components. *Then comes an antisymmetric* **H** *extension to* $4L$ *components.* Figure 8.8 shows the basic period of $4L$, which repeats.

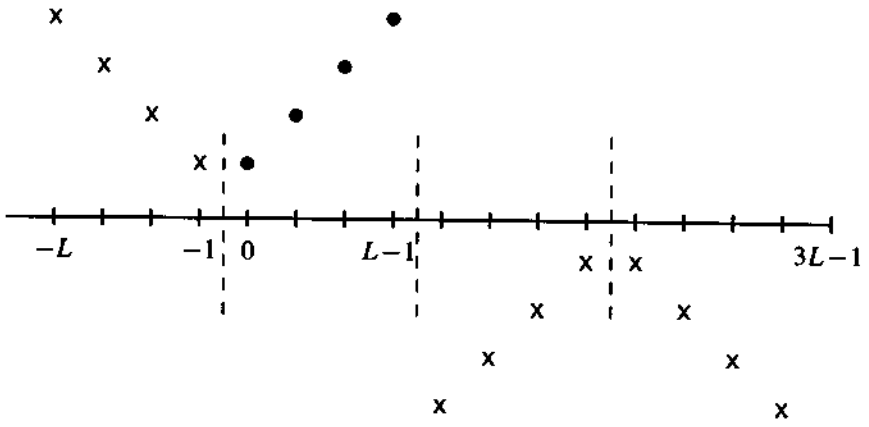



**Figure 8.8:** $x^{IV}(n)$ is even across $n = -\frac{1}{2}$ and odd across $n = L - \frac{1}{2}$. Here $L = 4$.

The basis functions for $E^{IV} f(t)$ in continuous time were $\cos(k+\frac{1}{2})t$. Those came from odd frequencies $\ell = 2k+1$ and from period $4\pi$. The basis functions in discrete time have period $4L$ and the same odd frequencies $\ell = 2k+1$. There is also the shift from $n$ to $n+\frac{1}{2}$, which moves the point of symmetry to zero and the point of antisymmetry to $L$. Then the basis functions appear in the $L \times L$ DCT-IV matrix

$$C^{IV}(k,n) = \sqrt{\frac{2}{L}} \cos \frac{\pi(k+\frac{1}{2})(n+\frac{1}{2})}{L}. \tag{8.16}$$

Notice the good properties of this matrix. It is symmetric. It needs no $b(k)$ to insert $\sqrt{2}$ into a zero-frequency term. Its columns are the even-odd basis vectors, and Figure 8.9 shows the first four vectors in the $C^{IV}$ basis of length $L = 32$. The DCT matrix also has *orthogonality* and *fast execution*, because of its direct link to the DFT matrix of order $2L$ — which we now explain.

### Matrix Factorizations and Orthogonality: DCT-II and DCT-IV

By slightly adapting Wickerhauser's very neat exposition [W, pp. 90–96], we can achieve three useful purposes at the same time:

1. Include the discrete *sine* transforms: DST-II and DST-IV
2. Prove the *orthogonality* of all four transforms: DCT and DST, II and IV
3. Show explicitly how these four transforms connect by extremely sparse matrices to the DFT matrix $\boldsymbol{F}_{2L}$.

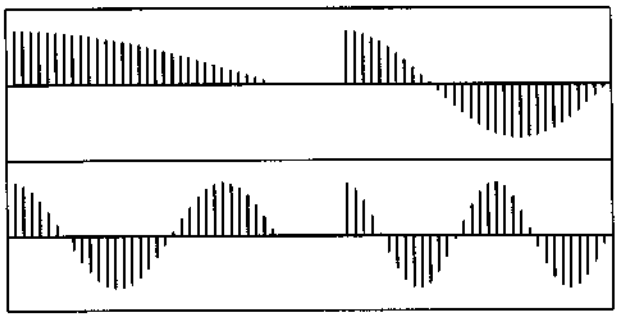


**Figure 8.9:** The first four basis functions of the DCT-IV, for $M = 32$.

These results come directly from two matrix factorizations, which we will display but not prove. A MATLAB exercise verifies their correctness for each $L$. You will notice that the DCT-IV and DST-IV formulas are cleaner, except for the multiplier $e^{-\pi i/4L}$ from the shift by $\frac{1}{2}$. Type II must assign $1/\sqrt{2}$ to the zero frequency DC term. Here are the formulas that prove everything.

$$\textit{Factorization IV:} \quad e^{-\pi i/4L} R^T F_{2L} R = \begin{bmatrix} C^{\text{IV}} & \\ & -iS^{\text{IV}} \end{bmatrix} \tag{8.17}$$

$$\textit{Factorization II:} \quad \overline{P}^T F_{2L} Q = \begin{bmatrix} C^{\text{II}} & \\ & S^{\text{II}} \end{bmatrix} \tag{8.18}$$

All matrices on the left are square and orthogonal of size $2L$. Then each block on the right is square and orthogonal of size $L$. The sine and cosine transforms preserve length (discrete Parseval identity as for the DFT). Note that $R$ is only transposed while $P$ is also conjugated.

It remains to display the sparse matrices, with at most two entries in each column and row. $J$ is the reverse identity, with 1's on the antidiagonal. The combination of $I$ and $J$ gives the Gray ordering in (8.21). The matrices $R$ and $P$ clearly have orthogonal columns, and the division by $\sqrt{2}$ yields unit vectors:

$$R = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & & & & 1 & & & \\ & \overline{W} & & & & \overline{W} & & \\ & & \ddots & & & & \ddots & \\ & & & \overline{W}^{L-1} & & & & \overline{W}^{L-1} \\ & & & W^L & & & & -W^L \\ & & \iddots & & & & \iddots & \\ & W^2 & & & & -W^2 & & \\ W & & & & -W & & & \end{bmatrix}, \quad W = e^{\pi i/2L}$$

$$P = \frac{1}{\sqrt{2}} \begin{bmatrix} \sqrt{2} & & & & 0 & & & \\ & W & & & & W & & \\ & & \ddots & & & & \ddots & \\ & & & W^{L-1} & & & W^{L-1} & 0 \\ & & & 0 & & & & i\sqrt{2} \\ & & & \overline{W}^{L-1} & & & -\overline{W}^{L-1} & \\ & & \iddots & & & \iddots & & \\ 0 & \overline{W} & & & -\overline{W} & & & \end{bmatrix}, \quad Q = \frac{1}{\sqrt{2}} \begin{bmatrix} I & I \\ J & -J \end{bmatrix}.$$

**Summary:** We now have two discrete cosine transforms, II and IV. There are two further types, I and III, from W extensions. Those are definitely less useful. If we omit the $\sqrt{2/L}$ normalization for simplicity, the DCT-II and DCT-IV transform a length $L$ input signal into vectors $X^{\mathrm{II}}$ and $X^{\mathrm{IV}}$:

$$X^{\mathrm{II}}(k) = b(k)\sum_{n=0}^{L-1} x(n)\cos\frac{\left(n+\frac{1}{2}\right)k\pi}{L} \tag{8.19}$$

$$X^{\mathrm{IV}}(k) = \sum_{n=0}^{L-1} x(n)\cos\frac{\left(n+\frac{1}{2}\right)\left(k+\frac{1}{2}\right)\pi}{L}. \tag{8.20}$$

Malvar describes the DCT-II as better for transform coding. The DCT-IV is good for spectrum estimation and adaptive filtering. For us the key point is the efficiency of the algorithms.

Several implementations are possible and we describe them briefly, for even $L$. First we connect each DCT to the DFT. Then we connect DCT-II to DCT-IV.

**DCT-II via DFT.** The trick is in this reordering of $x(n)$ (the Gray order):

$$y(n) = x(2n) \text{ and } y(L-n-1) = x(2n+1) \text{ for } n = 0, 1, \ldots, \frac{L}{2} - 1. \tag{8.21}$$

Now take the DFT of $y(n)$. The DCT-II coefficients are

$$X^{\mathrm{II}}(k) = \cos\frac{\pi k}{2L}\,\mathrm{Re}[\widehat{y}(k)] - \sin\frac{\pi k}{2L}\,\mathrm{Im}[\widehat{y}(k)]. \tag{8.22}$$

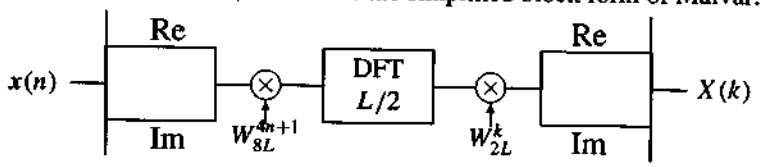
The numbers $X^{\mathrm{II}}(k)$ and $X^{\mathrm{II}}(L-k)$ come from a plane rotation by $\frac{\pi k}{2L}$. Each step (permutation, DFT, rotation) preserves vector length. Again, an *orthogonal matrix*.

**DCT-IV via DFT.** The same reordering and modulation will create $\frac{L}{2}$ complex numbers

$$c(n) = (x(2n) + ix(L-1-2n))\,e^{-i(n+\frac{1}{4})\pi/L}. \tag{8.23}$$

Now follows the half-length DFT. The $k$th output is multiplied by $e^{-ik\pi/L}$. Then the real and imaginary parts of these $\frac{L}{2}$ complex numbers give the DCT-IV coefficients $X^{\mathrm{IV}}(k)$.

Since codes are available, we imitate the simplified block form of Malvar:




The algebra behind this algorithm is expressed in two lines, which require superhuman patience. The first line separates even and odd components $x(n)$:

$$X^{\mathrm{IV}}(k) = \sum_{n=0}^{\frac{L}{2}-1}\left[x(2n)\cos\frac{\left(2n+\frac{1}{2}\right)\left(k+\frac{1}{2}\right)\pi}{L} + x(L-1-2n)\cos\frac{\left(L-1-2n+\frac{1}{2}\right)\left(k+\frac{1}{2}\right)\pi}{L}\right].$$

Replace cosines by exponentials and $x$'s by $c$'s. Then separate into the real and imaginary parts of an $\frac{L}{2}$-point DFT times a modulation, for a fast DCT-IV:

$$\begin{matrix} X^{\mathrm{IV}}(2k) \\ X^{\mathrm{IV}}(L-1-2k) \end{matrix} = \begin{matrix} \mathrm{Re} \\ -\mathrm{Im} \end{matrix} \left[ e^{-ik\pi/L} \sum_{n=0}^{\frac{L}{2}-1} c(n) e^{-4i\pi kn/L} \right] \tag{8.24}$$

**DCT-II via DCT-IV.** Finally we show that $X^{\mathrm{II}}$ of length $L$ is immediately available from a DCT-II and a DCT-IV of length $\frac{L}{2}$. Recursively, this produces $X^{\mathrm{II}}$ from a sequence of DCT-IV's — whose fast computation was just given. The reduction of $X^{\mathrm{II}}$ comes by rewriting (8.19) in the form II plus IV:

$$\begin{aligned} X^{\mathrm{II}}(k) = b(k) &\sum_{n=0}^{\frac{L}{2}-1} [x(n) + x(L-1-n)] \cos \frac{\left(n+\frac{1}{2}\right)k\pi}{L/2} \\ &+ \sum_{n=0}^{\frac{L}{2}-1} [x(n) - x(L-1-n)] \cos \frac{\left(n+\frac{1}{2}\right)\left(k+\frac{1}{2}\right)\pi}{L/2}. \end{aligned} \tag{8.25}$$

This recursion to a shorter DCT-II and DCT-IV was known early. It requires only human patience to verify. When a fast DCT-IV was not available, the connection was useless. Now the DCT-IV algorithm gives a unified approach to the two most important versions of the discrete cosine transform.

In closing we emphasize two points. First, orthogonality of the DCT comes from extension followed by DFT followed by restriction. Second, there are still blocking effects from the limited smoothness. Symmetric extension is a big improvement on periodicity. The next section shows how to smooth the basis functions much more.

### Problem Set 8.3

1. Find the DCT-II and DCT-IV transforms of the signals $x = (1, 1, 1, 1)$ and $x = (1, -1, 1, -1)$.
2. Find the DCT-II and DCT-IV transforms of the signals $x = (1, 0, 0, 0)$ and $x = (0, 1, 0, 0)$.
3. Show that the Gray reordering $y(n)$ in equation (8.21) transforms (8.19) into

$$x^{\mathrm{II}}(k) = b(k) \sum_{n=0}^{L-1} y(n) \cos \frac{\pi(4n+1)k}{2L}.$$

4. Write out explicitly the $2 \times 2$ and $3 \times 3$ DCT-II matrices and show that they are orthogonal.
5. The DCT-IV matrix has entries $C_{kn} = \cos[\frac{\pi}{L}(k+\frac{1}{2})(n+\frac{1}{2})]$. Write down this symmetric $L \times L$ matrix for $L = 2$ and 3. Verify that columns 0 and 1 are orthogonal.
6. Execute this MATLAB verification of (8.17–8.18), connecting the DCT-DST-DFT:

```
L=16; i=eye(L); k=[0:L-1]; n=k';
dctiv=sqrt(2/L)*cos((k+1/2)*(n+1/2)*pi/L);  % orthogonal dctiv matrix
dctii=sqrt(2/L)*cos(k*(n+1/2)*pi/L);
dctii(1,:)=sqrt(1/2)*dctii(1,:);            % orthogonal dctii matrix
dft=sqrt(1/(2*L))*fft(eye(2*L));            % orthogonal dft matrix

Q=sqrt(1/2)*[i;i(L:-1:1,:)];                % Q matrix
W=exp(sqrt(-1)*pi/(2*L)); Wb=conj(W);       % W=2L-th root of one
```

```
p1=diag([sqrt(2)*W.^ (1:L-1)]);     % P matrix
p2=diag([Wb.^ (1:L-1)]);
P=sqrt(1/2)*[p1;zeros(1,L);[zeros(L-1,1) p2(L-1:-1:1,:)]];
c=P'*dft*Q;                         % dft to dctii mapping

r1=diag(Wb.^ (0:L-1)); r2=diag(W.^ (1:L));
R=sqrt(1/2)*[r1;r2(L:-1:1,:)];      % R matrix
z=exp(-sqrt(-1)*pi/(4*L));          % z=conjugate of 4L-th root of one
d=real(z*conj(R)'*dft*R);           % dft to dctiv
                                  % conj(x') gives x' without conjugate!
max(max(abs(c-dctii)))              % peak error, should be VERY small
max(max(abs(d-dctiv)))              % peak error, should be VERY small
```

## 8.4 Smooth Local Cosine Bases

To "localize" a cosine, just multiply it by a real window function $g(t)$. Windowing is the central idea of the Short Time Fourier Transform (STFT). When we use $e^{-i\omega t}$ rather than cosines, this is the representation pioneered by Gabor:

$$\text{The windowed transform of } f(t) \text{ is } F(\omega, a) = \int_{-\infty}^{\infty} f(t)g(t-a)e^{-i\omega t}\,dt. \tag{8.26}$$

Note the two variables $\omega$ = oscillation frequency and $a$ = window position. We have a *time-frequency plane*, in the same way that $k$ and $j$ for the wavelet $w\left(2^j t - k\right)$ place us on a *time-scale plane*. With all frequencies $\omega$ and all positions $a$, we expect redundancy in $F(\omega, a)$ and cannot expect orthogonality. Nevertheless $f(t)$ can be recovered:

$$\text{The inverse windowed transform is } f(t) = C_g \iint F(\omega, a)g(t-a)e^{i\omega t}dt\,da.$$

The constant $C_g$ depends on the window. This STFT is competition for the continuous wavelet transform. Both are described in Section 2.6.

With discrete frequencies and positions, we have an oscillation (a cosine) inside a modulating envelope (the window). When the window is a simple box, each block is independent. A compressed image looks as if it is built of tiles. This blocking effect is reduced by filtering and by smooth windows, but ringing often persists and the result is not perfect. The DCT was established as the JPEG standard in 1992, after six years of work by the Joint Photographic Experts Group. But the technology continues to move forward.

This whole construction has recently been made sharper, in order to obtain an ***orthonormal basis***. Specific frequencies and specific windows are chosen, and this is what we want to explain. The frequencies involve the factor $k + \frac{1}{2}$ that is responsible for the even-odd symmetry of the extension $E^{IV}$ and the discrete DCT-IV. Requirements are imposed on the windows to give orthogonality where one window overlaps its translate. This is not orthogonality of the windows alone or the cosines alone, but orthogonality of their products $g_{nk}(t)$ for $k = 0, 1, \ldots$ and $-\infty < n < \infty$:

$$g_{nk}(t) = g(t-n)\cos\left[\left(k+\tfrac{1}{2}\right)\pi(t-n)\right]. \tag{8.27}$$

If $g(t)$ is the box function, then $g(t-n) = 1$ on the time interval $[n, n+1]$. We have the basis functions (8.13) of the even-odd extension. They are the continuous analog of the DCT-IV basis. Moving to smoother windows $g(t-n)$ will reduce the blocking effects at the box edges.

A more general form, very useful in applications, allows windows of *varying length*. The time line is divided into unequal intervals $[t_n, t_{n+1}]$. There is a window function $g_n(t)$ for each interval. It equals one on an inner interval, and it drops toward zero as it passes $t_n$ and $t_{n+1}$. The window reaches zero well before $t_{n-1}$ on the left and $t_{n+2}$ on the right. This function $g_n(t)$ overlaps only its two neighbors $g_{n-1}(t)$ and $g_{n+1}(t)$. We mention immediately one key requirement for orthogonality of the basis:

$$\sum (g_n(t))^2 \equiv 1. \tag{8.28}$$

The cosines within the window still have the even-odd symmetry associated with DCT-IV. They are *even* when continued across the left end $t_n$ of the window, and they are *odd* across the right end $t_{n+1}$. This symmetry leads to orthogonality, when the windows have even symmetry at both ends and satisfy $\sum (g_n(t))^2 = 1$. The more general form of the basis functions is

$$g_{nk}(t) = g_n(t) \cos \frac{\left(k + \frac{1}{2}\right)\pi\,(t - t_n)}{t_{n+1} - t_n}. \tag{8.29}$$

This reduces to the previous form when $t_n = n$ and the window lengths are equal and the windows are translates of one window $g(t)$.

Varying window lengths is an extremely important advantage in speech processing. We attack and hold phonemes for very different lengths of time — and the waveforms in those intervals are very different. We indicate below how the windows lead to an orthogonal basis $g_{nk}(t)$ in this nonuniform case also. For simplicity of exposition, we concentrate first on the equal-interval construction of (8.27), which is important in discrete time too.

This section analyzes cosine windows in continuous time. The next chapter will study ***cosine-modulated filter banks*** in discrete time.

### The Window Functions

Start with a single window $g(t)$ and its translates $g(t - n)$. This window function is supported on an interval $[-a, 1 + a]$ that stretches beyond $[0, 1]$. If the extension length is $a < \frac{1}{2}$ at both ends, $g(t)$ will overlap only its nearest neighbors $g(t - 1)$ and $g(t + 1)$.

The window rises from zero at $t = -a$ to one at $t = a$. Here is a continuous rise:

$$g(t) = \sin\left[\frac{\pi}{4}\left(1 + \frac{t}{a}\right)\right] \quad \text{for } -a \le t \le a.$$

At $t = -a$ this is $g(-a) = \sin 0 = 0$. At $t = a$ the window reaches $g(a) = 1$. The window continues with $g(t) \equiv 1$ on the inner interval $[a, 1 - a]$. At the right end $g(t)$ drops to zero in the same way that it rose. Then the window is symmetric around its center point $t = \frac{1}{2}$, and the reflection $t \to 1 - t$ leaves it unchanged:

$$g(t) = \sin\left[\frac{\pi}{4}\left(1 + \frac{1 - t}{a}\right)\right] \quad \text{for } 1 - a \le t \le 1 + a.$$

We have drawn this $g(t)$ in Figure 8.10. The overlap with $g(t - 1)$ is crucial. The key property in that overlap region is

$$(g(t-1))^2 + (g(t))^2 = \sin^2\left[\frac{\pi}{4}\left(1 - \frac{t}{a}\right)\right] + \sin^2\left[\frac{\pi}{4}\left(1 + \frac{t}{a}\right)\right] = 1.$$

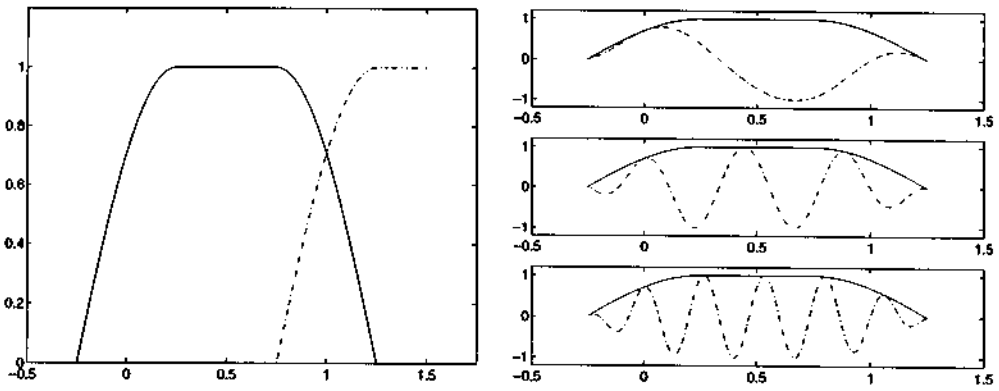


**Figure 8.10:** The window functions $g(t)$ and $g(t-1)$ for $a = 0.25$ (left). The basis functions of the local cosine transform for $k = 1, 4$ and 7.

This is really (cosine)$^2$ + (sine)$^2$ = 1. It leads directly to $\sum (g(t-n))^2 \equiv 1$ for all time. Away from the overlap intervals, a single window has unit height and all other windows are at zero.

This particular window can be made smoother. Replace $t$ in its definition by $\sin \frac{\pi t}{2a}$. The derivative $g'(t)$ has a factor $\cos \frac{\pi t}{2a}$ from the chain rule, and that factor is zero at $t = \pm a$. Each repetition of this step introduces one more continuous derivative in $g(t)$. (We do not know of a careful search for an optimal $a$ and an optimal window.) Allowing for unequal spacings and varying windows, the requirements at the overlap around $t = t_n$ are

$$(g_{n-1}(t))^2 + (g_n(t))^2 = 1 \quad \text{and} \quad g_{n-1}(t) = g_n(2t_n - t) \text{ for } |t - t_n| \le a_n. \tag{8.30}$$

In the time-invariant case, with uniform spacing, a fixed window $g(t)$ is shifted to $g_n(t) = g(t-n)$. The conditions (8.30) around $n = 0$ and $t_0 = 0$ apply at every $t_n = n$. These are the conditions that we work with first:

$$(g(t+1))^2 + (g(t))^2 = 1 \quad \text{and} \quad g(t+1) = g(-t) \text{ for } |t| \le a. \tag{8.31}$$

## Local Orthogonal Bases

Suppose the window $g(t)$ is a box on [0, 1]. The basis functions $\cos(k + \frac{1}{2})\pi t$ are orthogonal within that box. They are even functions around $t = 0$, where the cosine equals one. They are odd functions around $t = 1$, where the cosine of $(k + \frac{1}{2})\pi$ equals zero. The main point is that *we can smooth those windows and retain orthogonality*. The first author learned the proof from lectures at MIT by Stéphane Mallat, which will lead to a book on time-frequency resolutions including wavelets [Mt].

**Theorem 8.1** *The functions* $g_{nk}(t) = g(t-n)\cos\left[\left(k + \frac{1}{2}\right)\pi(t-n)\right]$ *are an orthogonal basis for* $L^2(\mathbf{R})$.

**Proof.** The function $g_{0k}(t)$ is centered on [0, 1] and extends over $[-a, 1+a]$. On the interval to the left, $g_{-1\ell}(t)$ is centered on $[-1, 0]$ and reaches as far right as $t = a$. The two functions overlap on the interval $[-a, a]$, where

$$g_{0k}(t)\, g_{-1\ell}(t) = g(t)\, g(t+1) \cos\left[\left(k + \tfrac{1}{2}\right)\pi t\right] \cos\left[\left(\ell + \tfrac{1}{2}\right)\pi(t+1)\right]. \tag{8.32}$$

We must show that the integral is zero and "the tails are orthogonal".

The product $g(t)g(t+1) = g(t)g(-t)$ is an even function around $t = 0$. So is the cosine of $(k+\frac{1}{2})\pi t$. The second cosine is odd, because the addition formula for cosines gives a sine:

$$\cos\left[\left(\ell+\tfrac{1}{2}\right)\pi(t+1)\right] = -\sin\left[\left(\ell+\tfrac{1}{2}\right)\pi t\right]\sin\left[\left(\ell+\tfrac{1}{2}\right)\pi\right].$$

The reader notices that the key is in $\cos(\ell+\frac{1}{2})\pi = 0$! Then the product (8.32) is odd and its integral over $[-a, a]$ is zero. This proves orthogonality of any two $g_{nk}$'s with different $n$'s.

Now consider $g_{0k}(t)g_{0\ell}(t)$. Both factors are centered on [0, 1], with different frequencies. Define the product of cosines $C_{k\ell}(t)$ as $\cos\left[\left(k+\frac{1}{2}\right)\pi t\right]\cos\left[\left(\ell+\frac{1}{2}\right)\pi t\right]$. Then $C_{k\ell}(t)$ is even around $t = 0$ and also around $t = 1$. (Odd times odd is even.) The other factor is $(g(t))^2$. At the left end, the key is that $(g(t))^2 + (g(-t))^2 = 1$:

$$\begin{aligned}\int_{-a}^{a}(g(t))^2 C_{k\ell}(t)\,dt &= \int_0^a (g(-t))^2 C_{k\ell}(t)\,dt + \int_0^a (g(t))^2 C_{k\ell}(t)\,dt \\ &= \int_0^a C_{k\ell}(t)\,dt. \end{aligned} \tag{8.33}$$

Around the right endpoint we use $(g(t))^2 + (g(t-1))^2 = 1$:

$$\int_{1-a}^{1+a}(g(t))^2 C_{k\ell}(t)\,dt = \int_{1-a}^{1} C_{k\ell}(t)\,dt. \tag{8.34}$$

The center interval $[a, 1-a]$ has $g(t) = 1$. Again we are integrating $C_{k\ell}(t)$, the product of cosines. Orthogonality of the cosines on [0, 1] gives orthogonality of $g(t)$ times cosines on the larger interval. Combine (8.33) and (8.34) to see that $g(t)$ disappears:

$$\boxed{\int_{-a}^{1+a}(g(t))^2 C_{k\ell}(t)\,dt = \int_0^1 C_{k\ell}(t)\,dt = \tfrac{1}{2}\delta(k-\ell).} \tag{8.35}$$

This proves that the local cosines $g_{nk}(t)$ are **orthogonal**. The integral on the left can go from $-\infty$ to $\infty$, since $g(t)$ is zero on the rest of the line. It remains to prove that the local cosines are a **basis** for $L^2(\mathbf{R})$. How do we express an arbitrary $f(t)$ as a combination of $g_{nk}(t)$?

The ordinary cosines $\cos\left[\left(k+\frac{1}{2}\right)\pi t\right]$ are an orthogonal basis on [0, 1]. We "fold" the function $f(t)$ into this interval, creating $h(t)$:

$$\textbf{Folding} = h(t) = \begin{cases} g(t)f(t) + g(-t)f(-t) & \text{on } [0, a] \\ f(t) & \text{on } [a, 1-a] \\ g(t)f(t) - g(2-t)f(2-t) & \text{on } [1-a, 1] \end{cases}$$

Within [0, 1] expand $h(t)$ in the basis $\cos\left(k+\frac{1}{2}\right)\pi t$. Those cosines extend $h(t)$ outside [0, 1], keeping it even around $t = 0$ and odd around $t = 1$. These symmetries are built into the definition of $h(t)$, so there is no change in formula on the larger interval $[-a, 1+a]$. Multiplying by $g(t)$ turns $h(t)$ = *combination of cosines* into $g(t)h(t)$ = *combination of* $g_{0k}(t)$. Here is $g(t)h(t)$:

$$g(t)h(t) = \begin{cases} (g(t))^2 f(t) + g(t)g(-t)f(-t) & \text{on } [-a, a] \\ f(t) & \text{on } [a, 1-a] \\ (g(t))^2 f(t) - g(t)g(2-t)f(2-t) & \text{on } [1-a, 1+a] \end{cases}$$

A similar step takes place on the interval $[-1, 0]$. Replace $t$ by $t + 1$ in all formulas. Then $h_{-1}(t)$ is defined from $f(t)$ and expressed as a combination of cosines. Multiplication by the shifted window gives $g(t+1)h_{-1}(t)$ as a combination of windowed cosines $g_{-1\ell}$, while $g(t)h(t)$ is produced out of $g_{0k}(t)$. All we have to check is that on the overlap interval $[-a, a]$, the sum $g(t)h(t) + g(t+1)h_{-1}(t)$ adds to $f(t)$. Since $g(t+1) = g(-t)$ in the overlap, we have it:

$$(g(t))^2 f(t) + g(t)g(-t)f(-t) + (g(-t))^2 f(t) - g(-t)g(t)f(-t) = f(t). \tag{8.36}$$

Thus every $f(t)$ is a combination of local cosines. Those are an orthogonal basis. *Theorem proved.*

Note that $g(t)h(t)$ is the even part of $f(t)$ around $t = 0$, and $g(t+1)h_{-1}(t)$ is the odd part. Equation (8.36) is simply (even part) + (odd part) $= f(t)$. An early construction by Wilson alternated $\cos k\pi t$ in one interval with $\sin k\pi t$ in the next interval. The cosines are even at both ends, the sines are odd at both ends, and each overlap interval again has even and odd — which gives orthogonality and reproduces $f(t)$ as above.

The proof also yields a ***fast algorithm***. To expand $f(t)$ in the local cosines, we expanded $h(t)$ and $h_{-1}(t)$ in ordinary cosines on $[0, 1]$ and $[-1, 0]$. So compute those coefficients $b_{0k}$ and $b_{-1\ell}$ by a fast DCT-IV. The same computation on other unit intervals gives all coefficients in $f(t) = \sum b_{nk}g_{nk}(t)$. We record this fact for the basic interval $[0, 1]$ and the coefficients $b_{0k}$:

$$2b_{0k} = \int_{-\infty}^{\infty} f(t)\, g_{0k}(t)\, dt = \int_0^1 h(t) \cos\left[\left(k + \tfrac{1}{2}\right)\pi t\right] dt. \tag{8.37}$$

The local cosine coefficients of $f(t)$ are the ordinary cosine coefficients of the "folded" function $h(t)$. The normalization by 2 comes from (8.35).

### Unequal Spacing and Different Windows

The analysis was simplified above, by assuming all windows to be translates $g(t - n)$ of a basic window. This is not necessary. The construction allows different windows $g_n(t)$ on intervals of different lengths. The time variable $t - n$ in the cosines can become $(t - t_n) / (t_{n+1} - t_n)$. The functions $g_{nk}(t)$ still provide an orthonormal basis.

The changes are easy to indicate, and their confirmation is a good exercise. The window $g_n(t)$ in Figure 8.11 extends from $t_n - a_n$ to $t_{n+1} + a_{n+1}$. The previous window $g_{n-1}(t)$ ends at $t_n + a_n$, and on the overlap interval the requirements are (8.30). The condition $a_n + a_{n+1} \le t_{n+1} - t_n$ avoids any overlap of $g_{n-1}(t)$ with $g_{n+1}(t)$. This is the non-uniform equivalent of $2a \le 1$. Then we have $\sum (g_n(t))^2 \equiv 1$ as desired.

The local cosine coefficients $b_{nk}$ of an arbitrary $f(t)$ are still the ordinary cosine coefficients of *folded functions* $h_n(t)$:

$$\int_{-\infty}^{\infty} f(t)g_n(t)\cos\left[\left(k + \tfrac{1}{2}\right)\pi\frac{t - t_n}{t_{n+1} - t_n}\right] dt = \int_{t_n}^{t_{n+1}} h_n(t)\cos\left[\left(k + \tfrac{1}{2}\right)\pi\frac{t - t_n}{t_{n+1} - t_n}\right] dt. \tag{8.38}$$

The latter integral moves to the interval $[0, 1]$ and is quickly computed by the DCT-IV. The folded function is

$$h_n(t) = \begin{cases} g_n(t)f(t) + g_{n-1}(t)f(2t_n - t) & \text{on } [t_n, t_n + a_n] \\ f(t) & \text{on } \left[t_n + a_n, t_{n+1} - a_{n+1}\right] \\ g_n(t)f(t) - g_{n+1}(t)f(2t_{n+1} - t) & \text{on } \left[t_{n+1} - a_{n+1}, t_{n+1}\right] \end{cases}$$

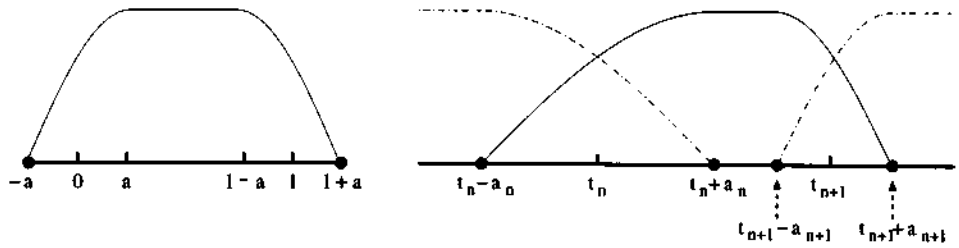



**Figure 8.11:** Window $g(t)$ and its non-uniform translates at $t_n$ and $t_{n+1}$.

This even-odd folding matches the even-odd property of $\cos(k+\frac{1}{2})\pi t$. An even-even extension would fail! (Unless it is alternated with odd-odd.) The construction is neat and efficient, and must be regarded as strong competition for wavelets. A fully detailed description is given by Wickerhauser [W]. It has a close relation to wavelets, as we now indicate.

**Sinc wavelets and Meyer wavelets** The local cosines give a splitting of the time axis. The sinc wavelets are also local cosines, but on the frequency axis. Remember from Section 2.3 that the sinc wavelet and its Fourier transform are

$$w(t) = \frac{\sin 2\pi t - \sin \pi t}{\pi t} = \phi(2t) - \phi(t)$$

$$\widehat{w}(\omega) = \begin{cases} 1 & \text{for } \pi \le |\omega| < 2\pi \\ 0 & \text{otherwise.} \end{cases}$$

When the wavelet is dilated and shifted to $w\left(2^j t - k\right)$, its Fourier transform is moved to the interval $2^j\pi \le |\omega| < 2^{j+1}\pi$. It is modulated by $\exp\left(-i\omega k/2^j\right)$. The transform becomes a local cosine, or rather a local exponential. It is not smooth in $\omega$ because the wavelet $w(t)$ does not decay quickly in time.

Note that the order of accuracy is $p = \infty$! The transform is *infinitely flat* at $\omega = 0$ and $\omega = \pi$. (The scaling function is a box function in $\omega$, the ideal lowpass filter.) This corresponds to the ***spectral method*** in the numerical solution of partial differential equations, which also achieves $p = \infty$. Finite differences and finite elements have finite $p$, like FIR filters. An infinite value of $p$ can only be achieved by functions $\phi(t)$ and $w(t)$ with infinite support. They are nonzero on the whole line, but the simplest sinc construction has slow decay.

Smoothing the local cosines (in $\omega$) will produce faster decay (in $t$). This leads to the Meyer wavelets, which are *windowed in frequency*. The window can have infinite smoothness in $\omega$, so the time decay can be faster than any power of $t$. The orthogonality of the sincs is maintained by the conditions on the windows. But windowing in time (to get smooth local cosines) is more practical.

## Biorthogonal Local Cosines

Every wavelet construction becomes more general when the orthogonality requirement is lifted. The same is true of local cosines. For *biorthogonal* bases, the sum of squares $S(t) = \sum_{-\infty}^{\infty}(g(t-n))^2$ of window functions is not necessarily one. Then a ***dual window*** is defined by

$$\widetilde{g}(t) = \frac{g(t)}{S(t)}. \tag{8.39}$$

The dual basis functions are the local cosines using the dual window:

$$\tilde{g}_{nk}(t) = \tilde{g}(t-n)\cos\left[\left(k+\tfrac{1}{2}\right)\pi\,(t-n)\right]$$

[Matv] gives a nice discussion of biorthogonality of the dual bases. This follows directly from orthogonality with window functions $g_{\text{orth}}(t-n) = g(t-n)/\sqrt{S(t)}$ on the interval $\left[-\frac{1}{2}, \frac{1}{2}\right]$. The reader sees immediately that the sum of $g^2_{\text{orth}}(t-n)$ is $S(t)/S(t) = 1$.

The condition number of the dual bases governs stability. This number is one for an orthonormal basis (using the earlier windows $g(t-n)$) and greater than one for a biorthogonal basis (using $g(t)$ and $\tilde{g}(t)$). The condition number naturally depends on the distance of $S(t)$ from 1. Matviyenko shows how to construct windows with $S_{\max}(t) \le 2S_{\min}(t)$. They give better compression than the orthogonal local cosines, and stability is well in control. He carries out an approximate optimization of $g(t)$ for the compression of sinusoids $\cos(\omega t + \alpha)$ of arbitrary frequency and phase, and tabulates the resulting windows. The graphs of $\tilde{g}(t)$ show a double peak which looks more alarming than it is.

The main point is *freedom versus constraints*, in the choice of window as in the choice of wavelets. Freedom gives the possibility of smooth windows. Constraints make those windows successful. A possible constraint is ***accuracy***: to reproduce exactly a constant function (or all polynomials of degree less than $p$). We can further limit the search to functions $g(t) = \sum c_k \cos\left[\left(k+\frac{1}{2}\right)\pi t\right]$ around $t = 0$, and ask for maximum smoothness at $t = \pm\frac{1}{2}$. This leads to "spline windows." The real tests of coding gain and compression are still ahead.

**Note.** The ***folding operator*** is defined above by $h(t) = \boldsymbol{F}f(t)$. There is also an ***unfolding operator***. When they use the same window, these operators $\boldsymbol{F}$ and $\boldsymbol{U}$ are transposes. The plus sign and minus sign on the first and third lines of $h(t)$ are reversed for the unfolded $u(t) = \boldsymbol{U}h(t)$. When $\boldsymbol{F}$ and $\boldsymbol{U}$ use dual windows $g(t)$ and $\tilde{g}(t)$, these operators are inverses. When they use the self-dual windows that have $S(t) = \sum g^2(t-n) \equiv 1$, the operators are both transposes and inverses. Therefore they are unitary operators. The families $\{\boldsymbol{F}_n\}$ and $\{\boldsymbol{U}_n\}$ which fold and unfold around $t = n$ (or more generally $t = t_n$) are the foundation for a full theory of smooth local cosines.

Jawerth, Liu, and Sweldens point out that in image compression, *folding can be seen as a preprocessing step for JPEG*. It is a generalized extension. The extension can be even or odd, symmetric or antisymmetric, and it changes the signal also *inside* the basic interval. The overlapping intervals allow perfect reconstruction — orthogonal or biorthogonal — of the signal.

### Problem Set 8.4

1. Why does the linear rise $g(t) = \frac{1}{2}\left(1 + \frac{t}{a}\right)$ in the interval $[-a, a]$ not lead to orthogonality? The problem is not lack of smoothness.
2. A rise function can be written as $g(t) = \sin\left(\theta\left(\frac{t}{a}\right)\right)$. Here $\theta(-1) = 0$ and $\theta(1) = \frac{\pi}{2}$. Show that $\theta(t) + \theta(-t) = \frac{\pi}{2}$ assures $(g(t))^2 + (g(-t))^2 = 1$.
3. Compute a cubic spline $\theta(t)$, increasing from $\theta(-1) = 0$ to $\theta(1) = \frac{\pi}{2}$, that meets the requirement in Problem 2.
4. (Thesis project) Experiment with several uniform windows $g(t)$ and several overlap lengths $a$ to optimize compression. Compare with the blocking effects from a window box.

5. Find the parity (even or odd?) of the function $\sin\left(k+\frac{1}{2}\right)\pi t$ around the endpoints of $[0, 1]$. This is the sine-IV basis used in the DST-IV transform.

6. Show that the conditions on the uniform windows $g(t-n)$ can be written as

$$\begin{array}{lll} g^2(t)+g^2(-t)=1 & -\frac{1}{2} \le x \le \frac{1}{2} \\ g(t)=g(1-t) & \frac{1}{2} \le x \le \frac{3}{2} \\ g(t)=0 & \text{elsewhere.} \end{array}$$

   This yields $g(t) \equiv 1$ in $[a, 1-a]$ where there is no overlap.

7. (a) Draw a linear function $f(t)$ on $[0, 1]$ and its folding $h(t) = Ff(t)$.

   (b) Draw a linear function $h(t)$ on $[0, 1]$ and its unfolding $u(t) = Uh(t)$.

8. Explain (8.34) by following the proof of (8.33).

## 8.5 Boundary Filters and Wavelets

The striking fact about FIR filter banks is that ***the banded analysis matrix has a banded inverse.*** That inverse is the synthesis matrix.

Banded matrices correspond to FIR filters and compactly supported wavelets. The basis functions have finite length. The matrices may even be orthogonal. The question is how to maintain these properties at a boundary, by adding new rows and new functions:

*Boundary filters are the* "end rows" *of an* $L \times L$ *filter matrix* $\boldsymbol{H}_L$.

*Boundary scaling functions and wavelets are the* "end functions" *of a basis.*

If we construct boundary filters, the dilation equation yields boundary functions. The reverse direction also succeeds. The functions have dilation coefficients that go into the filters. All constructions are in the time domain, because transform methods have major difficulty at boundaries. We begin with boundary filters (completing a matrix) and then create boundary functions (completing a basis). Those are really equivalent.

The matrices begin with the lowpass $\boldsymbol{H}_0$ and the highpass $\boldsymbol{H}_1$, subsampled. These are infinite $1 \times 2$ block Toeplitz matrices. The rows have a double shift from $(\downarrow 2)$. *We interleave the filters* $(\downarrow 2)\boldsymbol{H}_0$ *and* $(\downarrow 2)\boldsymbol{H}_1$ *to produce* $2 \times 2$ *blocks.* Then there is an ordinary Toeplitz shift of *one block* between rows of $\boldsymbol{H}_b$:

$$\boldsymbol{H}_b = \begin{bmatrix} \cdots & \cdots & & & & & & \\ h_0(N) & h_0(N-1) & \cdots & \cdots & h_0(1) & h_0(0) & & \\ h_1(N) & h_1(N-1) & \cdots & \cdots & h_1(1) & h_1(0) & & \\ & & h_0(N) & h_0(N-1) & \cdots & \cdots & h_0(1) & h_0(0) \\ & & h_1(N) & h_1(N-1) & \cdots & \cdots & h_1(1) & h_1(0) \\ & & & & & & \cdots & \cdots \end{bmatrix}$$

Chapter 9 will have $M$ filters and $(\downarrow M)$. Then $\boldsymbol{H}_b$ has $M \times M$ blocks.

The synthesis filters have $F_0(z) = H_1(-z)$ and $F_1(z) = -H_0(-z)$. In the orthogonal case, $F_0$ is the ordinary flip (the transpose) of $H_0$. Therefore $H_1$ is the alternating flip. The key point is that both polyphase matrices, $\boldsymbol{H}_p(z)$ in analysis and $\boldsymbol{H}_p^{-1}(z)$ in synthesis, are *polynomial.* The determinant of $\boldsymbol{H}_p(z)$ is $z^{-N}$, by the halfband condition on $H_0(z)H_1(-z)$ that makes everything work.

When signals have finite length $L$, the infinite matrix $\boldsymbol{H}_b$ must change to $L \times L$. We assume that $L > N$ and probably $L \gg N$. Then the "middle" of the matrices is not affected, but the "ends" will be new. We have to choose those end rows — the *boundary filters* in the $L \times L$ matrix $\boldsymbol{H}$. The first question is whether FIR boundary filters can yield $\boldsymbol{H}^{-1}\boldsymbol{H} = \boldsymbol{I}$.

The answer is *yes*. The inverse matrix stays banded. Then the problem is to choose among boundary filters. We will indicate some reasonable choices, but more experiments and experience are needed. This is a research problem of great interest.

In the orthogonal case, the boundary filters are to preserve $\boldsymbol{H}^T\boldsymbol{H} = \boldsymbol{I}$. The middle filters (interior filters) have this orthogonal property — the middle is unchanged from $\boldsymbol{H}_b^T\boldsymbol{H}_b = \boldsymbol{I}$ in the infinite case. We have to be sure that the new end rows of $\boldsymbol{H}$ and the new end columns of $\boldsymbol{H}^{-1}$ have limited length $\leq N$ and not full length $L$.

Wavelets have similar difficulties. When $\phi(t-n)$ and $w(t-n)$ fall across the boundary, they must change. This happens at each scale, and the multiresolution $V_0 \subset V_1 \subset \cdots$ should be maintained. The boundary functions are to be combinations of internal functions and boundary functions at scale $2t$. The shapes are the same at all scales. The coefficients in that boundary dilation equation and boundary wavelet equation give the boundary filters!

### Filter Bank Completions

Our goal is an $L \times L$ analysis matrix $\boldsymbol{H}_L$. The synthesis matrix will be its inverse. In the orthogonal case, which we emphasize most, this inverse is $\boldsymbol{H}_L^T$.

$\boldsymbol{H}_L$ begins with rows from the infinite filter matrix. For the interior part $\boldsymbol{H}_{\text{in}}$, we only save *complete rows*. Our example will be the Daubechies 4-tap filter with coefficients $(a, b, c, d) = \big(1+\sqrt{3}, 3+\sqrt{3}, 3-\sqrt{3}, 1-\sqrt{3}\big)/4\sqrt{2}$. For convenience we save only *one* lowpass and highpass row:

$$\boldsymbol{H}_{\text{in}} = \begin{bmatrix} 0 & d & c & b & a & 0 \\ 0 & -a & b & -c & d & 0 \end{bmatrix}.$$

These rows are orthonormal. Therefore $(\boldsymbol{H}_{\text{in}})(\boldsymbol{H}_{\text{in}}^T) = \boldsymbol{I}$. The task is to add four new orthonormal rows — the boundary filters — to achieve $(\boldsymbol{H}_L)(\boldsymbol{H}_L^T) = \boldsymbol{I}$ with square matrices.

Those four filters will separate into left end and right end, and into lowpass and highpass. The square matrix will have the form

$$\boldsymbol{H}_L = \begin{bmatrix} \boldsymbol{H}_{\text{left}} \\ \boldsymbol{H}_{\text{in}} \\ \boldsymbol{H}_{\text{right}} \end{bmatrix} = \begin{bmatrix} r & s & t & 0 & 0 & 0 \\ u & v & w & 0 & 0 & 0 \\ 0 & d & c & b & a & 0 \\ 0 & -a & b & -c & d & 0 \\ 0 & 0 & 0 & e & f & g \\ 0 & 0 & 0 & x & y & z \end{bmatrix}. \tag{8.40}$$

Notice the freedom that has been removed and the freedom that remains. The boundary filters are allowed only three coefficients. At first this seems difficult, because the short rows $(r, s, t)$ and $(u, v, w)$ must be orthogonal to $(0, d, c)$ and also $(0, -a, b)$. Fortunately those vectors are parallel! The product $bd + ac$ is zero, from the key property of the Daubechies coefficients — that $(a, b, c, d)$ is orthogonal to its double shift. The Gram-Schmidt process can easily produce $(r, s, t)$ and $(u, v, w)$. Similarly we find the two short boundary filters at the right end, which go in the last two rows of $\boldsymbol{H}_L$.

What freedom remains? We can premultiply the first two rows by any $2 \times 2$ orthogonal matrix. We use that freedom to make $u + v + w = 0$; that boundary filter becomes highpass. Similarly $x + y + z = 0$ makes the last row orthogonal to DC inputs (constant inputs), which therefore go through the lowpass channel. The actual coefficients are

$$(r, s, t) = (\ 0.93907,\ 0.29767,\ -0.17186) \qquad (e, f, g) = (0.40345,\ 0.69879,\ \ 0.59069)$$
$$(u, v, w) = (-0.34372,\ 0.81326,\ -0.46954) \qquad (x, y, z) = (0.25535,\ 0.51155,\ -0.80690).$$

In practice the interior matrix $\boldsymbol{H}_{\text{in}}$ has many more rows ($L - 4$ rows). These inner rows are not truncated and remain safely orthogonal. They separate the left end from the right end.

We now show that a similar construction of boundary filters will succeed for other (longer) interior filters. It is almost simple enough to do by hand. Herley gives a MATLAB code. The starting point is $(\boldsymbol{H}_{\text{in}})(\boldsymbol{H}_{\text{in}}^T) = \boldsymbol{I}$. Remember that $\boldsymbol{H}_{\text{in}}$ is rectangular. The product $(\boldsymbol{H}_{\text{in}}^T)(\boldsymbol{H}_{\text{in}})$ in the opposite order cannot equal $\boldsymbol{I}$. But the discrepancy is all near boundaries:

$$\boldsymbol{P} = \boldsymbol{I} - (\boldsymbol{H}_{\text{in}}^T)(\boldsymbol{H}_{\text{in}}) = \begin{bmatrix} \boldsymbol{P}_{\text{left}} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \boldsymbol{P}_{\text{right}} \end{bmatrix}. \tag{8.41}$$

This matrix satisfies $\boldsymbol{H}_{\text{in}}\boldsymbol{P} = \boldsymbol{H}_{\text{in}} - (\boldsymbol{H}_{\text{in}})(\boldsymbol{H}_{\text{in}}^T)(\boldsymbol{H}_{\text{in}}) = \boldsymbol{0}$. ***The columns of $\boldsymbol{P}_{\text{left}}$ and $\boldsymbol{P}_{\text{right}}$ are orthogonal to the rows of $\boldsymbol{H}_{\text{in}}$.*** These columns with short vectors (boundary filters) can complete a full set of orthogonal rows in $\boldsymbol{H}_L$. The rank of $\boldsymbol{P}$ is necessarily the difference between the number of columns and rows in $\boldsymbol{H}_{\text{in}}$. This is the correct number to complete a square matrix.

For the $2 \times 6$ example $\boldsymbol{H}_{\text{in}}$ we find a $6 \times 6$ matrix $\boldsymbol{P}$:

$$\boldsymbol{P} = \begin{bmatrix} 1 & 0 & 0 & & & \\ 0 & d^2 + a^2 & cd - ab & & & \\ 0 & cd - ab & c^2 + b^2 & & & \\ & & & b^2 + c^2 & ab - cd & 0 \\ & & & ab - cd & a^2 + d^2 & 0 \\ & & & 0 & 0 & 1 \end{bmatrix}.$$

The double-shift orthogonality $ac + bd = 0$ gave the zero blocks. Here $\boldsymbol{P}_{\text{left}}$ is $3 \times 3$ but its rank must be 2. The first two columns (or rows) of $\boldsymbol{P}$ are orthogonal to the rows of $\boldsymbol{H}_{\text{in}}$. ***They give two left boundary filters.*** After normalization to unit length, and rotation to make the second one highpass, they become $(r, s, t)$ and $(u, v, w)$.

Some patience is needed to find the right number of zero columns for $\boldsymbol{H}_{\text{in}}$. For $D_4$, one zero column at each end gave two boundary filters at each end. [HerVet] propose a way to keep this number even for the Daubechies filters $D_{2p}$. We illustrate for $D_6$ a slightly different approach, which keeps four rows of $\boldsymbol{H}_{\text{in}}$ with no zero columns:

$$\boldsymbol{H}_{\text{in}} = \begin{bmatrix} f & e & d & c & b & a & & \\ -a & b & -c & d & -e & f & & \\ & & f & e & d & c & b & a \\ & & -a & b & -c & d & -e & f \end{bmatrix}. \tag{8.42}$$

Thus $\boldsymbol{H}_{\text{in}}$ is $4 \times 8$ (and $8 - 4$ is a multiple of 4). Using $(\boldsymbol{H}_{\text{in}})(\boldsymbol{H}_{\text{in}}^T) = \boldsymbol{I}$, we find $4 \times 4$ blocks in $\boldsymbol{P}$:

$$\boldsymbol{P} = \boldsymbol{I} - (\boldsymbol{H}_{\text{in}}^T)(\boldsymbol{H}_{\text{in}}) = \begin{bmatrix} \boldsymbol{P}_{\text{left}} & 0 \\ 0 & \boldsymbol{P}_{\text{right}} \end{bmatrix}. \tag{8.43}$$

$P$ has rank 4 and each block has rank 2. We obtain two 4-tap boundary filters at each end. When $H_L$ has more inner filters, for signal lengths $L > 8$, they start and end with at least four zeros. This makes all inner filters orthogonal to our boundary filters. The $L \times L$ orthogonal analysis bank is complete.

**General case:** Biorthogonal filters have $(H_{\text{in}})(F_{\text{in}}) = I$ but not $(F_{\text{in}})(H_{\text{in}}) = I$. Again these are rectangular matrices with no truncation of the interior filters. We need boundary filters to complete square matrices. Here is the key:

$$P = I - (F_{\text{in}})(H_{\text{in}}) \quad \text{has} \quad (H_{\text{in}})P = H_{\text{in}} - H_{\text{in}}F_{\text{in}}H_{\text{in}} = 0.$$

The columns of $P$ contain boundary filters that can be included with the columns of $F_{\text{in}}$. Similarly $P(F_{\text{in}}) = 0$. The rows of $P$ contain boundary filters that are included with the rows of $H_{\text{in}}$. Finally we biorthogonalize these boundary filters. A useful identity (to put it mildly) is $P^2 = P$.

### Time-varying Filter Banks: A Remark

Changing from length 4 filters to length 6 will create an internal boundary, at the moment of change. One way to maintain orthogonality is to use boundary filters just before the change and just after. Such a transition has no overlap. A smoother approach is to create *transition filters* that cross the internal boundary. In this case $H_{\text{in}}$ has filters purely to the left and purely to the right.

Again $(H_{\text{in}})(H_{\text{in}}^T) = I$. Now $P = I - H_{\text{in}}^T H_{\text{in}}$ contains the transition filters. See [HeKoRaVe] for details of this important construction.

### Direct Extrapolation at the Boundary

We briefly mention another approach to boundary filters. It is closer to the usual treatment of boundary conditions for differential equations. *The governing principle is to fit a polynomial to the data and extend that polynomial.* The job of extrapolation is to define functions and signals beyond the boundary — so the filter can be applied.

Zero-padding and symmetric extension are basic extrapolation methods. Higher degree polynomials give higher degree extrapolation and higher accuracy.

To be consistent with multiresolution, the input $x(n)$ should give two half-length outputs. If the inputs are coefficients of $\phi(2t - n)$ in a function $f(t)$, the output are coefficients of $\phi(t - n)$ and $w(t - n)$ for $f(t)$ at the next scale. In the interior all is normal. *At the boundary we determine a polynomial $F_{p-1}(t)$ of degree $p - 1$ that models $f(t)$.* Using only the $p$ numbers $x(0), \ldots, x(p-1)$ to determine the $p$ coefficients in the polynomial gives a square matrix — but this is dangerous. The preference in [WiAm] is to use more inputs $x(0), \ldots, x(N)$ and determine the polynomial coefficients by least squares.

By extending the polynomial we extrapolate the signal. Then filter and downsample as usual. This corresponds to a splitting of $V_1$ into $V_0 + W_0$. It also corresponds to a completion of the filter matrix near the boundary. For the lowpass Daubechies $D_4$, new coefficients appear in the first row of $H_L$. The second lowpass row contains the usual $d, c, b, a$:

$$\begin{bmatrix} 0.8924 & 0.5174 & 0.0129 & -0.0085 & 0 & 0 \\ -0.1294 & 0.2241 & 0.8365 & 0.4830 & 0 & 0 \\ & & \cdots & \cdots & \cdots & \cdots \end{bmatrix}.$$

Orthogonality is lost; accuracy is preserved; highpass coefficients are in [WiAm]. Also important in differential equations is the fact that *boundary conditions can be built into the extrapolation.*

There is a third "quick and dirty" approach that deals entirely with the matrix entries. We follow [KwTa] by considering a biorthogonal example with attractive coefficients:

$$\boldsymbol{H} = \tfrac{1}{4}\begin{bmatrix} -1 & 3 & 3 & -1 & & \\ -1 & 3 & -3 & 1 & & \\ & & -1 & 3 & 3 & -1 \\ & & -1 & 3 & -3 & 1 \\ & & & & & \cdots \end{bmatrix} \quad \text{and} \quad \boldsymbol{H}^{-1} = \tfrac{1}{4}\begin{bmatrix} 1 & 1 & & & \\ 3 & 3 & & & \\ 3 & -3 & 1 & 1 & \\ 1 & -1 & 3 & 3 & \\ & & 3 & -3 & \cdots \\ & & 1 & -1 & \cdots \end{bmatrix}. \tag{8.44}$$

The column coefficients $1, 3, 3, 1$ in $\boldsymbol{H}^{-1}$ come from $(1+z^{-1})^3$. The synthesis scaling function is a quadratic spline (accuracy $p = 3$). The row coefficients $-1, 3, 3, -1$ come from $(1+z^{-1})(-1+4z^{-1}-z^{-2})$. This single zero gives $\widetilde{p} = 1$. The product filter is the familiar halfband $P(z)$ of degree 6. This is a linear phase choice, not to be iterated too often! We noted in Section 7.2 that Condition **E** is violated. The cascade algorithm will not converge for the filter $\boldsymbol{H}$.

Nevertheless it is a good example. Suppose the second column shown for $\boldsymbol{H}$ corresponds to the sample $x(0)$. The first column would normally multiply $x(-1)$, *which does not exist.* Extrapolation could create $x(-1)$. Equivalently, this external column $\boldsymbol{c} = -\frac{1}{4}, -\frac{1}{4}$ can be folded into the internal columns by one of these rules:

***Zero-padding***: Delete the external column $\boldsymbol{c}$.

***Constant extrapolation***: Add $\boldsymbol{c}$ to the zeroth column $(3/4, 3/4)$.

***Linear extrapolation***: Add $2\boldsymbol{c}$ to column 0 and $-\boldsymbol{c}$ to column 1.

***Quadratic extrapolation***: Add $3\boldsymbol{c}, -3\boldsymbol{c}, \boldsymbol{c}$ to columns 0, 1, 2.

The key point is that $\boldsymbol{H}$ and $\boldsymbol{H}^{-1}$ both maintain their stacked double-shift form (rows of $\boldsymbol{H}$ and columns of $\boldsymbol{H}^{-1}$). The boundary filters based on constant extrapolation $x(-1) = x(0)$, which adds $\boldsymbol{c}$ to column zero, keep their lowpass and highpass character:

$$\boldsymbol{H}_L = \tfrac{1}{4}\begin{bmatrix} 2 & 3 & -1 & & & \\ 2 & 3 & -1 & & & \\ & -1 & 3 & 3 & -1 & \\ & -1 & 3 & -3 & 1 & \\ & & & & \cdots & \end{bmatrix} \quad \text{and} \quad \boldsymbol{H}_L^{-1} = \tfrac{1}{4}\begin{bmatrix} 4 & 4 & & & \\ 3 & -3 & 1 & 1 & \\ 1 & -1 & 3 & 3 & \\ & & 3 & -3 & \cdots \\ & & 1 & -1 & \cdots \end{bmatrix}. \tag{8.45}$$

For longer filters the extrapolation rules are similar, but care is needed. There is still much room for experiment and comparison of boundary filters.

### Lowpass Cascade and the Boundary Dilation Equation

Scaling functions come from lowpass filters. The lowpass coefficients go into the dilation equation. This equation is solved for $\phi(t)$ by iteration — which means cascade. A good example is the synthesis filter bank $\boldsymbol{H}_L^{-1}$ given above. Its interior lowpass columns contain $(\frac{1}{4}, \frac{3}{4}, \frac{3}{4}, \frac{1}{4})$ and those coefficients lead to the quadratic spline $\phi(t)$ on $[0, 3]$:

$$\phi(t) = \tfrac{1}{4}\phi(2t) + \tfrac{3}{4}\phi(2t-1) + \tfrac{3}{4}\phi(2t-2) + \tfrac{1}{4}\phi(2t-3). \tag{8.46}$$

This spline has an explicit formula, which we mention but do not need. $2\phi(t)$ equals $t^2$ and $-2t^2 + 6t - 3$ and $(t-3)^2$ on the intervals $[0, 1]$ and $[1, 2]$ and $[2, 3]$.

The boundary filter $(1, \frac{3}{4}, \frac{1}{4})$ in the first column of $\boldsymbol{H}^{-1}$ leads to a boundary scaling function $\phi_b(t)$. The boundary dilation equation uses those coefficients:

$$\phi_b(t) = \phi_b(2t) + \tfrac{3}{4}\phi(2t) + \tfrac{1}{4}\phi(2t-1) \quad \text{for} \quad t \geq 0. \tag{8.47}$$

Notice how $\phi_b$ is included with the functions at scale $2t$. Since $\phi(2t)$ and $\phi(2t-1)$ are already known, this is an "*inhomogeneous dilation equation*" for the boundary function $\phi_b(t)$. It can be solved in at least three ways:

1. By creating a general method for inhomogeneous two-scale equations.

2. By an inspired guess.

3. By iteration, cascading the lowpass filter.

We choose Method 2, because the insight is important. The boundary filter comes from constant extrapolation, which preserves minimum accuracy $p = 1$. The constant function will belong to $V_0$. In the interior we do have $\sum \phi(t-n) \equiv 1$. But near $t = 0$, the pieces from $\phi(t+1)$ and $\phi(t+2)$ are missing — those functions cross the boundary. It is natural to suspect that the boundary function $\phi_b(t)$ takes their place — but only on the right side $t \geq 0$:

$$\text{Inspired guess:} \quad \phi_b(t) = \phi(t+1) + \phi(t+2) \quad \text{for} \quad t \geq 0. \tag{8.48}$$

The support is $[0, 2]$. We try this function in the boundary dilation equation (8.47). On the right side it contributes $\phi(2t+1) + \phi(2t+2)$. On the left side, replace $\phi(t+1)$ and $\phi(t+2)$ using the interior dilation equation (8.46). Compare only for $t \geq 0$, so terms involving $\phi(2t+3)$ and $\phi(2t+4)$ can be and must be ignored. Equation (8.47) is satisfied!

This function $\phi_b(t) = \sum_{n<0} \phi(t-n)$ will appear again, as a direct construction in continuous time. Figure 8.12 shows how it builds $f(t) \equiv 1$ into $V_0$.

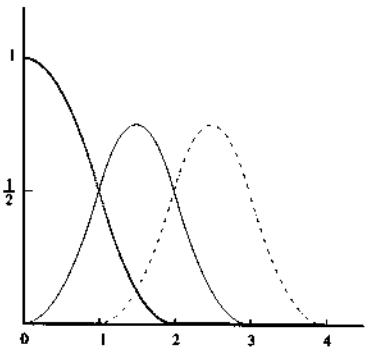



**Figure 8.12:** Quadratic splines and $\phi_b(t)$ add to 1.

Its dilation equation reveals the boundary filter. In that order, $\phi_b(t)$ produces the filter matrix rather than vice versa.

It is helpful also to see the lowpass cascade (Method 3). This means powers of the double-shift lowpass synthesis matrix:

$$\boldsymbol{L} = \tfrac{1}{4}\begin{bmatrix} 4 & 3 & 1 & & & & & \\ & 1 & 3 & 3 & 1 & & & \\ & & & 1 & 3 & 3 & 1 & \cdots \end{bmatrix}.$$

Squaring leaves the unit column sums, and produces a *quadruple shift*:

$$\boldsymbol{L}^2 = \tfrac{1}{16}\begin{bmatrix} 16 & 15 & 13 & 10 & 6 & 3 & 1 & \\ & 1 & 3 & 6 & 10 & 12 & 12 & \cdots \\ & & & & & 1 & 3 & \cdots \end{bmatrix}.$$

The boundary filter affects only the first row. Elsewhere, convolving 1, 3, 3, 1 with 1, 0, 3, 0, 3, 0, 1 gives the interior sequence 1, 3, 6, 10, 12, 12, 10, 6, 3, 1. This corresponds to $H(z)H(z^2)$ — which is not $[H(z)]^2$. *The cascade includes rescaling.* $\boldsymbol{L}$ relates to integer spacing $\Delta t = 1$, while $\boldsymbol{L}^2$ relates to $\Delta t = \frac{1}{2}$. The fact that all column sums are 1 means that, at convergence, the scaling functions $\phi_b(t), \phi(t), \phi(t-1), \ldots$ add to 1. This brings back our formula (8.9) for $\phi_b(t)$.

It is unusual to solve a dilation equation so explicitly. But convergence of the cascade is not difficult to establish. Condition E holds for the interior filter. The boundary requires Condition $\mathbf{E}_b$ on a small square matrix in the corner of $\boldsymbol{L}$. That $2 \times 2$ matrix has eigenvalues 1 and $\frac{1}{4}$ and the cascade converges.

For this example, the powers $\boldsymbol{L}^j$ are the slow way to solve the dilation equation. *For most examples $\boldsymbol{L}^j$ is the only way.* The cascade algorithm is also called the "graphical algorithm," because a finite power of $\boldsymbol{L}$ yields an approximate graph of the scaling functions. Then one application of the other submatrix $\boldsymbol{B}$ (from the highpass filters) produces the boundary and interior wavelets.

## Basis Completion by Boundary Functions

For a direct construction of boundary functions, suppose $\phi(t)$ is supported on $[0, 3]$. Our overall interval is finite, say $[0, 4]$. Then we can only shift once, to $\phi(t-1)$. All other translates of $\phi(t)$ cross the boundary and have to be changed. We are looking for four basis functions in $V_0$ and $4 \cdot 2^j$ basis functions in $V_j$. There are three important ways to construct them [CoDaVi]:

**1. Periodic Extension:** The shifted function $\phi(t-2)$ is supported on $[2, 5]$. The segment on $[4, 5]$ is wrapped back to $[0, 1]$. Similarly, the piece of $\phi(t-3)$ on $[4, 6]$ is wrapped back to $[0, 2]$. We are just periodizing the original functions.

This corresponds in the discrete case to *circulant matrices*. The lowpass analysis filter is a $4 \times 4$ circulant containing the four filter coefficients. It stretches to $4 \times 8$ with double shift from $(\downarrow 2)$. The highpass filter, similarly stretched, completes the $8 \times 8$ block circulant $\boldsymbol{H}_L$.

**2. Symmetric Extension:** This would apply to the linear phase example based on the coefficients $(1, 3, 3, 1)$. An even-length symmetric filter (H filter) requires an H extension, centered at half-samples. The detailed discussion is in Section 8.2. Our point here is that the extension is implicitly creating boundary filters.

Symmetric extension is used for symmetric scaling functions. Those are not self-orthogonal; we are in the biorthogonal case. Our 1, 3, 3, 1 example has an even number of taps, equal in analysis and synthesis. The scaling function $\phi(t+1)$ on $[-1, 2]$ extends outside the basic interval $[0, 4]$, and is ***folded*** back inside by a symmetric extension to the whole line:

$$f^{\text{fold}}(t) = \sum_{-\infty}^{\infty} [f(t - 8n) + f(8n - t)]. \tag{8.49}$$

The terms with $n = 0$ are $f(t) + f(-t)$, even across $t = 0$. The other terms keep this symmetry and produce the double period 8. Thus any folded function is even with period 8:

$$f^{\text{fold}}(-t) = f^{\text{fold}}(t) \quad \text{and} \quad f^{\text{fold}}(t + 8) = f^{\text{fold}}(t). \tag{8.50}$$

Figure 8.13 shows an idealized (!) picture of a symmetric $\phi(t)$ extended to the one-period in-

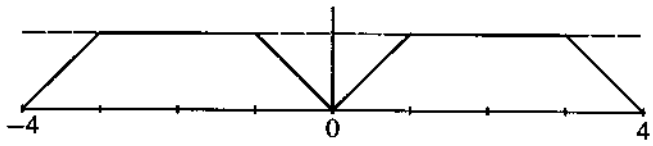



**Figure 8.13:** Symmetric folding of four symmetric scaling functions.

terval $[-4, 4]$. Their restrictions to $[0, 4]$ span the space $V_0$ with the correct dimension $4 \cdot 2^j = 4$. These are the four scaling functions and there are four corresponding wavelets. Similarly there will be four synthesis functions $\widetilde{\phi}(t)$ and wavelets $\widetilde{w}(t)$. *Furthermore biorthogonality is preserved* (Problems 1-2).

The original scaling functions $\phi(t + n)$ on the whole line add to the constant function 1, because the lowpass filter has the required zero at $\pi$. This good property is preserved by symmetric extension. *The folded functions still add to* 1. In our idealized figure the flat middle segments have height $\frac{1}{2}$. This piecewise linear $\phi(t)$ comes from a simple filter that cannot be part of an FIR filter bank (Problems 3–4). In general all the pieces of scaling functions that add to 1 are folded back into the interval so constants are in $V_0$.

Even if $H(z)$ has extra zeros at $\pi$, we do not expect $V_0$ to give accuracy beyond $p = 1$. Symmetric extension (folding) generally produces jumps in the slope at the boundary. We turn to general methods that can achieve $p > 1$.

**3. Boundary Functions:** We need a completion method for the orthogonal case, when symmetric extension is unnatural. We also want to maintain the accuracy $p$ of the internal functions. The following method will apply to all cases, orthogonal or not and linear phase or not. *We focus on completing the lowpass synthesis part* (the space $V_0$ and the filter $\boldsymbol{F}_0$). A parallel construction applies to the analysis space $\widetilde{V}_0$ and the filter $\boldsymbol{H}_0$. Then multiresolution makes $W_0$ and $\widetilde{W}_0$ orthogonal to $\widetilde{V}_0$ and $V_0$.

The plan is to start with internal scaling functions supported inside the given interval, and add new functions at each end. To start, the new ones are linearly independent "*boundary pieces.*" We insist on short support and a dilation equation, so that $V_0 \subset V_1$. Additional properties lead to full accuracy. ***Those short pieces are orthogonalized against the existing scaling function and wavelets.*** In the orthogonal case those are the internal $\phi(t - n)$ and $w(t - n)$; in general they

are the synthesis functions. The algorithm is just Gram-Schmidt or its biorthogonal extension to "bi-Gram-Schmidt."

The key point is that this orthogonalization produces functions supported *near the boundary*. Each boundary piece, however chosen, is already orthogonal to all scaling functions that are far inside the interval. There is no overlap. One source of boundary pieces is the tails of existing $\phi(t+k)$ and $w(t+k)$. Numerically this is a poor choice. The tails are very small for longer wavelets, which leads to severe ill-conditioning.

A better construction [CoDaVi] is to build polynomials up to degree $p-1$ into $V_0$. On the whole line we have $\sum \phi(t-n) \equiv 1$. Then the first boundary piece ($t \geq 0$ only) is

$$\phi_b(t) = 1 - \textstyle\sum_{n\geq 0} \phi(t-n) = \sum_{n<0} \phi(t-n). \tag{8.51}$$

This is just a *sum of tails*. It is orthogonal to all internal $\phi(t-n)$, $n \geq 0$. Its support is $[0, N-1]$ because $\phi(t), \phi(t-1), \ldots$ already add to 1 beyond $t = N-1$.

The boundary function $\phi_b(t)$ satisfies a dilation equation. Replacing $t$ by $2t$ gives $\phi_b(2t) = 1 - \sum_{n\geq 0} \phi(2t-n)$. Use this to substitute for 1 in (8.51):

$$\phi_b(t) = \phi_b(2t) + \sum_{n\geq 0} [\phi(2t-n) - \phi(t-n)]. \tag{8.52}$$

Replace each $\phi(t-n)$ using its ordinary dilation equation. Thus $\phi_b(t)$ is a combination of $\phi_b(2t)$ and internal $\phi(2t-n)$. It belongs to $V_1$ and we are on our way.

**Higher Accuracy.** Suppose the original scaling function $\phi(t-n)$ can reproduce linear functions. The filter has at least two zeros at $\pi$. The wavelets have two vanishing moments. The combination $\sum n\phi(t-n)$ equals $\alpha t + \beta$; the constants $\alpha \neq 0$ and $\beta$ are determined by the filter coefficients. Our task is to ensure that $\alpha t + \beta$ is in $V_0$ when we remove all $\phi(t-n)$ for $n < 0$. In their place goes $\phi_b(t)$ and a second boundary piece $\phi_{bb}(t)$:

$$\phi_{bb}(t) = \alpha t + \beta - \sum_{n\geq 0} n\phi(t-n) = \sum_{n<0} n\phi(t-n). \tag{8.53}$$

The sum over $n < 0$ shows that the support is $[0, N-1]$. This piece need not be orthogonal to $\phi_b(t)$ or to the internal $\phi(t-n)$. We must orthogonalize it (or biorthogonalize it). Also we check that $\phi_{bb}(t)$ satisfies a dilation equation, so the multiresolution hierarchy $V_0 \subset V_1$ is still secure. Replacing $t$ by $2t$ in (8.53) gives

$$\phi_{bb}(2t) = 2\alpha t + \beta - \sum_{n\geq 0} n\phi(2t-n). \tag{8.54}$$

Divide by 2 and substitute for $\alpha t$ in (8.54):

$$\phi_{bb}(t) = \tfrac{1}{2}\phi_{bb}(2t) + \tfrac{1}{2}\beta + \sum_{n\geq 0} n\left[\tfrac{1}{2}\phi(2t-n) - \phi(t-n)\right]. \tag{8.55}$$

Replace each $\phi(t-n)$ using its dilation equation, and replace $\frac{1}{2}\beta$ by $\left[\frac{1}{2}\beta\phi_b(2t) + \sum \phi(2t-n)\right]$. Then (8.55) guarantees that $\phi_{bb}(t)$ is in the next space $V_1$.

It is important to count the functions! With $p$ zeros at $\pi$, we are choosing $p$ boundary pieces at each end to build the polynomials $1, \ldots, t^{p-1}$ into $V_0$. An interval of length $L$ will have $L - 2p + 2$ internal Daubechies functions. For $L = 4$ and $p = 2$ we had $\phi(t)$ on $[0, 3]$ and $\phi(t-1)$ on $[1, 4]$. *We only need* $2p-2$ *additional functions, but we have* $2p$.

To make space for $2p$ boundary pieces, we throw out the perfectly good $\phi(t)$ at the left end and the outermost scaling function at the right end. The term with $n = 0$ moves to the other side of (8.51). This only means that $\phi_b(t)$ has support $[0, N]$. A similar replacement at the other end assures that constant functions are in $V_0$.

The simplest highpass construction starts with scaling functions $\phi(2t - n)$ that cross the boundary. Subtract their projections on $V_0$ to get *boundary wavelets* [CoDaVi, p. 73]. Together with the internal wavelets this produces $W_0 \perp V_0$ and $W_0 \subset V_1$. We indicate the $D_4$ case with $p = 2$ and normal Daubechies coefficients $(a, b, c, d)$:

$$\boldsymbol{F}_{\text{low}} = \begin{bmatrix} \times & \times & \times & & & & & & & & \\ \times & \times & \times & \times & \times & & & & & & \\ & & & d & c & b & a & & & & \\ & & & & & \times & \times & \times & \times & \times & \\ & & & & & & & \times & \times & \times & \end{bmatrix}. \tag{8.56}$$

The $5 \times 10$ highpass matrix completes the $10 \times 10$ synthesis bank.

After orthogonalization, the filter coefficients are computed once and for all by [CoDaVi]. Those boundary filters are available from netlib@research.att.com with the message `send/stat/data/wavelets`.

### Problem Set 8.5

1. Verify from (8.49) that folding plus restriction preserves inner products:

$$\int_0^4 f^{\text{fold}}(t) g^{\text{fold}}(t)\, dt = \int_{-\infty}^{\infty} f(t) g(t)\, dt.$$

2. If the functions $\phi(t-n)$ and $\widetilde{\phi}(t-n)$ are biorthogonal on the whole line, verify that their folded versions are still biorthogonal on the finite interval. This interval is $[0, 4]$ when the folding in (8.49) has period 8.

3. What symmetric 4-tap filter would produce the piecewise linear $\phi(t)$ shown in Figure 8.13. How many zeros at $\pi$ in $H(\omega)$?

4. Show that the filter with $h = (\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4})$ has $H(i) = H(-i) = 0$. It cannot be part of a PR filter bank! Why can no synthesis filter make $P(z) = F(z)H(z)$ into a halfband filter with $P(z) + P(-z) = z^l$?

5. Take two rows of (8.44) in $\boldsymbol{H}_{\text{in}}$ and two columns of the inverse matrix in $\boldsymbol{F}_{\text{in}}$. Verify $\boldsymbol{H}_{\text{in}}\boldsymbol{F}_{\text{in}} = \boldsymbol{I}_{2\times 2}$ and compute $\boldsymbol{P} = \boldsymbol{I} - \boldsymbol{F}_{\text{in}}\boldsymbol{H}_{\text{in}}$ (which is $6 \times 6$). Determine a set of boundary filters.

6. From the double-shift orthogonality of Daubechies $D_6$, verify that the zero blocks of $\boldsymbol{P}$ in equation (8.43) really are zero.

7. Suppose $\boldsymbol{H}_{\text{in}}$ is $K \times L$ and $\boldsymbol{H}_{\text{in}}\boldsymbol{H}_{\text{in}}^T = \boldsymbol{I}_{K\times K}$. Verify that $\boldsymbol{Q} = \boldsymbol{H}_{\text{in}}^T\boldsymbol{H}_{\text{in}}$ and $\boldsymbol{P} = \boldsymbol{I} - \boldsymbol{Q}$ satisfy $\boldsymbol{Q}^2 = \boldsymbol{Q}$ and $\boldsymbol{P}^2 = \boldsymbol{P}$ and $\boldsymbol{Q}\boldsymbol{H}_{\text{in}}^T = \boldsymbol{H}_{\text{in}}^T$. $\boldsymbol{Q}$ is the projection onto the column space of $\boldsymbol{H}_{\text{in}}^T$ and $\boldsymbol{P}$ is the projection onto the nullspace of $\boldsymbol{H}_{\text{in}}$.

8. (MATLAB) Zero-padding removes the first column $c$ of $\boldsymbol{H}$ in the $-1, 3, 3, 1$ matrix in (8.44). Find the inverse of the resulting matrix.

9. (MATLAB) Linear extrapolation adds $2c$ and $-c$ to the next columns of $\boldsymbol{H}_L$, when the first column $c$ is removed. Find the upper left corner of the resulting $\boldsymbol{H}_L^{-1}$. What are the boundary dilation equations in analysis and synthesis?