Chapter 2

Wavelet Algorithms

2.1 Numerical Evaluation of ϕ and ψ

In general, there are no explicit formulas for the basic functions ϕ and ψ. Hence, most algorithms concerning scaling functions and wavelets are formulated in terms of the filter coefficients. We compute the function values of ϕ and ψ. The task of computing the function values of ϕ and ψ is a good example. These algorithms are very much useful when one wants to make plots of scaling functions and wavelets or of linear combinations of such functions.

2.1.1 Computing ϕ at Integers

The scaling function ϕ has support on the interval $[0, D-1]$, with $\phi(0) = 0$ and $\phi(D-1) = 0$ because it is continuous for $D \geq 4$ (see [Dau92]). We discard $\phi(D-1)$ in our computations, but keep $\phi(0)$. Putting $x = 0, 1, \cdots, D-2$ in the dilation equation (1.31), we obtain a homogeneous linear system of equations. For example, when $D = 6$, we obtain the following system of equations:

$$\begin{pmatrix} \phi(0) \\ \phi(1) \\ \phi(2) \\ \phi(3) \\ \phi(4) \end{pmatrix} = \sqrt{2} \begin{pmatrix} a_0 & & & & \\ a_2 & a_1 & a_0 & & \\ a_4 & a_3 & a_2 & a_1 & a_0 \\ & a_5 & a_4 & a_3 & a_2 \\ & & & a_5 & a_4 \end{pmatrix} \begin{pmatrix} \phi(0) \\ \phi(1) \\ \phi(2) \\ \phi(3) \\ \phi(4) \end{pmatrix} = \mathbf{A_0}\mathbf{\Phi}(0), \tag{2.1}$$

where we have defined the vector valued function

$$\mathbf{\Phi}(x) = [\phi(x), \phi(x+1), ..., \phi(x+D-2)]^T.$$

Consider the eigenvalue problem for $\mathbf{A}_0$ as

$$\mathbf{A}_0\mathbf{\Phi}(0) = \lambda\mathbf{\Phi}(0) \tag{2.2}$$

Equation (2.1) has a solution if $\lambda = 1$ is among the eigenvalues of $\mathbf{A}_0$ and hence, the computational problem is entitled to find the eigen solutions of (2.2). The present case (2.1) does, indeed, have an eigensolution corresponding to an eigenvalue.

2.1.2 Computing ϕ at Dyadic Rational

Given $\mathbf{\Phi}(0)$ from (2.1), we can use dilation equation (1.31) again to obtain ϕ at all midpoints between integers in the interval, namely the vector $\mathbf{\Phi}(\frac{1}{2})$. Substituting

$$x = \frac{1}{2}, \frac{3}{2}, \frac{5}{2}, \ldots$$

into the dilation equation (1.31) yields another matrix equation of the form (shown for $D = 1$):

$$\mathbf{\Phi}\left(\frac{1}{2}\right) = \begin{pmatrix} \phi(\frac{1}{2}) \\ \phi(\frac{3}{2}) \\ \phi(\frac{5}{2}) \\ \phi(\frac{7}{2}) \\ \phi(\frac{9}{2}) \end{pmatrix} = \sqrt{2} \begin{pmatrix} a_1 & a_0 & & & \\ a_3 & a_2 & a_1 & a_0 & \\ a_5 & a_4 & a_3 & a_2 & a_1 \\ & & a_5 & a_4 & a_3 \\ & & & & a_5 \end{pmatrix} \begin{pmatrix} \phi(0) \\ \phi(1) \\ \phi(2) \\ \phi(3) \\ \phi(4) \end{pmatrix} = \mathbf{A_1}\mathbf{\Phi}(0). \tag{2.3}$$

Equation (2.3) is an explicit formula and hence,

$$\mathbf{\Phi}(0) = \mathbf{A_0}\mathbf{\Phi}(0),$$

$$\mathbf{\Phi}\left(\frac{1}{2}\right) = \mathbf{A_1}\mathbf{\Phi}(0).$$

This pattern continuous to integers of the form $\frac{k}{4}$, where k is odd, and we get the following system

$$\mathbf{\Phi}\left(\frac{1}{2}\right) = \begin{pmatrix} * & \phi(\frac{1}{4}) \\ \circ & \phi(\frac{3}{4}) \\ * & \phi(\frac{5}{4}) \\ \circ & \phi(\frac{7}{4}) \\ * & \phi(\frac{9}{4}) \\ \circ & \phi(\frac{11}{4}) \\ * & \phi(\frac{13}{4}) \\ \circ & \phi(\frac{15}{4}) \\ * & \phi(\frac{17}{4}) \\ \circ & \phi(\frac{19}{4}) \end{pmatrix} = \sqrt{2} \begin{pmatrix} a_0 & & & & \\ a_1 & a_0 & & & \\ a_2 & a_1 & a_0 & & \\ a_3 & a_2 & a_1 & a_0 & \\ a_4 & a_3 & a_2 & a_1 & a_0 \\ a_5 & a_4 & a_3 & a_2 & a_1 \\ & a_5 & a_4 & a_3 & a_2 \\ & & a_5 & a_4 & a_3 \\ & & & a_5 & a_4 \\ & & & & a_5 \end{pmatrix} \begin{pmatrix} \phi(\frac{1}{2}) \\ \phi(\frac{3}{2}) \\ \phi(\frac{5}{2}) \\ \phi(\frac{7}{2}) \\ \phi(\frac{9}{2}) \end{pmatrix}. \tag{2.4}$$

We observe that if we split it in two systems, one with the equations marked with $*$ and one marked with $\circ$, we can reuse the matrices $\mathbf{A_0}$ and $\mathbf{A_1}$. This pattern repeats itself as follows:

$$\begin{aligned}
\mathbf{\Phi}\left(\frac{1}{4}\right) &= \mathbf{A_0}\mathbf{\Phi}\left(\frac{1}{2}\right), \\
\mathbf{\Phi}\left(\frac{3}{4}\right) &= \mathbf{A_1}\mathbf{\Phi}\left(\frac{1}{2}\right), \\
\mathbf{\Phi}\left(\frac{1}{8}\right) &= \mathbf{A_0}\mathbf{\Phi}\left(\frac{1}{4}\right), \\
\mathbf{\Phi}\left(\frac{5}{8}\right) &= \mathbf{A_1}\mathbf{\Phi}\left(\frac{1}{4}\right), \\
\mathbf{\Phi}\left(\frac{3}{8}\right) &= \mathbf{A_0}\mathbf{\Phi}\left(\frac{3}{4}\right), \\
\mathbf{\Phi}\left(\frac{7}{8}\right) &= \mathbf{A_1}\mathbf{\Phi}\left(\frac{3}{4}\right), \\
\mathbf{\Phi}\left(\frac{1}{16}\right) &= \mathbf{A_0}\mathbf{\Phi}\left(\frac{1}{8}\right), \\
\mathbf{\Phi}\left(\frac{9}{16}\right) &= \mathbf{A_1}\mathbf{\Phi}\left(\frac{1}{8}\right), \\
\mathbf{\Phi}\left(\frac{3}{16}\right) &= \mathbf{A_0}\mathbf{\Phi}\left(\frac{3}{8}\right), \\
\mathbf{\Phi}\left(\frac{11}{16}\right) &= \mathbf{A_1}\mathbf{\Phi}\left(\frac{3}{8}\right), \\
\mathbf{\Phi}\left(\frac{5}{16}\right) &= \mathbf{A_0}\mathbf{\Phi}\left(\frac{5}{8}\right), \\
\mathbf{\Phi}\left(\frac{13}{16}\right) &= \mathbf{A_1}\mathbf{\Phi}\left(\frac{5}{8}\right), \\
\mathbf{\Phi}\left(\frac{7}{16}\right) &= \mathbf{A_0}\mathbf{\Phi}\left(\frac{7}{8}\right), \\
\mathbf{\Phi}\left(\frac{15}{16}\right) &= \mathbf{A_1}\mathbf{\Phi}\left(\frac{7}{8}\right).
\end{aligned}$$

This is the reason why we keep $\mathbf{\Phi}(0)$ in the initial eigenvalue problem (2.1). We can use the same two matrices for all steps in the algorithm and we can continue as follows until a desired resolution 2^q is obtained. So, for $j = 2, 3, \cdots, q$, and for $k = 1, 3, 5, \cdots, 2^{j-1} - 1$, we have

$$\begin{aligned} \mathbf{\Phi}\left(\frac{k}{2^j}\right) &= \mathbf{A}_0 \mathbf{\Phi}\left(\frac{k}{2^{j-1}}\right), \\ \mathbf{\Phi}\left(\frac{k}{2^j} + \frac{1}{2}\right) &= \mathbf{A}_1 \mathbf{\Phi}\left(\frac{k}{2^{j-1}}\right). \end{aligned}$$

2.1.3 Function Values of the Basic Wavelet

Function values of ψ follows from the computed values of ϕ by the wavelet equation (1.32). However, the function values of ϕ are not needed at even numerators:

$$\psi\left(\frac{m}{2^q}\right) = \sqrt{2} \sum_{k=0}^{D-1} b_k \phi\left(\frac{2m}{2^q} - k\right).$$

2.2 Evaluation of Scaling Function Expansion

2.2.1 Non Periodic Case

Let ϕ be the scaling function of genus D and assume that ϕ is known at the dyadic rationales $\frac{m}{2^q}, m = 0, 1, \cdots, (D-1)2^q$, for some chosen $q \epsilon \mathbf{N}$. We want to compute the function

$$f(x) = \sum_{l=-\infty}^{\infty} c_{j,l} \phi_{j,l}(x), \tag{2.5}$$

at the grid points

$$x = x_k = \frac{k}{2^r}, \quad k \in \mathbf{Z}, \tag{2.6}$$

where $r \in \mathbf{N}$ corresponds to some chosen (dyadic) resolution of the real line. Using (1.16), we find that

$$\begin{aligned} \phi_{j,l}\left(\frac{k}{2^r}\right) &= 2^{j/2} \phi(2^j (k/2^r) - l) \\ &= 2^{j/2} \phi(2^{j-r} k - l) \\ &= 2^{j/2} \phi((2^{j+q-r} k - 2^q l)/2^q) \\ &= 2^{j/2} \phi(m(k,l)/2^q), \end{aligned} \tag{2.7}$$

where

$$m(k,l) = k2^{j+q-r} - l2^q. \tag{2.8}$$

Hence, $m(k,l)$ serves as an index into the vector of pre-computed values of ϕ. For this to make sense, $m(k,l)$ must be an integer, which leads to the following restriction:

$$j + q - r \geq 0. \tag{2.9}$$

Only $D-1$ terms of (2.5) can be non zero for any given x_k. From (2.7), we see that these terms are determined by the condition

$$0 < \frac{m(k,l)}{2^q} < D - 1.$$

Hence, the relevant values of l are $l = l_0(k), l_0(k)+1, ..., l_0(k)+D-2$, where

$$l_0(k) = \lceil k2^{j-r} \rceil - D + 1.$$

The sum (2.5), for x given by (2.6), can therefore be written as

$$f\left(\frac{k}{2^r}\right) = 2^{\frac{j}{2}} \sum_{l=l_0(k)}^{l_0(k)+D-2} c_{j,l}\phi\left(\frac{m(k,l)}{2^q}\right), \quad k \in \mathbf{Z}.$$

2.2.2 Periodic Case

We want to compute the function

$$f(x) = \sum_{l=0}^{2^j-1} c_{j,l}\tilde{\phi}_{j,l}(x), \quad x \in [0,1] \tag{2.10}$$

for $x = x_k = k/2^r$, $k = 0, 1, 2, \cdots, 2^r - 1$ where $r \in \mathbf{N}$. Hence, we have

$$\begin{aligned} f\left(\frac{k}{2^r}\right) &= \sum_{l=0}^{2^j-1} c_{j,l}\tilde{\phi}_{j,l}\left(\frac{k}{2^r}\right) \\ &= \sum_{l=0}^{2^j-1} c_{j,l} \sum_{n\in Z} \tilde{\phi}_{j,l}\left(\frac{k}{2^r} + n\right) \\ &= 2^{j/2} \sum_{l=0}^{2^j-1} c_{j,l} \sum_{n\in Z} \phi\left(\frac{m(k,l) + 2^{j+q}n}{2^q}\right) \end{aligned}$$

with $m(k,l) = k2^{j+q-r} - l2^q$ by the same manipulation as in (2.7). Now, assuming that $j \geq J_0$ where J_0 is a non-negative integer, we have $2^j \geq D - 1$. Using Lemma 2.2.1 (proved below), we

obtain the expression

$$f\left(\frac{k}{2^r}\right) = 2^{j/2}\sum_{l=0}^{2^j-1} c_{j,l}\phi\left(\frac{\langle m(k,l)\rangle_{2^{j+q}}}{2^q}\right), \quad k = 0, 1, 2, \cdots, 2^r - 1. \tag{2.11}$$

Lemma 2.2.1 *Let ϕ be a scaling function with support $[0, D-1]$ and let $m, n, j, q \in \mathbf{Z}$ with $q \geq 0$ and $2^j \geq D - 1$. Then*

$$\sum_{n=-\infty}^{\infty} \phi\left(\frac{m(k,l) + 2^{j+q}n}{2^q}\right) = \phi\left(\frac{\langle m\rangle_{2^{j+q}}}{2^q}\right).$$

Proof: See [Nie98].

2.3 DST and IDST - Matrix Formulation

Equation (2.11) is a linear mapping from 2^j scaling function coefficients to 2^r samples of f, so it has a matrix formulation. Let $c_j = [c_{j,0}, c_{j,1}, \cdots, c_{j,2^j-1}]^T$ and $f_r = [f(0), f(1/2^r), \cdots, f((2^r - 1)/2^r)]^T$. We denote the mapping

$$\mathbf{f}_r = \mathbf{T}_{r,j}\mathbf{c}_j. \tag{2.12}$$

When $r = j$, (2.12) becomes

$$\mathbf{f}_j = \mathbf{T}_{j,j}\mathbf{c}_j, \tag{2.13}$$

where $\mathbf{T}_{j,j}$ is a square matrix of order $N = 2^j$. In the case of (2.12), we will often drop the subscripts and write simply

$$\mathbf{f} = \mathbf{T}\mathbf{c}. \tag{2.14}$$

This has the form (shown here for $j = 3, D = 4$)

$$\begin{pmatrix} f(0) \\ f\left(\frac{1}{8}\right) \\ f\left(\frac{2}{8}\right) \\ f\left(\frac{3}{8}\right) \\ f\left(\frac{4}{8}\right) \\ f\left(\frac{5}{8}\right) \\ f\left(\frac{6}{8}\right) \\ f\left(\frac{7}{8}\right) \end{pmatrix} = \begin{pmatrix} \phi(0) & & & & & & \phi(2) & \phi(1) \\ \phi(1) & \phi(0) & & & & & & \phi(2) \\ \phi(2) & \phi(1) & \phi(0) & & & & & \\ & \phi(2) & \phi(1) & \phi(0) & & & & \\ & & \phi(2) & \phi(1) & \phi(0) & & & \\ & & & \phi(2) & \phi(1) & \phi(0) & & \\ & & & & \phi(2) & \phi(1) & \phi(0) & \\ & & & & & \phi(2) & \phi(1) & \phi(0) \end{pmatrix} \begin{pmatrix} c_{3,0} \\ c_{3,1} \\ c_{3,2} \\ c_{3,3} \\ c_{3,4} \\ c_{3,5} \\ c_{3,6} \\ c_{3,7} \end{pmatrix}.$$

The matrix $\mathbf{T}$ is non-singular and we can write

$$\mathbf{c} = \mathbf{T}^{-1}\mathbf{f} \tag{2.15}$$

We denote (2.15) as the **discrete scaling function transform (DST)** and (2.14) as the **inverse discrete scaling function transform (IDST)**.

We now consider the computational problem of interpolating a function $f \in C([0,1])$ between samples at resolution r. That is, we want to use function values $f(k/2^r)$, $k = 0, 1, 2, \cdots, 2^r - 1$ to compute approximations to $f(k/2^{r'})$, $k = 0, 1, 2, \cdots, 2^{r'} - 1$, for some $r' > r$. There are two steps. The first is to solve the system

$$\mathbf{T}_{r,r}\mathbf{c}_r = \mathbf{f}_r$$

for c_r. Then second is to compute the vector $\mathbf{f}_{r'}$ defined by

$$\mathbf{f}_{r'} = \mathbf{T}_{r',r}\mathbf{c}_r. \tag{2.16}$$

Equation (2.16) is illustrated below for the case $r = 3$, and $r' = 4$.

$$\begin{pmatrix} f(0) \\ f\left(\frac{1}{16}\right) \\ f\left(\frac{2}{16}\right) \\ f\left(\frac{3}{16}\right) \\ f\left(\frac{4}{16}\right) \\ f\left(\frac{5}{16}\right) \\ f\left(\frac{6}{16}\right) \\ f\left(\frac{7}{16}\right) \\ f\left(\frac{8}{16}\right) \\ f\left(\frac{9}{16}\right) \\ f\left(\frac{10}{16}\right) \\ f\left(\frac{11}{16}\right) \\ f\left(\frac{12}{16}\right) \\ f\left(\frac{13}{16}\right) \\ f\left(\frac{14}{16}\right) \\ f\left(\frac{15}{16}\right) \end{pmatrix} = 2^{3/2} \begin{pmatrix} \phi(0) & & & & & & \phi(2) & \phi(1) \\ \phi\left(\frac{1}{2}\right) & & & & & & \phi\left(\frac{5}{2}\right) & \phi\left(\frac{3}{2}\right) \\ \phi(1) & \phi(0) & & & & & & \phi(2) \\ \phi\left(\frac{3}{2}\right) & \phi\left(\frac{1}{2}\right) & & & & & & \phi\left(\frac{5}{2}\right) \\ \phi(2) & \phi(1) & \phi(0) & & & & & \\ \phi\left(\frac{5}{2}\right) & \phi\left(\frac{3}{2}\right) & \phi\left(\frac{1}{2}\right) & & & & & \\ & \phi(2) & \phi(1) & \phi(0) & & & & \\ & \phi\left(\frac{5}{2}\right) & \phi\left(\frac{3}{2}\right) & \phi\left(\frac{1}{2}\right) & & & & \\ & & \phi(2) & \phi(1) & \phi(0) & & & \\ & & \phi\left(\frac{5}{2}\right) & \phi\left(\frac{3}{2}\right) & \phi\left(\frac{1}{2}\right) & & & \\ & & & \phi(2) & \phi(1) & \phi(0) & & \\ & & & \phi\left(\frac{5}{2}\right) & \phi\left(\frac{3}{2}\right) & \phi\left(\frac{1}{2}\right) & & \\ & & & & \phi(2) & \phi(1) & \phi(0) & \\ & & & & \phi\left(\frac{5}{2}\right) & \phi\left(\frac{3}{2}\right) & \phi\left(\frac{1}{2}\right) & \end{pmatrix} \begin{pmatrix} c_{3,0} \\ c_{3,1} \\ c_{3,2} \\ c_{3,3} \\ c_{3,4} \\ c_{3,5} \\ c_{3,6} \\ c_{3,7} \end{pmatrix}.$$

2.4 Periodic Function in Interval (a, b)

Consider the problem of expressing a periodic function f defined in interval $]a, b[$, where $a, b \in \mathbf{R}$ instead of unit interval. This can be accomplished by mapping the interval $[a, b]$ linearly to unit interval and we use the machinery derived in the previous Section 2.3.

We impose the resolution 2^r on the unit interval $[a, b]$, i.e.

$$x_k = k\frac{b-a}{2^r} + a, \quad k = 0, 1, 2, ..., 2^r - 1.$$

The linear mapping of the interval $a \leq x \leq b$ to the interval $0 \leq y \leq 1$ is given by

$$y = \frac{x-a}{b-a}, \quad a \leq x < b,$$

hence $y_k = k/2^r, k = 0, 1, 2, ..., 2^r - 1$. Let

$$g(y) = f(x) = f\left((b-a)y + a\right); 0 \leq y < 1.$$

Thus, we have from (2.11),

$$g(y_k) = g\left(\frac{k}{2^r}\right) = 2^{\frac{j}{2}} \sum_{l=0}^{2^j-1} c_{j,l}\phi\left(\frac{\langle m(k,l)\rangle_{2^{j+q}}}{2^q}\right)$$

and transforming back to the interval $[a, b]$ yields $f(x_k) = g(y_k)$. Thus, we have effectively obtained an expression of $f \in [a, b]$ in terms of scaling functions *stretched* to fit this interval at its dyadic subdivisions.

2.5 Fast Wavelet Transforms

The orthogonality of scaling functions and wavelets together with the dyadic coupling between MRA spaces lead to a relation between scaling function coefficient and the wavelet coefficients on different scales. This yields a fast and accurate algorithm due to Mallat [Mal89a] and is denoted by the **pyramid algorithm** or the **fast wavelet transform (FWT)**. Let $f \in L^2(\mathbf{R})$ and consider the projection

$$(P_{V_j} f)(x) = \sum_{l=-\infty}^{\infty} c_{j,l}\phi_{j,l}(x), \tag{2.17}$$

which is given in terms of scaling function only. We know from the Definition 1.4.2 that $P_{V_j} f = P_{V_{j-1}} f + P_{W_{j-1}} f$, so the projection also has a formulation in terms of scaling functions and wavelets:

$$(P_{V_j} f)(x) = \sum_{l=-\infty}^{\infty} c_{j-1,l}(x) + \sum_{l=-\infty}^{\infty} d_{j-1,l}\psi_{j-1,l}(x). \tag{2.18}$$

Our goal here is to derive a mapping between the sequence $\{c_{j,l}\}_{l\in\mathbf{Z}}$ and the sequences $\{c_{j-1,l}\}_{l\in\mathbf{Z}}$, $\{d_{j-1,l}\}_{l\in\mathbf{Z}}$. The key to the derivations is the dilation equation (1.31) and the wavelet equation

(1.32). Using (1.31), we derive the identity

$$\begin{aligned}\phi_{j-1,l}(x) &= 2^{(j-1)/2}\phi(2^{j-1}x-1)\\ &= 2^{j/2}\sum_{k=0}^{D-1}a_k\phi(2^jx-2l-k)\\ &= \sum_{k=0}^{D-1}a_k\phi_{j,2l+k}(x),\end{aligned} \tag{2.19}$$

and from (1.32), we have

$$\psi_{j-1,l}(x)=\sum_{k=0}^{D-1}b_k\phi_{j,2l+k}(x). \tag{2.20}$$

Substituting the first identity into the definition of $c_{j,l}$ from (1.25), we obtain

$$\begin{aligned}c_{j-1,l} &= \int_{-\infty}^{\infty}f(x)\sum_{k=0}^{D-1}a_k\phi_{j,2l+k}(x)dx\\ &= \sum_{k=0}^{D-1}a_k\int_{-\infty}^{\infty}f(x)\phi_{j,2l+k}(x)dx\\ &= \sum_{k=0}^{D-1}a_kc_{j,2l+k}.\end{aligned}$$

Similarly, using the relation (1.28), we can find the similar expression for $d_{j-1,l}$. Hence, we obtain the following two relations:

$$c_{j-1,l}=\sum_{k=0}^{D-1}a_kc_{j,2l+k}, \tag{2.21}$$

$$d_{j-1,l}=\sum_{k=0}^{D-1}b_kc_{j,2l+k}, \tag{2.22}$$

which define a linear mapping from the coefficients in (2.17) to the coefficients in (2.18). We refer to this as the **partial wavelet transform (PWT)**. To decompose the space V_j further, one applies the mapping to the sequence $\{c_{j-1,l}\}_{l\in\mathbf{Z}}$ to obtained the new sequences $\{c_{j-2,l}\}_{l\in\mathbf{Z}}$ and $\{d_{j-2,l}\}_{l\in\mathbf{Z}}$. This pattern can further be repeated to give the full FWT: Applying (2.21) and (2.22) recursively for $j=J,J-1,J-2,...,J_0+1$, starting with the initial sequence $\{c_{j,l}\}_{l\in\mathbf{Z}}$, will then produce the wavelet coefficients in the expansion given in (1.26). Note that once the elements $d_{j-1,l}$ have been computed, they are not modified in subsequent steps. This means that the FWT is very efficient in terms of computational work.

The inverse mapping can be derived in a similar fashion. Equating (2.17) with (2.18) and using (2.20) and (2.21) again, we obtain

$$\begin{aligned}\sum_{l=-\infty}^{\infty} c_{j,l}\phi_{j,l}(x) &= \sum_{n=-\infty}^{\infty} c_{j-1,n}\phi_{j-1,n}(x) + \sum_{n=-\infty}^{\infty} d_{j-1,n}\psi_{j-1,n}(x) \\ &= \sum_{n=-\infty}^{\infty} c_{j-1,n} \sum_{k=0}^{D-1} a_k \phi_{j,2n+k}(x) + \sum_{n=-\infty}^{\infty} d_{j-1,n} \sum_{k=0}^{D-1} b_k \phi_{j,2n+k}(x) \\ &= \sum_{k=0}^{D-1} \sum_{n=-\infty}^{\infty} [c_{j-1,n}a_k + d_{j-1,n}b_k]\phi_{j,2n+k}(x).\end{aligned}$$

We now introduce the variable $l = 2n + k$ in the last expression. Since $k = l - 2n$ and $k \in [0, D-1]$, we find for given l, the following bounds on n:

$$\left\lceil \frac{l-d+1}{2} \right\rceil \equiv n_1(l) \leq n \leq n_2(l) \equiv \left\lfloor \frac{l}{2} \right\rfloor . \tag{2.23}$$

Hence,

$$\sum_{l=-\infty}^{\infty} c_{j,l}\phi_{j,l}(x) = \sum_{l=-\infty}^{\infty} \sum_{n=n_1(l)}^{n_2(l)} [c_{j-1,n}a_{l-2n} + d_{j-1,n}b_{l-2n}]\phi_{j,l}(x),$$

and equating coefficients, we get the reconstruction formula

$$c_{j,l} = \sum_{n=n_1(l)}^{n_2(l)} [c_{j-1,n}a_{l-2n} + d_{j-1,n}b_{l-2n}] . \tag{2.24}$$

We call this the **inverse partial wavelet transform (IPWT)**. Consequently, the **inverse fast wavelet transform (IFWT)** is obtained by repeated application of the reconstruction formula (2.24), for $j = J_0 + 1, J_0 + 2, \cdots, J$.

2.6 Periodic FWT

If the underlying function f is periodic, we also have periodicity in the scaling function and wavelet coefficients: $c_{j,l} = c_{j,l+2^j p}$ (given in (1.72)) and $d_{j,l} = d_{j,l+2^j p}$ (given in (1.73)). Hence, it is enough to consider 2^j coefficients of either type at level j. The periodic PWT is thus given by

$$c_{j-1,l} = \sum_{k=0}^{D-1} a_k c_{j,<2l+k>_{2^j}} \tag{2.25}$$

2.7 Matrix Representation of the FWT

Let $\mathbf{c}_j = [c_{j,0}, c_{j,1}, \cdots, c_{j,2^j-1}]^T$ and similarly $\mathbf{d}_j = [d_{j,0}, d_{j,1}, \cdots, d_{j,2^j-1}]^T$. Since (2.25) and (2.26) are linear mappings from $\mathbf{R}^{2^j}$ onto $\mathbf{R}^{2^{j-1}}$, the PWT can be represented as

$$\mathbf{c}_{j-1} = \mathbf{A}_j \mathbf{c}_j,$$

$$\mathbf{d}_{j-1} = \mathbf{B}_j \mathbf{c}_j$$

where $\mathbf{A}_j$ and $\mathbf{B}_j$ are $2^{j-1} \times 2^j$ matrices containing the filter coefficients. For $D = 6$ and $J = 4$, we have

$$\mathbf{A}_4 = \begin{pmatrix}
a_0 & a_1 & a_2 & a_3 & a_4 & a_5 & & & & & & & & & & \\
 & & a_0 & a_1 & a_2 & a_3 & a_4 & a_5 & & & & & & & & \\
 & & & & a_0 & a_1 & a_2 & a_3 & a_4 & a_5 & & & & & & \\
 & & & & & & a_0 & a_1 & a_2 & a_3 & a_4 & a_5 & & & & \\
 & & & & & & & & a_0 & a_1 & a_2 & a_3 & a_4 & a_5 & & \\
 & & & & & & & & & a_0 & a_1 & a_2 & a_3 & a_4 & a_5 & \\
a_4 & a_5 & & & & & & & & & & a_0 & a_1 & a_2 & a_3 & \\
a_2 & a_3 & a_4 & a_5 & & & & & & & & & & a_0 & a_1 &
\end{pmatrix}$$

and

$$\mathbf{B}_4 = \begin{pmatrix}
b_0 & b_1 & b_2 & b_3 & b_4 & b_5 & & & & & & & & & & \\
 & & b_0 & b_1 & b_2 & b_3 & b_4 & b_5 & & & & & & & & \\
 & & & & b_0 & b_1 & b_2 & b_3 & b_4 & b_5 & & & & & & \\
 & & & & & & b_0 & b_1 & b_2 & b_3 & b_4 & b_5 & & & & \\
 & & & & & & & & b_0 & b_1 & b_2 & b_3 & b_4 & b_5 & & \\
 & & & & & & & & & b_0 & b_1 & b_2 & b_3 & b_4 & b_5 & \\
b_4 & b_5 & & & & & & & & & & b_0 & b_1 & b_2 & b_3 & \\
b_2 & b_3 & b_4 & b_5 & & & & & & & & & & b_0 & b_1 &
\end{pmatrix}$$

These are shift-circulant matrices. Moreover, the rows are orthonormal by equations (1.36) and (1.37). $\mathbf{B}_j$ is similar, but with b_k's in place of a_k's; hence the rows of $\mathbf{B}_j$ are also orthonormal. Moreover, the inner product of any row in $\mathbf{A}_j$ with any row in $\mathbf{B}_j$ is zero by Proposition 1.4.1, i.e. $\mathbf{A}_j \mathbf{B}_j^T = 0$. We now compute $\mathbf{A}_j$ and $\mathbf{B}_j$ to obtain $2^j \times 2^j$ matrix

$$\mathbf{Q}_j = \begin{bmatrix} \mathbf{A}_j \\ \mathbf{B}_j \end{bmatrix}$$

which represents the combined mapping

$$\begin{bmatrix} \mathbf{c}_j - 1 \\ \mathbf{d}_j - 1 \end{bmatrix} = \mathbf{Q}_j \mathbf{c}_j. \tag{2.29}$$

Equation (2.29) is the matrix representation of PWT step as defined by equation (2.25) and (2.26). The matrix $\mathbf{Q}_j$ is orthogonal; since

$$\mathbf{Q}_j \mathbf{Q}_j^T = \begin{bmatrix} \mathbf{A}_j \\ \mathbf{B}_j \end{bmatrix} \begin{bmatrix} \mathbf{A}_j^T & \mathbf{B}_j^T \end{bmatrix} = \begin{bmatrix} \mathbf{A}_j \mathbf{A}_j^T & \mathbf{B}_j \mathbf{B}_j^T \\ \mathbf{B}_j \mathbf{A}_j^T & \mathbf{B}_j \mathbf{B}_j^T \end{bmatrix} = \begin{bmatrix} \mathbf{I}_j & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_j \end{bmatrix}$$

Hence, the mapping (2.29) is inverted by

$$\mathbf{c}_j = \mathbf{Q}_j^T \begin{bmatrix} \mathbf{c}_{j-1} \\ \mathbf{d}_{j-1} \end{bmatrix} \tag{2.30}$$

Equation (2.30) is the matrix representation of an IPWT step as defined in (2.27). The FWT from level J to level $J_0 = J - \lambda$ can now be expressed as the matrix-vector product

$$\mathbf{d} = \mathbf{W}^\lambda \mathbf{c} \tag{2.31}$$

where $\mathbf{c} = \mathbf{c}_J$,

$$\mathbf{d} = \begin{pmatrix} \mathbf{c}_{J_0} \\ \mathbf{d}_{J_0} \\ \mathbf{d}_{J_0+1} \\ . \\ . \\ \mathbf{d}_{J-1} \end{pmatrix}$$

and

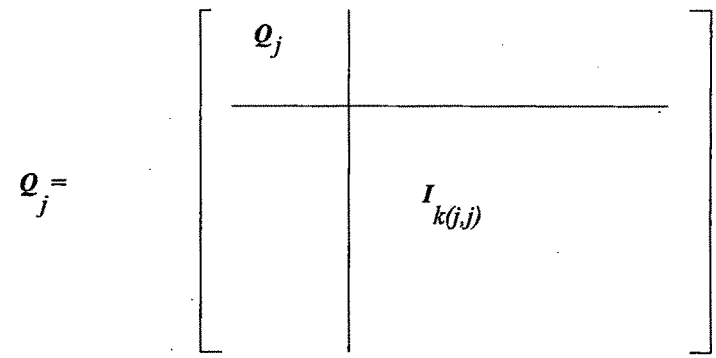
$$\mathbf{W}^\lambda = \hat{\mathbf{Q}}_{J_0} \hat{\mathbf{Q}}_{J_0+1} \ldots . \hat{\mathbf{Q}}_{J-1}.$$

The matrix $\hat{\mathbf{Q}}$ of order 2^j is given by where $\mathbf{I}_{k(j,J)}$ denotes the $k \times k$ identity matrix with $k = k(j, J) = 2^J - 2^j$. It follows that $\mathbf{W}^\lambda$ is orthogonal and hence the IFWT is given by

$$\mathbf{c} = \left(\mathbf{W}^\lambda\right)^T \mathbf{d}.$$

2.8 A More General FWT

Suppose that elements to be transformed may contain other prime factors than 2, i.e. $N = K2^J$ with $J, K \in Z$ and $\langle K \rangle_2 = 1$ (see equation (A.3) in Appendix). In this case, the FWT can still



be defined by mappings of the form (2.25) and (2.26) with a maximal depth $\lambda = J$, i.e. $J_0 = 0$. It will be convenient to use a slightly different notation for the algorithms in next chapters. Hence, let $S_i = \frac{N}{2^i}$ and

$$c_k^i \equiv c_{J-i,k}$$
$$d_k^i \equiv d_{J-i,k}$$

for $i = 0, 1, 2, \cdots, \lambda - 1$ and $k = 0, 1, 2, \cdots, S_i - 1$. We can now define the FWT as follows:

Definition 2.8.1 ***Fast wavelet transform (FWT):*** *Let $J, K \in N$ with $(K)_2 = 1$ and $N = K2^J$. Let $\mathbf{c}^0 = \{c_k^0\}_{k=0}^{N-1}$ be given. The FWT is then computed by the recurrence relations*

$$\begin{array}{rcl} c_n^{i+1} & = & \sum_{l=0}^{D-1} a_l c_{\langle l+2n \rangle_{S_i}}^i \\ d_n^{i+1} & = & \sum_{l=0}^{D-1} b_l c_{\langle l+2n \rangle_{S_i}}^i \end{array} \tag{2.32}$$

where $i = 0, 1, \cdots, \lambda - 1$, $S_i = \frac{N}{2^i}$, and $n = 0, 1, 2, \cdots, S_{i+1} - 1$.

The relation between the scaling function coefficients and the underlying function is unimportant for the FWT algorithm. In numerical applications we often need to apply the FWT to vectors of function values or even arbitrary vectors. Hence let $\mathbf{x} \in \mathbf{R}^n$. Then the expression

$$\breve{\mathbf{x}} = \mathbf{W}^\lambda \mathbf{x} \tag{2.33}$$

is defined by putting $\mathbf{c}^0 = \mathbf{x}$, performing λ steps of (2.32), and letting

$$\breve{\mathbf{x}} = \left\{ \{c_k^\lambda\}_{k=0}^{S_\lambda - 1}, \left\{ \{d_k^i\}_{k=0}^{S_i - 1} \right\}_{i=\lambda, \lambda-1, \cdots, 1} \right\}$$

If λ is omitted in (2.33), it will assume its maximal value J. Note that $\mathbf{W}^\lambda = \mathbf{I}$ for $\lambda = 0$. The IFWT is written similarly as

$$\mathbf{x} = \left(\mathbf{W}^\lambda\right)^T \breve{\mathbf{x}} \tag{2.34}$$

corresponding to the recurrence formula

$$c_l^i = \sum_{n=n_1(l)}^{n_2(l)} a_{l-2n} c_{\langle n \rangle S_{i+1}}^{i+1} + b_{l-2n} d_{\langle n \rangle S_{i+1}}^{i+1}$$

where $i = \lambda - 1, \lambda - 2, \cdots, 1, 0$, $S_i = \frac{N}{2^i}$, $l = 0, 1, \cdots, S_i - 1$, and $n_1(l)$ and $n_2(l)$ are defined as in (2.23).

2.9 Complexity of the FWT Algorithm

One step of (2.32), the PWT, involves DS_i additions and DS_i multiplications. The number of floating point operations is therefore

$$F_{PWT}(S_i) = 2DS_i \tag{2.35}$$

Let N be the total number of elements to be transformed and assume that the FWT is carried out to depth λ. The FWT consists of λ applications of the PWT to successively shorter vectors so that the total work is

$$\begin{aligned} F_{PWT}(N) &= \sum_{i=0}^{\lambda-1} F_{PWT}\left(\frac{N}{2^i}\right) \\ &= \sum_{i=0}^{\lambda-1} 2D\left(\frac{N}{2^i}\right) \\ &= 2DN\frac{1-\frac{1}{2^\lambda}}{1-\frac{1}{2}} \\ &= 4DN\left(1-\frac{1}{2^\lambda}\right) < 4DN \end{aligned} \tag{2.36}$$

D is normally constant throughout wavelet analysis, so the complexity is $O(N)$. The IFWT has the same complexity as the FWT. For comparison, we mention that the complexity of the fast Fourier transform (FFT) is $O(N \log_2 N)$.

2.10 Two Dimensional Fast Wavelet Transform

Using the definition of $\mathbf{W}^\lambda$ from the previous section, we define the $2D$ FWT as

$$\breve{\mathbf{X}} = \mathbf{W}^{\lambda_M} \mathbf{X} \left(\mathbf{W}^{\lambda_N}\right)^T \tag{2.37}$$

where $\mathbf{X}, \breve{\mathbf{X}} \in \mathbf{R}^{M,N}$. The parameters λ_M and λ_N are the transform depths in the first and second dimensions, respectively. Equation (2.37) can be expressed as M $1D$ wavelet transforms of the rows of $\mathbf{X}$ followed by N $1D$ wavelet transforms of the columns of $\mathbf{X}(\mathbf{W}^{\lambda_N})^T$, i.e.

$$\breve{\mathbf{X}} = \mathbf{W}^{\lambda_M} \left(\mathbf{W}^{\lambda_N}\mathbf{X}\right)^T.$$

Therefore, it is straightforward to compute the $2D$ FWT given in the $1D$ FWT from Definition-2.8.1. It follows that the inverse $2D$ FWT is defined as

$$\breve{\mathbf{X}} = \left(\mathbf{W}^{\lambda_M}\right)^T \breve{\mathbf{X}}\mathbf{W}^{\lambda_N} \tag{2.38}$$

Using (2.36), we find that the computational work of the $2D$ FWT as follows:

$$\begin{aligned} F_{FWT2}(M,N) &= MF_{FWT}(N) + NF_{FWT}(M) \\ &= M4DN\left(1 - \frac{1}{2^{\lambda_M}}\right) + N4DM\left(1 - \frac{1}{2^{\lambda_N}}\right) \\ &= 4DMN\left(2 - \frac{1}{2^{\lambda_M}} - \frac{1}{2^{\lambda_N}}\right) < 8DMN \end{aligned} \tag{2.39}$$

The computational work of the inverse $2D$ FWT is the same.

2.11 Accuracy of the Multiresolution Space

2.11.1 Approximation Properties of V_J

In this section we discuss the point wise approximation error introduced when a function f is approximated by an expansion in scaling functions at level J. Let $J \in \mathbf{Z}$, $f \in L^2(\mathbf{R})$ and assume that $f \in C^P(\mathbf{R})$. For an arbitrary, but fixed x, we defined the point wise error as

$$e_J(x) = f(x) - (P_{V_J}f)(x), \quad x \in \mathbf{R}$$

where $(P_{V_J}f)(x)$ is the orthogonal projection of f onto the approximation space V_J as given in Definition 1.4.2. Recall that $P_{V_J}f$ has expansion in terms of the scaling functions as well as in terms of wavelets. The wavelet expansion for $P_{V_J}f$ is

$$(P_{V_j}f)(x) = \sum_{k=-\infty}^{\infty} c_{J_0,k}\phi_{J_0,k}(x) + \sum_{j=J_0}^{J-1}\sum_{k=-\infty}^{\infty} d_{j,k}\psi_{j,k}(x), \tag{2.40}$$

and by letting $J \to \infty$ temporarily we get a wavelet expansion for f itself:

$$f(x) = \sum_{k=-\infty}^{\infty} c_{J_0,k}\phi_{J_0,k}(x) + \sum_{j=J_0}^{\infty}\sum_{k=-\infty}^{\infty} d_{j,k}\psi_{j,k}(x), \tag{2.41}$$

Then subtracting (2.41) from (2.40), we obtain an expression for error e_J in terms of the wavelets at scales $j \geq J$:

$$e_J(x) = \sum_{j=J}^{\infty} \sum_{k=-\infty}^{\infty} d_{j,k}\psi_{j,k}(x). \tag{2.42}$$

Define

$$\begin{aligned} C_\psi &= \max_{x \in I_{j,k}} |\psi(2^j - k)| \\ &= \max_{y \in [0, D-1]} |\psi(y)|. \end{aligned}$$

Hence, $\max_{x \in I_{j,k}} |\psi_{j,k}(x)| = 2^{j/2} C_\psi$ and using Theorem 1.4.2, we find that

$$|d_{j,k}\psi_{j,k}(x)| \leq C_P 2^{-jP} \max_{\xi \in I_{j,k}} |f^{(P)}(\xi)| C_\psi.$$

Recall that

$$supp(\psi_{j,k}) = I_{j,k} = \left[\frac{k}{2^j}, \frac{k + D - 1}{2^j} \right].$$

Hence, there are at most $D - 1$ intervals $I_{j,k}$ containing a given value of x. Thus, for any x only $D - 1$ terms in the inner summation in (2.42) are nonzero. Let I_j be the union of all these intervals. i.e.

$$I_j(x) = \bigcup_{\{l : x \in I_{j,l}\}} I_{j,l}$$

and let

$$\mu_j^P(x) = \max_{\xi \in I_j(x)} |f^P(\xi)|.$$

Then one finds a common bound for all terms in the inner sum as

$$\sum_{k=-\infty}^{\infty} |d_{j,k}\psi_{j,k}| \leq C_\psi C_P 2^{-jP} (D - 1) \mu_j^P(x).$$

The outer sum can now evaluated by using the fact that

$$\mu_J^P \geq \mu_{J+1}^P \geq \mu_{J+2}^P(x) \geq \ldots$$

and we establish the bound

$$\begin{aligned} |e_j(x)| &\leq C_\psi C_P (D - 1) \mu_J^P(x) \sum_{j=J}^{\infty} 2^{-jP} \\ &= C_\psi C_P (D - 1) \mu_J^P(x) \frac{2^{-JP}}{1 - 2^{-P}}. \end{aligned}$$

Thus, we see that for an arbitrary but fixed x the approximation error will be bounded as

$$|e_J(x)| = O(2^{-JP}).$$

This is exponential decay with respect to the resolution J. Furthermore, the greater the number of vanishing moments P, the faster the decay. Finally, note that each error term $d_{j,k}\psi_{j,k}(x)$ is zero for $x \notin I_{j,k}$, and hence, $e_J(x)$ depends only on $f(y), y \in [x, x + (D-1)/2^J]$.

2.11.2 Approximation Properties of $\tilde{V}_J$

We now consider the approximation error in the periodic case. Let $f \in L^2([0,1])$ and assume that its periodic extension (1.66) is P times differentiable everywhere. Furthermore, let $J \geq J_0$ be the smallest integer such that $2^{J_0} \geq D-1$ and define the approximation error as

$$\tilde{e}_J(x) = f(x) - (P_{\tilde{V}_J} f)(x), \quad x \in [0,1]$$

where $(P_{\tilde{V}_J})(x)$ is the orthogonal projection of f onto the approximate space $\tilde{V}_J$ as defined in Definition 1.5.2. Using the periodic wavelet expansion

$$f(x) = \sum_{l=0}^{2^{J_0}-1} c_{J_0,l}\tilde{\phi}_{J_0,l}(x) + \sum_{j=J_0}^{J-1}\sum_{l=0}^{2^j-1} d_{j,l}\tilde{\psi}_{j,l}(x), \quad x \in [0,1]$$

and proceeding as in the non-periodic case, we find that

$$\tilde{e}_J(x) = \sum_{j=J}^{\infty}\sum_{k=0}^{2^j-1} d_{j,k}\tilde{\psi}_{j,k}(x) \tag{2.43}$$

Since the coefficients $d_{j,k}$ are the same as in the non periodic case by by equation (1.28), Theorem 1.4.2 applies and we can repeat the analysis from Subsection 2.11.1 to obtain

$$|\tilde{e}_J(x)| = O(2^{-JP}) \quad x \in [0,1].$$

We now consider the infinity norm of $\tilde{e}_J$ defined by

$$\|\tilde{e}_J(x)\|_\infty = \max_{x\in[0,1]} |\tilde{e}_J(x)|.$$

A similar analysis yields

$$\begin{aligned}
\|\tilde{e}_J\|_\infty &\leq \sum_{j=J}^{\infty}\sum_{k=0}^{2^j-1} |d_{j,k}| \max_{x\in I_{j,k}} |\tilde{\psi}_{j,k}(x)| \\
&\leq C_\psi C_P \sum_{j=J}^{\infty}\sum_{k=0}^{2^j-1} |2^{j/2}2^{-j(P+\frac{1}{2})}| \max_{\xi\in I_{j,k}} |f^P(\xi)| \\
&= C_\psi C_P \max_{\xi\in[0,1]} |f^{(p)}(\xi)| \sum_{j=J}^{\infty} 2^{-jP} \\
&= C_\psi C_P \max_{\xi\in[0,1]} |f^{(p)}(\xi)| \frac{2^{-jP}}{1-2^{-P}}
\end{aligned}$$

Hence

$$\|\tilde{e}_J(x)\|_\infty = O(2^{-JP}).$$

Finally, consider the L^2 norm of $\tilde{e}_J$:

$$\begin{aligned}
\|\tilde{e}_J\|_2^2 &= \int_0^1 \tilde{e}_J^2(x)dx \\
&\leq \|\tilde{e}_J\|_\infty^2 \int_0^1 dx \\
&= \|\tilde{e}_J\|_\infty^2
\end{aligned}$$

Therefore, we obtain

$$\|\tilde{e}_J(x)\|_2 = O(2^{-JP}).$$

2.12 Wavelet Transform of a Circulant Matrix

2.12.1 Introduction

In this section, we will describe an algorithm for computing the $2D$ fast wavelet transform of a circulant $N \times N$ matrix $\mathbf{A}$. The $2D$ FWT is defined in (2.37) and circulant matrices are discussed in [Fra99]. Recall that the $2D$ FWT is a mapping $\mathbf{A} \rightarrow \mathbf{H}$ given as

$$\mathbf{H} = \mathbf{WAW}^T$$

where $\mathbf{W}$ is defined in (2.33). We will show that this can be dine in $O(N)$ steps and that $\mathbf{H}$ can represented using $O(N)$ elements in a suitable structure. Using the structure we define an

efficient algorithm for computing the matrix vector product **Hx** where **x** is an arbitrary vector of length N. This algorithm has complexity $O(N)$ and will be used in Chapters-4 in a wavelet method for solving PDE's. This topic is based on [Nie98] and the approaches follows ideas proposed by Charton [Cha96]. However, many of the details data structures and the complexity analysis are represented by us.

2.12.2 The Wavelet Transform Revisited

Let N and λ be integers of the form given in definition and let $\mathbf{c}^i$ be a given vector with elements $\left[c_0^i, c_1^i, \cdots, c_{N-1}^i\right]$ and let d^i be defined similarly. Recall that $1D$ FWT is defined by the recurrence formulas from definition:

$$c_m^{i+1} = \sum_{k=0}^{D-1} a_k c_{\langle k+2m \rangle_{S_i}}^i$$

$$d_m^{i+1} = \sum_{k=0}^{D-1} b_k c_{\langle k+2m \rangle_{S_i}}^i$$

for $i = 0, 1, \cdots, \lambda - 1$, $m = 0, 1, 2, \cdots, S_{i+1} - 1$ and $S_i = \frac{N}{2^i}$. Figure-2.1 illustrates that how a vector is transformed.

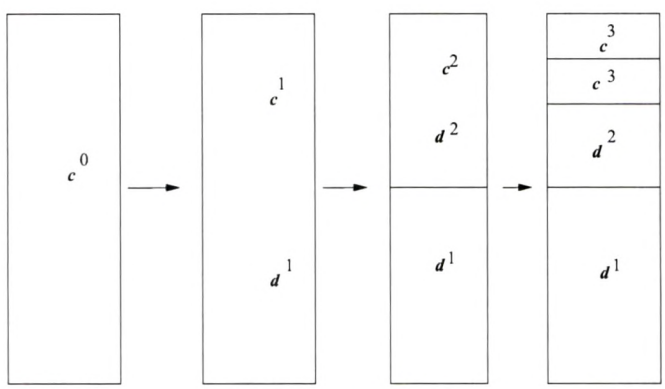


Figure 2.1: **The steps of a $1D$ wavelet transform for $\lambda = 3$ and $J = 4$.**

The $2D$ FWT defined in (2.37) has also a recursive formulation. The $2D$ recurrence formulas are straightforward generalizations of the $1D$ formulas and they decompose a matrix analogously to the way the $1D$ recurrence formulas decompose a vector.

The recursion is initialized by assigning a matrix $A \in \mathbf{R}^{N\times N}$ to the initial block which we denote $\mathbf{CC}^{00}$. This block is then successively split into the smaller blocks denoted $\mathbf{CC}^{i,j}, \mathbf{DC}^{i,j}, \mathbf{CD}^{i,j}, \mathbf{DD}^{i,j}$ for $i, j = 1, 2, \cdots, \lambda$. The block dimensions are determined by the superscripts i, j. A block with indices i, j has $S_i = \frac{N}{2^i}$ rows and $S_i = \frac{N}{2^j}$ columns. The steps of the $2D$ wavelet transform for $\lambda = 3$ are shown Figure-2.2 with $\mathbf{H}$ being the aggregation of all blocks after the final steps shown in the upper right corner of the figure (comparison with the $1D$ wavelet transform shown in Figure 2.1 can be helpful fore understanding the scheme). Note that each step of the transform produces blocks that have attained their final values, namely those of the type $\mathbf{DD}^{i,j}$ and subsequent steps work on blocks of the type $\mathbf{CC}^{i,j}$, $\mathbf{CD}^{i,j}$ and $\mathbf{DC}^{i,j}$. The formulas for three types of blocks are given below.

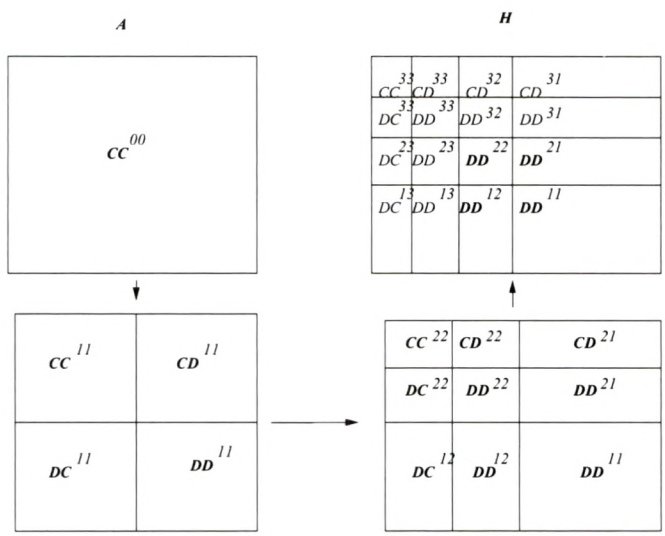


Figure 2.2: **The steps of a $2D$ wavelet transform for $\lambda = 3$ and $J = 4$**

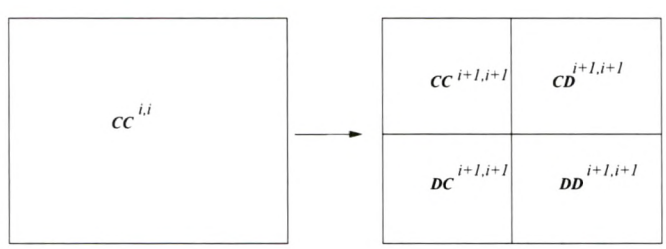


Figure 2.3: **The transform of a square block on the diagonals yields four new square blocks.**

Blocks on the diagonal

Consider the square blocks $\mathbf{CC}^{i,i}$ in Figure-2.3. Each step of the $2D$ wavelet transforms splits such a block into four new blocks denoted $\mathbf{CC}^{i+1,i+1}, \mathbf{CD}^{i+1,i+1}, \mathbf{DC}^{i+1,i+1}$ and $\mathbf{DD}^{i+1,+i+1}$. Figure-2.3 illustrates the decomposition of this type. Let $CC^{i,j}_{m,n} = [\mathbf{CC}^{i,j}]_{m,n}$ and similarly for $CD^{i+1}_{m,n}, DC^{i+1}_{m,n}, DD^{i+1}_{m,n}$. The recurrence formulas for these decomposition are then given as follows:

$$CC^{i+1,i+1}_{m,n} = \sum_{k=0}^{D-1}\sum_{l=0}^{D-1} a_k a_l CC^{i,i}_{<k+2m>_{s_i},<l+2n>_{s_i}} \tag{2.44}$$

$$CD^{i+1,i+1}_{m,n} = \sum_{k=0}^{D-1}\sum_{l=0}^{D-1} a_k b_l CC^{i,i}_{<k+2m>_{s_i},<l+2n>_{s_i}} \tag{2.45}$$

$$DC^{i+1,i+1}_{m,n} = \sum_{k=0}^{D-1}\sum_{l=0}^{D-1} b_k a_l CC^{i,i}_{<k+2m>_{s_i},<l+2n>_{s_i}} \tag{2.46}$$

$$DD^{i+1,i+1}_{m,n} = \sum_{k=0}^{D-1}\sum_{l=0}^{D-1} b_k b_l CC^{i,i}_{<k+2m>_{s_i},<l+2n>_{s_i}} \tag{2.47}$$

for $i = 0, 1, \cdots, \lambda - 1$ and $m, n = 0, 1, 2, \cdots, S_{i+1} - 1$

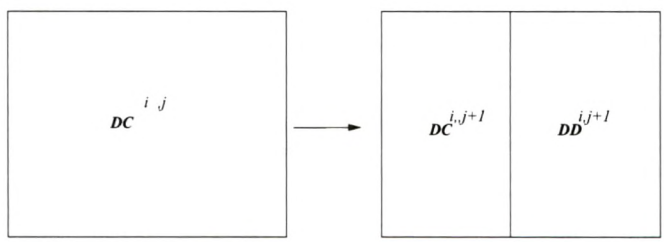


Figure 2.4: **The transform of a block below the diagonals yields two new rectangular blocks.**

Blocks below the diagonal

Blocks of the type $\mathbf{DC}^{i,j} (j \geq i)$ are split in one direction only as indicated in Figure-2.4. The recurrence formulas are the $1D$ formulas applied to each row of the blocks $\mathbf{DC}^{i,j}$:

$$DC^{i,j+1}_{m,n} = \sum_{k=0}^{D-1} a_l DC^{i,j}_{m,\langle l+2n\rangle_{S_j}} \tag{2.48}$$

$$DD_{m,n}^{i,j+1} = \sum_{kl=0}^{D-1} b_l DC_{m,\langle l+2n\rangle_{S_j}}^{i,j} \tag{2.49}$$

for $j = i, i+1, \cdots, \lambda - 1$ and $m = 0, 1, 2, \cdots, S_i - 1$, $n = 0, 1, 2, \cdots, S_{j+1} - 1$.

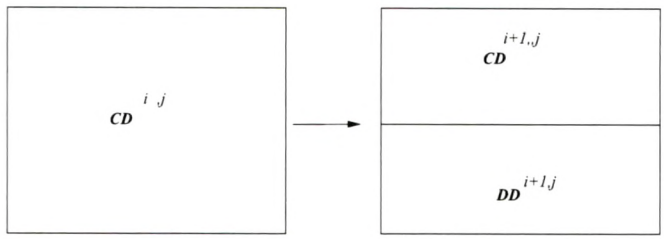


Figure 2.5: **Transform of a block above the diagonals yields two new rectangular blocks.**

Blocks above the diagonal

For blocks $\mathbf{CD}_{i,j}$ with $i \geq j$ we have a splitting as shown in Figure-2.5. The recurrence formulas are the $1D$ formulas applied to each column of $\mathbf{CD}_{i,j}$:

$$CD_{m,n}^{i+1,j} = \sum_{k=0}^{D-1} a_k CD_{\langle k+2n\rangle_{S_i},n}^{i,j} \tag{2.50}$$

$$DD_{m,n}^{i+1,j} = \sum_{l=0}^{D-1} b_k CD_{\langle k+2n\rangle_{S_i},n}^{i,j} \tag{2.51}$$

for $i = j, j+1, \cdots, \lambda - 1$ and $m = 0, 1, 2, \cdots, S_{i+1} - 1$, $n = 0, 1, 2, \cdots, S_j - 1$.

2.13 $2D$ Wavelet Transform of a Circulant Matrix

The $2D$ FWT of circulant matrices give rise to structured matrix blocks which we will call **shift-circulant matrices**. We begin by giving the definition.

Let $\mathbf{A}$ be an $M \times M$ circulant matrix as defined in [Fra99] and let $\{a_m\}_{m=0,1,\cdots,M-1}$ be the first column of $\mathbf{A}$. Then

$$[\mathbf{A}]_{m,n} = a_{\langle m-n\rangle_M}, m, n = 0, 1, \cdots, M-1.$$

A shift circulant matrix is a generalization of a circulant matrix which we define as follows:

Definition 2.13.1 ***Shift-circulant matrix:***

1. *Let* $\mathbf{A}$ *be an* $M \times N$ *matrix where* $M \geq N$ *with* M *divisible by* N*, and let* $\{a_m\}_{m=0,1,\cdots,M-1}$ *be the first column of* $\mathbf{A}$*. Then,* $\mathbf{A}$ *is* ***column-shift-circulant*** *if*

$$[\mathbf{A}]_{m,n} = a_{\langle m-\sigma n\rangle_M}, \quad m = 0, 1, \cdots, M-1, \quad n = 0, 1, ..., N-1$$

   where $\sigma = M/N$

2. *Let* $\mathbf{A}$ *be an* $M \times N$ *matrix where* $N \geq M$ *with* N *divisible by* M*, and let* $\{a_n\}_{n=0,1,\cdots,N-1}$ *be the first row of* $\mathbf{A}$*. Then* $\mathbf{A}$ *is* ***row-shift-circulant*** *if*

$$[\mathbf{A}]_{m,n} = a_{\langle n-\sigma m\rangle_N}, \quad m = 0, 1, \cdots, M-1, \quad n = 0, 1 \cdots, N-1.$$

   where $\sigma = N/M$

The number σ *is a positive integer that denotes the amount by which columns or rows are shifted.*

A column-shift-circulant 4×2 matrix ($\sigma = 2$) has the form

$$\begin{pmatrix} a_0 & a_2 \\ a_1 & a_3 \\ a_2 & a_0 \\ a_3 & a_1 \end{pmatrix}$$

A row-shift-circulant 2×4 matrix ($\sigma = 2$) has the form

$$\begin{pmatrix} a_0 & a_1 & a_2 & a_3 \\ a_2 & a_3 & a_0 & a_1 \end{pmatrix}.$$

Note that a circulant matrix is both column-shift-circulant and row-shift-circulant with $\sigma = 1$. Let $\{a_n\}_{n=0,1,\cdots,N-1}$ be the first column of a circulant matrix $\mathbf{A}$. Using this column vector as a point of a departure, we will now show how to compute a representation of $\mathbf{H}$ using only one vector per block. Note that according to the recurrence equations (2.44) to (2.51), the operations can be divided into $2D$ transforms of blocks on the diagonal and $1D$ row or column transforms of off-diagonal blocks. We will treat these cases separately.

2.13.1 Blocks on the Diagonal

Lemma 2.13.1 *Let* $\mathbf{CC}^{i,i}$ *be a* $S_i \times S_i$ *circulant matrix. Then* $\mathbf{CC}^{i+1,i+1}$, $\mathbf{CD}^{i+1,i+1}$, $\mathbf{DC}^{i+1,i+1}$ *and* $\mathbf{DD}^{i+1,j+1}$ *defined by (2.44) to (2.47) respectively, are circulant matrices.*

Proof: We will prove the lemma for $\mathbf{CD}^{i+1,i+1}$ only since the other cases are completely analogous.

By assumption $\mathbf{CC}^{i,i}$ is circulant, i.e.

$$\mathbf{CC}^{i,i}_{m,n} = cc^{i,i}_{\langle m-n\rangle_{S_i}}$$

where $\mathbf{cc}^{i,i}$ is the first column of $\mathbf{CC}^{i,i}$. Equation (2.45), then becomes

$$CD^{i+1,i+1}_{m,n} = \sum_{k=0}^{D-1}\sum_{l=0}^{D-1} a_k b_l cc^{i,i}_{\langle\langle k+2m\rangle_{S_i}-\langle l+2n\rangle_{S_i}\rangle_{S_i}} \tag{2.52}$$

Considering now the index of the typical term of (2.52) and using Lemma A.1.1 and A.1.2 mentioned in Appendix A, we find that

$$\begin{aligned}\langle\langle k+2m\rangle_{S_i} - \langle l+2n\rangle_{S_i}\rangle_{S_i} &= \langle\langle 2(m-n)\rangle_{S_i} + k - l\rangle_{S_i} \\ &= \langle\langle 2(m-n)\rangle_{S_i/2} + k - l\rangle_{S_i} \\ &= \langle\langle 2(m-n)\rangle_{S_{i+1}} + k - l\rangle_{S_i}\end{aligned}$$

This expression does not depend on the individual values of m and n but only on their *difference*. Therefore, $CD^{i+1,i+1}_{m,n}$ depends only on $\langle m-n\rangle_{S_{i+1}}$ which proves that $\mathbf{CD}^{i+1,i+1}$ is circulant. Hence,

$$CD^{i+1,i+1}_{m,n} = cd^{i+1,i+1}_{\langle m-n\rangle_{S_{i+1}}}$$

for some column vector $\mathbf{cd}^{i+1,i+1}$. Having established that $\mathbf{CD}^{i+1,i+1}$ is circulant, we can now give a formula for the vector $\mathbf{cd}^{i+1,i+1}$. Putting $n = 0$ in equation (2.52) and using Lemma A.1.1, we obtain the first column of $\mathbf{CD}^{i+1,i+1}$;

$$\begin{aligned}cd^{i+1,i+1}_m = CD^{i+1,i+1}_{m,0} &= \sum_{k=0}^{D-1}\sum_{l=0}^{D-1} a_k b_l cc^{i,i}_{\langle\langle 2m+k\rangle_{S_i}-\langle l\rangle_{S_i}\rangle_{S_i}} \\ &= \sum_{k=0}^{D-1}\sum_{l=0}^{D-1} a_k b_l cc^{i,i}_{\langle 2m+k-l\rangle_{S_i}}\end{aligned} \tag{2.53}$$

where $m = 0, 1, \cdots, S_{i+1} - 1$. A computationally more efficient expression for $cd^{i+1,i+1}_m$ can be derived by rearranging the terms in (2.53) so that we sum over the differences in the indices. More precisely, we split the double sum into terms where $k - l \geq 0$ and terms where $k - l < 0$ and rearrange these separately. Thus, let $p = k - l$. The terms in (2.53) with $k - l \geq 0$ are

$$\sum_{k=0}^{D-1}\sum_{l=0}^{k} a_k b_l cc^{i,i}_{\langle 2m+k-l\rangle_{S_i}} = \sum_{k=0}^{D-1}\sum_{l=0}^{k} a_{l+p} b_l cc^{i,i}_{\langle 2m+p\rangle_{S_i}} \tag{2.54}$$

k/l	0	1	2	3	4	5
0	0					
1	1	0				
2	2	1	0			
3	3	2	1	0		
4	4	3	2	1	0	
5	5	4	3	2	1	0

The following table shows p as a function of k and l (for $D = 6$): Summing over the diagonals yields an alternative expression for (2.54):

$$\sum_{k=0}^{D-1}\sum_{l=0}^{k} a_{l+p}b_l cc^{i,i}_{\langle 2m+p\rangle_{S_i}} = \sum_{p=0}^{D-1}\left[\sum_{l=0}^{D-1-p} a_{l+p}b_l\right] cc^{i,i}_{\langle 2m+p\rangle_{S_i}} \tag{2.55}$$

Similarly, we take the terms from (2.53) with $k - l < 0$ and set $p = l - k > 0$:

$$\sum_{k=0}^{D-1}\sum_{l=k+1}^{D-1} a_k b_{k+p} cc^{i,i}_{\langle 2m-p\rangle_{S_i}} = \sum_{p=1}^{D-1}\left[\sum_{k=0}^{D-1-p} a_k b_{k+p}\right] cc^{i,i}_{\langle 2m-p\rangle_{S_i}} \tag{2.56}$$

Now, let

$$q^p_{ab} = \sum_{k=0}^{D-1-p} a_k b_{k+p}, \quad p = 0, 1, \cdots, D-1.$$

Combining (2.55) and (2.56), we can rewrite (2.53) as

$$cd^{i+1,i+1}_m = q^0_{ab} cc^{i,i}_{2m} + \sum_{p=1}^{D-1}\left[q^p_{ab} cc^{i,i}_{\langle 2m-p\rangle_{S_i}} + q^p_{ba} cc^{i,i}_{\langle 2m+p\rangle_{S_i}}\right] \tag{2.57}$$

for $m = 0, 1, \cdots, S_{i+1} - 1$. The vectors $\mathbf{cc}^{i+1,i+1}$, $\mathbf{dc}^{i+1,i+1}$ and $\mathbf{dd}^{i+1,i+1}$ are computed similarly but with the filters q^p_{aa}, q^p_{ba}, q^p_{bb}, respectively. The formulas are

$$cc^{i+1,i+1}_m = q^0_{aa} cc^{i,i}_{2m} + \sum_{p=1}^{D-1}\left[q^p_{aa} cc^{i,i}_{\langle 2m-p\rangle_{S_i}} + q^p_{aa} cc^{i,i}_{\langle 2m+p\rangle_{S_i}}\right] \tag{2.58}$$

$$dc^{i+1,i+1}_m = q^0_{ba} cc^{i,i}_{2m} + \sum_{p=1}^{D-1}\left[q^p_{ba} cc^{i,i}_{\langle 2m-p\rangle_{S_i}} + q^p_{ab} cc^{i,i}_{\langle 2m+p\rangle_{S_i}}\right] \tag{2.59}$$

$$dd^{i+1,i+1}_m = q^0_{bb} cc^{i,i}_{2m} + \sum_{p=1}^{D-1}\left[q^p_{bb} cc^{i,i}_{\langle 2m-p\rangle_{S_i}} + q^p_{bb} cc^{i,i}_{\langle 2m+p\rangle_{S_i}}\right] \tag{2.60}$$

It follows from (1.36) that

$$q_{aa}^{0} = q_{bb}^{0} = 1$$

$$q_{ab}^{p} = q_{ba}^{p} = 0, \quad p \ even$$

$$q_{aa}^{p} = q_{bb}^{p} = 0, \quad p \ even, \quad p > 0$$

So, the computational work in computing (2.57)-(2.60) is reduced accordingly.

2.13.2 Blocks below the Diagonal

We now turn to the task of computing the lower off-diagonal blocks $\mathbf{DC}^{i,j+1}$ and $\mathbf{DD}^{i,j+1}$, $j \geq i$ defined by (2.48) and (2.49), respectively. The operations differ from those of the diagonal cases by being applied in one dimension only. Therefore, blocks tend to become more rectangular which means that they are not necessarily circulant in the ordinary sense. However, as we shall see, the typical block is still represented by a single vector, one which is shifted to match the rectangular shape. The block is then a *column-shift-circulant* matrix in the sense of Definition-2.13.1.

Lemma 2.13.2 *Let* $\mathbf{DC}^{i,j}$, $j \geq i$, *be a* $S_i \times S_j$ *column-shift-circulant matrix. Then* $\mathbf{DC}^{i,j+1}$ *and* $\mathbf{DD}^{i,j+1}$ *defined by (2.48) and (2.49), respectively, are column-shift-circulant matrices.*

Proof: We will give the proof for the case of $\mathbf{DC}^{i,j}$ only. By assumption that $\mathbf{DC}^{i,j}$ is column-shift-circulant, i.e.

$$DC_{m,n}^{i,j} = dc_{\langle m-\sigma n\rangle_{S_i}}^{i,j}$$

where $\mathbf{dc}^{i,j}$ is the first column of $\mathbf{DC}^{i,j}$ and $\sigma = S_i/S_j = 2^{j-i}$. Equation (2.48) then becomes

$$\begin{aligned} DC_{m,n}^{i,j+1} &= \sum_{l=0}^{D-1} a_l DC_{m,\langle l+2n\rangle_{S_j}}^{i,j} \\ &= \sum_{l=0}^{D-1} a_l dc_{\langle m-\sigma\langle l+2n\rangle_{S_j}\rangle_{S_i}}^{i,j} \end{aligned} \tag{2.61}$$

We consider the index of the typical term of (2.61) and use Lemma A.1.1 and A.1.2 in Appendix A to obtain the following:

$$\begin{aligned} \langle m-\sigma\langle l+2n\rangle_{S_j}\rangle_{S_i} &= \langle m-\langle\sigma\,(l+2n)\rangle_{\sigma S_j}\rangle_{S_i} \\ &= \langle m-\langle\sigma l+2\sigma n\rangle_{S_j}\rangle_{S_i} \\ &= \langle m-\sigma l-2\sigma n\rangle_{S_i} \end{aligned}$$

Therefore,

$$DC_{m,n}^{i,j+1} = \sum_{l=0}^{D-1} a_l dc_{\langle m-\sigma l-2\sigma n\rangle_{S_i}}^{i,j} \tag{2.62}$$

Equation (2.62) establishes the existence of a vector, $\mathbf{dc}^{i,j+1}$ say, such that $\mathbf{DC}^{i,j+1}$ has the desired column-shift-circulant form

$$DC_{m,n}^{i,j+1} = dc_{\langle m-2\sigma n\rangle_{S_i}}^{i,j+1}$$

for $m = 0, 1, \cdots, S_i - 1$ and $n = 0, 1, \cdots, S_{j+1} - 1$. We can now look for an explicit formula for the vector $\mathbf{dc}^{i,j+1}$. Taking the first column ($n = 0$) of $DC_{m,n}^{i,j+1}$ in (2.62) gives the result:

$$dc_m^{i,j+1} = DC_{m,0}^{i,j+1} = \sum_{l=0}^{D-1} a_l dc_{\langle m-\sigma l\rangle_{S_i}}^{i,j}, \quad m = 0, 1, \cdots, S_i - 1. \tag{2.63}$$

An analysis similar to the above establishes that $\mathbf{DD}^{i,j+1}$ is column-shift-circulant with respect to the vector $\mathbf{dd}^{i,j+1}$ given by

$$dd_m^{i,j+1} = DD_{m,0}^{i,j+1} = \sum_{l=0}^{D-1} b_l dc_{\langle m-\sigma l\rangle_{S_i}}^{i,j}, \quad m = 0, 1, \cdots, S_i - 1 \tag{2.64}$$

Since the initial block $\mathbf{DC}^{i,j}$ is circulant according to Lemma 2.13.1, it is also column-shift-circulant with $\sigma = S_i/S_i = 1$ and we can use the column vector $\mathbf{dc}^{i,j}$ as computed from (2.59) directly in (2.63) and (2.64) for the case $i = j$.

2.13.3 Blocks above the Diagonal

The *upper* off-diagonal blocks $\mathbf{CD}^{i+1,j}$ and $\mathbf{DD}^{i+1,j}$, $i \geq j$, are computed according to (2.50) and (2.51). The situation is completely analogous to (2.63) and (2.64) but the blocks are now row-shift-circulant matrices represented by row vectors $\mathbf{cd}^{i+1,j}$ and $\mathbf{dd}^{i+1,j}$, respectively, as stated by Lemma 2.13.3.

Lemma 2.13.3 *Let $\mathbf{CD}^{i,j}$, $i \geq j$ be a $S_i \times S_j$ row-shift-circulant matrix. Then $\mathbf{CD}^{i+1,j}$ and $\mathbf{DD}^{i+1,j}$ defined by (2.50) and (2.51), respectively, are row-shift-circulant matrices.*

Proof: We will now give the proof for the case of $\mathbf{CD}^{i,j}$ only. The proof is completely similar to that of Lemma 2.13.2, but the assumption is now that $\mathbf{CD}^{i,j}$ is row-shift-circulant, i.e.

$$CD_{m,n}^{i,j} = cd_{\langle n-\sigma m\rangle_{S_j}}^{i,j}$$

where $cd^{i,j}$ is the first row of $CD^{i,j}$ and $\sigma = S_i/S_j = 2^{i-j}$. Equation (2.50) then becomes

$$\begin{aligned} CD^{i+1,j}_{m,n} &= \sum_{k=0}^{D-1} a_k CD^{i,j}_{\langle k+2m\rangle_{S_i},n} \\ &= \sum_{k=0}^{D-1} a_k cd^{i,j}_{\langle n-\sigma\langle k+2m\rangle_{S_i}\rangle_{S_j}} \\ &= \sum_{k=0}^{D-1} a_k cd^{i,j}_{\langle n-\sigma k-2\sigma m\rangle_{S_j}} \end{aligned}$$

Therefore, $\mathbf{CD}^{i+1,j}$ has the desired row-shift-circulant form

$$CD^{i+1,j}_{m,n} = cd^{i+1,j}_{\langle n-\sigma m\rangle_{S_j}}$$

for $m = 0, 1, \cdots, S_{i+1} - 1$ and $n = 0, 1, \cdots, S_j - 1$.

The formulas for the row vectors $\mathbf{cd}^{i+1,j}$ and $\mathbf{dd}^{i+1,j}$ follow by taking the first rows ($m = 0$) of $\mathbf{CD}^{i+1,j}_{m,n}$ and $\mathbf{DD}^{i+1,j}_{m,n}$, respectively:

$$cd^{i+1,j}_n = CC^{i+1,j}_{0,n} = \sum_{k=0}^{D-1} a_k cd^{i,j}_{\langle n-\sigma k\rangle_{S_j}}, \quad n = 0, 1, \cdots, S_j - 1 \tag{2.65}$$

$$dd^{i+1,j}_n = DD^{i+1,j}_{0,n} = \sum_{k=0}^{D-1} b_k cd^{i,j}_{\langle n-\sigma k\rangle_{S_j}}, \quad n = 0, 1, \cdots, S_j - 1 \tag{2.66}$$

However, one minor issue remains to be dealt with before a viable algorithm can be established: While the initial blocks $\mathbf{CD}^{i,i}$ defined according to (2.45) are circulant and therefore also row-shift-circulant (with $\sigma = 1$), they are represented by column vectors when computed according to equation (2.57). However, (2.65) and (2.66) work with a row vector $\mathbf{cd}^{i,j}$. Therefore, we must modify each $\mathbf{cd}^{i,i}$ so that it represents the first row of $\mathbf{CD}^{i,i}$ instead of the first column. From Definition A.3.1 of a circulant matrix, we have that

$$CD^{i,i}_{m,n} = cd^{i,i}_{\langle m-n\rangle_{S_i}}, \quad n = 0, 1, \cdots, S_i - 1$$

where $\mathbf{cd}^{i,i}$ is the first column of $\mathbf{CD}^{i,i}$. Putting $m = 0$ then yields the first row:

$$CD^{i,i}_{0,n} = cd^{i,i}_{\langle -n\rangle_{S_i}}, \quad n = 0, 1, \cdots, S_i - 1.$$

To obtain a row representation for $\mathbf{CD}^{i,i}$, we can therefore take the result from equation (2.57) and convert it as follows

$$cd^{i,i}_n \leftarrow cd^{i,i}_{\langle -n\rangle_{S_i}}.$$

Alternatively, we can modify equation (2.57) to produce the row vector directly:

$$cd_n^{i+1,j+1} = q_{ab}^0 cc_{\langle -2n \rangle_{S_i}}^{i,i} + \sum_{p=1}^{D-1} \left[q_{ab}^p cc_{\langle -2n-p \rangle_{S_i}}^{i,i} + q_{ba}^p cc_{\langle -2n+p \rangle_{S_i}}^{i,i} \right] \tag{2.67}$$

for $n = 0, 1, \cdots, S_{i+1} - 1$. The equations for blocks above the diagonal (2.65) and (2.66) and equations for blocks below the diagonal (2.63) and (2.64) can now be computed with the same algorithm.

2.13.4 Algorithm

We will now state an algorithm for the $2D$ wavelet transform of a circulant matrix $\mathbf{A}$. Let $CIRPWT1$ be a function that implements $2D$ decompositions of blocks on the diagonal according to (2.58), (2.59), (2.60) and (2.67):

$$\left[\mathbf{cc}^{i+1,i+1}, \mathbf{cd}^{i+1,i+1}, \mathbf{dc}^{i+1,i+1}, \mathbf{dd}^{i+1,i+1}\right] = CIRPWT1(\mathbf{cc}^{i,i}).$$

Moreover, let $CIRPWT2$ be a function that implements $1D$ decompositions of the form described in equations (2.63) and (2.64). This function can also be used for computations of (2.65) and (2.66) as mentioned above.

$$\left[\mathbf{dc}^{i,j+1}, \mathbf{dd}^{i,j+1}\right] = CIRPWT2(\mathbf{dc}^{i,j}), \quad j \geq i$$

$$\left[\mathbf{cd}^{i+1,j}, \mathbf{dd}^{i+1,j}\right] = CIRPWT2(\mathbf{cd}^{i,j}), \quad i \geq j.$$

With these functions the example shown in Figure-2.2 can be computed as follows:

Let $\mathbf{cc}^{0,0}$ be the column vector representing the initial circulant matrix $\mathbf{CC}^{0,0}$. Then

$$\begin{aligned}
\left[\mathbf{cc}^{1,1}, \mathbf{cd}^{1,1}, \mathbf{dc}^{1,1}, \mathbf{dd}^{1,1}\right] &= CIRPWT1(\mathbf{cc}^{0,0}) \\
\left[\mathbf{cc}^{2,2}, \mathbf{cd}^{2,2}, \mathbf{dc}^{2,2}, \mathbf{dd}^{2,2}\right] &= CIRPWT1(\mathbf{cc}^{1,1}) \\
\left[\mathbf{dc}^{1,2}, \mathbf{dd}^{1,2}\right] &= CIRPWT2(\mathbf{dc}^{1,1}) \\
\left[\mathbf{cd}^{2,1}, \mathbf{dd}^{2,1}\right] &= CIRPWT2(\mathbf{cd}^{1,1}) \\
\left[\mathbf{cc}^{3,3}, \mathbf{cd}^{3,3}, \mathbf{dc}^{3,3}, \mathbf{dd}^{3,3}\right] &= CIRPWT1(\mathbf{cc}^{2,2}) \\
\left[\mathbf{dc}^{2,2}, \mathbf{dd}^{2,3}\right] &= CIRPWT2(\mathbf{dc}^{2,2}) \\
\left[\mathbf{cd}^{3,2}, \mathbf{dd}^{3,2}\right] &= CIRPWT2(\mathbf{cd}^{2,2}) \\
\left[\mathbf{dc}^{1,3}, \mathbf{dd}^{1,3}\right] &= CIRPWT2(\mathbf{dc}^{1,2}) \\
\left[\mathbf{cd}^{3,1}, \mathbf{dd}^{3,1}\right] &= CIRPWT2(\mathbf{cd}^{2,1})
\end{aligned}$$

In general, the algorithm is

For $i = 0, 1, \cdots, \lambda - 1$
$\left[\mathbf{cc}^{i+1,i+1}, \mathbf{cd}^{i+1,i+1}, \mathbf{dc}^{i+1,i+1}, \mathbf{dd}^{i+1,i+1}\right] = CIRPWT1(\mathbf{cc}^{i,i})$
For $j = i, i-1, \cdots, 1$
$\left[\mathbf{dc}^{i,j+1}, \mathbf{dd}^{i,j+1}\right] = CIRPWT2(\mathbf{dc}^{i,j})$
$\left[\mathbf{cd}^{i+1,j}, \mathbf{dd}^{i+1,j}\right] = CIRPWT2(\mathbf{cd}^{i,j})$
end
end

This algorithm describes the process of computing the $2D$ wavelet transform as described in the previous section. However, it dies not distinguish among vectors that should be kept in the final result and vectors that are merely intermediate stages of the transform. The vectors $\mathbf{dc}^{j,i}$ and $\mathbf{cd}^{i,j}$, for example, are part of the final result for $i = \lambda$ only, so all other vectors can be discarded at some point.

In practice, we prefer an algorithm that makes explicit use of a fixed storage area and that does not store unnecessary information. Therefore, we will introduce a modified notation that is suitable for such an algorithm that makes explicit use of a fixed storage area and that does not store unnecessary information. Therefore, we will introduce a modified notation that is suitable for such an algorithm and also convenient for the matrix-vector multiplication which is described in Section 2.15.

2.13.5 A Data Structure for the $2D$ Wavelet Transform

As described in Section 2.12.2, the result $\mathbf{H}$ of a $2D$ wavelet transform is a block matrix with a characteristic block structure (see Figure-2.2). We now introduce a new notation for these blocks as follows

$$\mathbf{H}^{i,j} = \begin{cases} \mathbf{CC}^{\lambda,\lambda} & \text{for } i,j = 0 \\ \mathbf{CD}^{\lambda,\lambda-j+1} & \text{for } i = 0, 1 \le j \le \lambda \\ \mathbf{DC}^{\lambda-i+1,\lambda} & \text{for } 1 \le i \le \lambda, j = 0 \\ \mathbf{DD}^{\lambda-i+1,\lambda-j+1} & \text{for } 1 \le i,j \le \lambda \end{cases} \tag{2.68}$$

We can now use indices $i, j = 0, 1, \cdots, \lambda$ to label the blocks of $\mathbf{H}$ in a straight-forward manner. It follows from the definition that each block $\mathbf{H}^{i,j}$ is an $N^i \times N^j$ matrix with

$$N^k = \begin{cases} N/2^{\lambda} &= S_{\lambda} \text{for } k = 0 \\ N/2^{\lambda-k+1} &= S_{\lambda-k+1} \text{for } 1 \le k \le \lambda \end{cases} \tag{2.69}$$

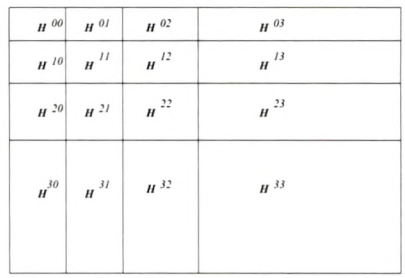


Figure 2.6: **The new notations for the block structure of H (for $\lambda = 3$).**

Note that $N^0 = N^1$. Because all the blocks $\mathbf{H}^{i,j}$ are either circulant or shift-circulant, we can represent them by vectors $\mathbf{h}^{i,j}$ with

$$h_m^{i,j} = \begin{cases} H_{m,0}^{i,j} & \text{for } i \geq j \\ H_{0,m}^{i,j} & \text{for } i < j \end{cases} \tag{2.70}$$

where $m = 0, 1, \cdots, \max\{N^i, N^j\} - 1$. It follows form (2.69) that the length of $\mathbf{h}^{i,j}$ is

$$\begin{cases} N^i & \text{for } i \geq j \\ N^j & \text{for } i < j \end{cases} \tag{2.71}$$

and the shift parameter σ is now given according to Definition-2.13.1 as

$$\sigma = \begin{cases} \frac{N^i}{N^j} & \text{for } i \geq j \\ \frac{N^j}{N^i} & \text{for } i < j \end{cases}$$

Equation (2.70) suggests a data structure consisting of the vector variables $\mathbf{h}^{i,j}$ which refer to the actual arrays representing the final stage of the wavelet transform (e.g. $\mathbf{dd}^{\lambda-i+1,\lambda-j+1}$). This structure can also be used to store the intermediate vectors from the recursion if we allow the variables $h^{i,j}$ to assume different values (and different lengths) during the computation - for example by using pointer variables. This is demonstrated in Figure-2.7. From Figure-2.7, we arrive at the final formulation of the algorithm in Subsection 2.13.4,

Algorithm: Circulant $2D$ wavelet transform ($CIRFWT$)

$\mathbf{h}^{\lambda,\lambda} \leftarrow \mathbf{cc}^{0,0}$

For $j = \lambda, \lambda - 1, \ldots\ldots, 1.$
$[\mathbf{h}^{-1,j-1}, \mathbf{h}^{-1,j}, \mathbf{h}^{-j,j-1}, \mathbf{h}^{j,j}] \leftarrow CIRPWT1(\mathbf{h}^{i,j})$
For $i = j + 1, j + 2, \ldots\ldots, \lambda.$

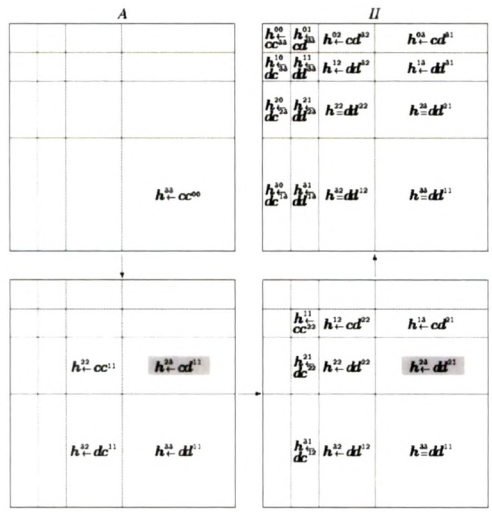


Figure 2.7: **The use of variables h_{ij} to implement the two dimensional wavelet transform of circulant matrix.**

$$[\mathbf{h}^{i,j-1}, \mathbf{h}^{i,j}] \leftarrow CIRPWT2(\mathbf{h}^{i,j})$$
$$[\mathbf{h}^{j-1,i}, \mathbf{h}^{j,i}] \leftarrow CIRPWT2(\mathbf{h}^{j,i})$$
end
end

where $CIRPWT1$ is derived from (2.58), (2.59), (2.60), and (2.67):

$$h_m^{j-1,j-1} \leftarrow q_{aa}^0 h_{2m}^{j,j} + \sum_{p=1}^{D-1} \left[q_{aa}^p h_{<2m-p>_{Nj}}^{j,j} + q_{aa}^p h_{<-2m+p>_{Nj}}^{j,j} \right] \tag{2.72}$$

$$h_m^{j-1,j} \leftarrow q_{ab}^0 h_{\langle -2m \rangle_{Nj}}^{j,j} + \sum_{p=1}^{D-1} \left[q_{ab}^p h_{<-2m-p>_{Nj}}^{j,j} + q_{aa}^p h_{<-2m+p>_{Nj}}^{j,j} \right] \tag{2.73}$$

$$h_m^{j,j-1} \leftarrow q_{ba}^0 h_{2m}^{j,j} + \sum_{p=1}^{D-1} \left[q_{ba}^p h_{<2m-p>_{Nj}}^{j,j} + q_{ab}^p h_{<2m+p>_{Nj}}^{j,j} \right] \tag{2.74}$$

$$h_m^{j,j} \leftarrow q_{bb}^0 h_{2m}^{j,j} + \sum_{p=1}^{D-1} \left[q_{bb}^p h_{<2m-p>_{Nj}}^{j,j} + q_{bb}^p h_{<2m+p>_{Nj}}^{j,j} \right] \tag{2.75}$$

for $m = 0, 1, 2, \cdots, N^{j-1} - 1$. For $\geq j$, $CIRPWT2$ is derived from (2.63) and (2.64):

$$h_m^{i,j-1} \leftarrow \sum_{l=0}^{D-1} a_l h_{<m-\sigma l>_{N^i}}^{i,j} \tag{2.76}$$

$$h_m^{i,j} \leftarrow \sum_{l=0}^{D-1} b_l h_{<m-\sigma l>_{N^i}}^{i,j} \tag{2.77}$$

for $m = 0, 1, 2, \cdots, N^{j-1} - 1$. For $i < j$ $CIRPWT2$ is:

$$h_m^{i-1,j} \leftarrow \sum_{l=0}^{D-1} a_l h_{<m-\sigma l>_{N^j}}^{i,j} \tag{2.78}$$

$$h_m^{i,j} \leftarrow \sum_{l=0}^{D-1} b_l h_{<m-\sigma l>_{N^j}}^{i,j} \tag{2.79}$$

where $m = 0, 1, 2, \cdots, N^j - 1$. Note that we use exactly the same code for $CIRPWT2$ by exchanging the indices i and j in the above algorithm. Our algorithm now takes the form

$$\begin{aligned}
[\mathbf{h}^{2,2}, \mathbf{h}^{2,3}, \mathbf{h}^{3,2}, \mathbf{h}^{3,3}] &\leftarrow CIRPWT1(\mathbf{h}^{3,3}) \\
[\mathbf{h}^{1,1}, \mathbf{h}^{1,2}, \mathbf{h}^{2,1}, \mathbf{h}^{2,2}] &\leftarrow CIRPWT1(\mathbf{h}^{2,2}) \\
[\mathbf{h}^{3,1}, \mathbf{h}^{3,2}] &\leftarrow CIRPWT1(\mathbf{h}^{3,2}) \\
[\mathbf{h}^{3,1}, \mathbf{h}^{3,2}] &\leftarrow CIRPWT1(\mathbf{h}^{3,2}) \\
[\mathbf{h}^{0,0}, \mathbf{h}^{0,1}, \mathbf{h}^{1,0}, \mathbf{h}^{1,1}] &\leftarrow CIRPWT1(\mathbf{h}^{1,1}) \\
[\mathbf{h}^{2,0}, \mathbf{h}^{2,1}] &\leftarrow CIRPWT1(\mathbf{h}^{2,1}) \\
[\mathbf{h}^{0,2}, \mathbf{h}^{1,2}] &\leftarrow CIRPWT1(\mathbf{h}^{1,2}) \\
[\mathbf{h}^{3,0}, \mathbf{h}^{3,1}] &\leftarrow CIRPWT1(\mathbf{h}^{3,1}) \\
[\mathbf{h}^{0,3}, \mathbf{h}^{1,3}] &\leftarrow CIRPWT1(\mathbf{h}^{1,3})
\end{aligned}$$

2.13.6 Computational Work

The estimate for complexity of an algorithm in Subsection 2.13.5 is as follows:

Lemma 2.13.4

$$\sum_{k=0}^{\lambda-1} N^k = \frac{N}{2}.$$

Proof: See [Nie98].

2.13.7 Storage

In this section, we will investigate the storage requirement for **H** as a function of the transform depth λ when the data structure proposed in Subsection 2.13.5 is used.

Lemma 2.13.5 *The number of elements needed to represent the result of λ steps of the wavelet transform of a circulant $N \times N$ matrix as computed by algorithm in Subsection 2.13.5 is*

$$S_N(\lambda) = N\left(1 + \sum_{k=1}^{\lambda} \frac{k}{2^{\lambda-k}}\right), \lambda = 0, 1, 2, ..., \lambda_{max}. \tag{2.80}$$

Proof: Ssee [Nie98].

2.14 2D Wavelet Transform of a Circulant, Banded Matrix

An important special case of a circulant matrix is when **A** is a banded circulant matrix such as the differentiation matrix **D** given in (4.7). In this case, each column of **A** consists of a piece which is zero and a piece which is regarded as non-zero. In certain columns, the non-zero part is wrapped around. Consequently, it is sufficient to store only the non-zero part of the first column along with an index δ determining how if must be aligned in the first column relative to the full-length vector. The length of the non-zero part is the **bandwidth** of **A**, and we will denote it by L.

It turns out that each block of the $2D$ wavelet transform retains a banded structure, so the vector representing it need only include the non-zero part. Therefore, the storage requirements can be considerably less than that given by (2.80). An example of this structure is given in Figure-2.8. For each block, we know the non-zero values of the first column (or row in the case of blocks above the diagonal) represented by the vector $\mathbf{v}^{i,j} = [v_0^{i,j}, v_1^{i,j}, \cdots, v_{L-1}^{i,j}]^T$, the amount by which it is shifted (σ) and how it is aligned with respect to the first element of the block

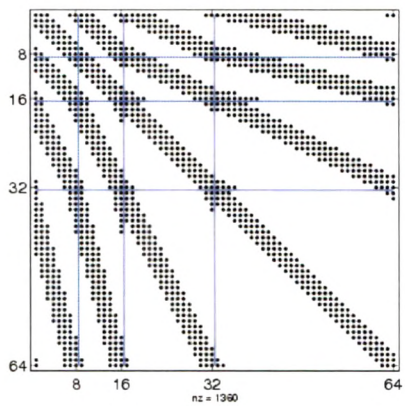


Figure 2.8: **The structure of a wavelet transform of a 64×64 circulant banded matrix. Here, $L = 3$, $D = 4$ and $\lambda = 3$.**

(δ).The block $\mathbf{H}^{3,1}$, say, has the general structure

$$\mathbf{H}^{3,1} = \begin{bmatrix} v_2^{3,1} & & & & \\ v_3^{3,1} & & & & \\ v_4^{3,1} & v_0^{3,1} & & & \\ & v_1^{3,1} & & & \\ & v_2^{3,1} & & & \\ & v_3^{3,1} & & & \\ & v_4^{3,1} & v_0^{3,1} & & \\ & & v_1^{3,1} & & \\ & & v_2^{3,1} & & \\ & & v_3^{3,1} & & \\ & & v_4^{3,1} & v_0^{3,1} & \\ & & & v_1^{3,1} & \\ & & & v_2^{3,1} & \\ & & & v_3^{3,1} & \\ & & & v_4^{3,1} & \\ v_0^{3,1} & & & & \\ v_1^{3,1} & & & & \end{bmatrix} \tag{2.81}$$

In this example, we have $\delta = 3$, $N^3 = 16$, $N^1 = 4$, $\sigma = 4$ (from Definition 2.13.1), bandwidth $L^{3,1} = 5$, and

$$h_m^{3,1} = \begin{cases} v_{<m+2>_{16}}^{3,1} & \text{for } < m+2 >_{16} \in [0,4] \\ 0 & \text{otherwise} \end{cases}$$

with

$$H^{3,1}_{m,n} = h^{3,1}_{<m-4n>_{16}}$$

If the matrix is not banded we have the special case $\mathbf{v}^{i,j} = \mathbf{h}^{i,j}$ and $\delta = 1$ so (2.80) applies exactly.

2.14.1 Calculation of Bandwidths

Given the bandwidth L of the original matrix $\mathbf{A}$, it is possible to derive a formula for the bandwidths of each block of the wavelet transform.

Let $L^{i,j}$ be the bandwidth of block $\mathbf{H}^{i,j}$ shown in Figure-2.6. We can then use the recurrence formulas for the $2D$ wavelet transform to obtain the desired formulas.

Blocks on the diagonal

We start with the blocks on the diagonal given by equation (2.44) to (2.47) and consider again only the cd block as the typical case. Let us recall (2.72):

$$h^{j-1,j-1}_m \leftarrow q^0_{aa} h^{j,j}_{2m} + \sum_{p=1}^{D-1} \left[q^p_{aa} h^{j,j}_{<2m-p>_{N^j}} + q^p_{aa} h^{j,j}_{<2m+p>_{N^j}} \right]$$

for $m = 0, 1, \cdots, N^{j-1} - 1$. Assume that $\mathbf{h}^{j,j}$ is banded, i.e. zero outside a band of length $L^{j,j}$ as shown in Figure-2.9. Then we can use the recurrence formula to compute the bandwidth of $\mathbf{h}^{j-1,j-1}$. Without loss of generality we may assume that the nonzero part is wholly contained in the vector; i.e. there is no wrap-around.

Since $\mathbf{h}^{j,j}$ is zero outside the interval starting at m_1 of length $L^{j,j}$, we see that $h^{j-1,j-1}_m$ will be zero only if $2m - p \geq m_1 + L^{j,j}$ or $2m + p < m_1$ for all $p \in [0, D-1]$. This leads immediately to the inequalities

$$2m + (D-1) < m_1$$

$$2m - (D-1) \geq m_1 + L^{j,j}$$

or

$$m \geq \frac{m_1 + L^{j,j} + D - 1}{2}$$

or

$$m < \frac{m_1 - D + 1}{2}.$$

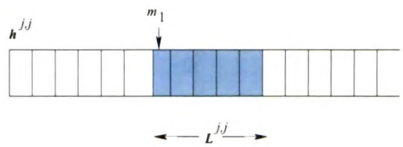


Figure 2.9: **The band of $\mathbf{h}^{i,j}$ has length $L^{i,j}$ and start at index m_1.**

The length of this interval is then the bandwidth of $\mathbf{h}^{j-1,j-1}$:

$$\begin{aligned} L^{j-1,j-1} &= \frac{m_1 + L^{j,j} + D - 1}{2} - \frac{m_1 - D + 1}{2} \\ &= \frac{L^{j,j}}{2} + D - 1 \end{aligned} \tag{2.82}$$

However, since the bandwidth is an integer the fraction must be rounded either up or down if $L^{j,j}$ is odd. Which of these operations to choose depends on m_1 as illustrated in Figure-2.10. We have chosen always to round upwards because it yields an upper bound for the bandwidth. Thus, the formula becomes

$$L^{j-1,j-1} = \lceil \frac{L^{j,j}}{2} \rceil + D - 1 \tag{2.83}$$

We observe that equation (2.83) has two fixed points, namely

$$L^{j-1,j-1} = L^{j,j} \begin{cases} 2D-2 \\ 2D-1 \end{cases}$$

and it turns out that there is convergence to one of these values depending on whether the initial $L^{j,j}$ is smaller than $2D-2$ or larger than $2D-1$. However, the important fact is that these fixed points are more related to the wavelet genus D than to the original bandwidth L.

Blocks below and above the diagonal

The bandwidths of the blocks below the diagonal are found from recurrence formulas of the form

$$h_m^{i,j-1} = \sum_{t=0}^{D-1} a_l h^{i,j}_{\langle m - \sigma l \rangle_{N^i}}.$$

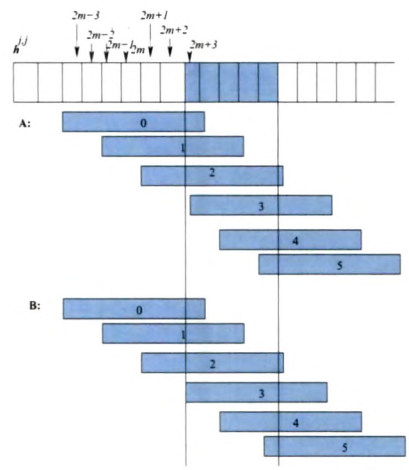


Figure 2.10: **The computation in equation (2.57) can be viewed as a sliding filter of length $2D-1$ applied to the band of $\mathbf{h}^{j,j}$. The numbers indicate offset with respect to $2m = \frac{m_1 - D + 1}{2}$. The resulting bandwidth depends on how the initial bandwidth is aligned with this sliding filter. In this example $L^{j,j} = 5$, $D = 4$ so $L^{j-1,j-1}$ is either 5 (case B) or 6 (case A) depending on how the convolutions happen to align with the non-zero block. Thus, case A corresponds to rounding up and case B corresponds to rounding down in (2.82).**

Again, we disregard wrapping and, proceeding as in the case of $\mathbf{h}^{j-1,j-1}$ above, we find that $h_m^{i,j-1}$ is zero only if $m - \sigma l \geq m_1 + L^{i,j}$ or $m - \sigma l < m_1$ for all $l \in [0, D-1]$. This leads to the inequalities

$$m < m_1$$

$$m - \sigma(D-1) \geq m_1 + L^{i,j}$$

Consequently, the interval length for which $h_m^{i,j-1} \neq 0$ is

$$\begin{aligned} L^{i,j-1} &= m_1 + L^{i,j} + \sigma(D-1) - m_1 \\ &= L^{i,j} + \sigma(D-1) \end{aligned}$$

Performing similar computations for blocks above the diagonal yields the result

$$L^{i,j-1} = L^{j-1,i} = L^{i,j} + \sigma(D-1) \tag{2.84}$$

Since σ is the ratio between N^i and N^j, equation (2.84) shows that the bandwidth grows exponentially as the difference between i and j increases. Figures-2.11 and 2.12 show examples of the bandwidths for different transform levels.

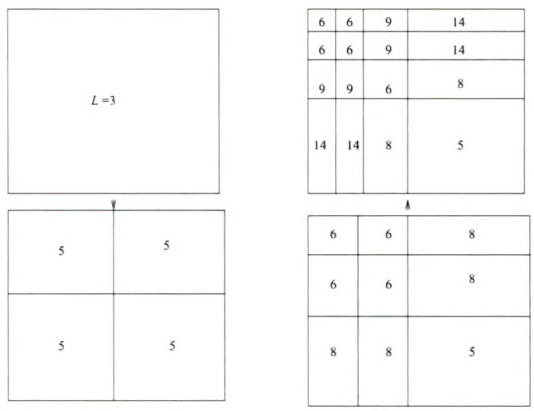


Figure 2.11: **The difference between a wavelet transform of depth $\lambda - 1$ and λ for $\lambda = 3$.**

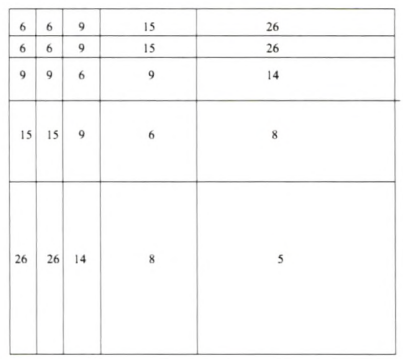


Figure 2.12: **The bandwidths for depths $\lambda = 0, 1, 2.3$. The initial bandwidth is 3 and $D = 4$.**

2.15 Matrix-vector Multiplication in a Wavelet Basis

We now turn to the problem of computing the matrix-vector product $\mathbf{y} = \mathbf{H}\mathbf{x}$ where $\mathbf{x}, \mathbf{y} \in \mathbf{R}^N$ and $\mathbf{H}$ is given as in (2.68). This system has the form shown in Figure-2.13.

The vector y may be computed block-wise as follows

$$\mathbf{y}^i = \sum_{j=0}^{\lambda} \mathbf{H}^{i,j}\mathbf{x}^j, \quad i = 0, 1, \cdots, \lambda \tag{2.85}$$

where λ is the depth of the wavelet transform. The symbols $\mathbf{x}^j$, $\mathbf{y}^i$, and $\mathbf{H}^{i,j}$ denote the different blocks of the $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{H}$ as indicated in Figure-2.6. The computation in (2.85) is thus broken

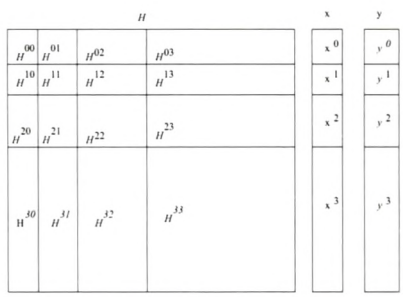


Figure 2.13: **The structure of y = Hx for $\lambda = 3$.**

down into the tasks of computing the products which we will denote by

$$\mathbf{y}^{i,j} = \mathbf{H}^{i,j}\mathbf{x}^j, \quad i,j = 0,1,\cdots,\lambda-1 \tag{2.86}$$

In the following, we distinguish between blocks on or below the diagonal($i \geq j$), and blocks above the diagonal ($i < j$).

2.15.1 Blocks on or below the diagonal

Let $\mathbf{v}^{i,j}, i \geq j$, be the vector of length $L^{i,j}$ representing the non-zero part of the first column of $\mathbf{H}^{i,j}$, i.e.

$$h_m^{i,j} = \begin{cases} v^{i,j}_{<m+\delta-1>_{N^i}} & \text{for } < m+\delta-1 >_{N^i} \in [0, L^{i,j}-1] \\ 0 & \text{otherwise} \end{cases}$$

and

$$\mathbf{H}^{i,j}_{m,n} = h^{i,j}_{\langle m-\sigma n\rangle_{N^i}} \tag{2.87}$$

where $m = 0,1,\cdots,N^i$, $\sigma = \frac{N^i}{N^j}$, and δ is the offset relative to the upper left element (see (2.81)). From equation (2.86), we see that the typical element of $y^{i,j}$ can be computed column wise as

$$\begin{aligned} y^{i,j}_m &= \sum_{n=0}^{N^j-1} H^{i,j}_{m,n} x^j_n \\ &= \sum_{n=0}^{N^j-1} h^{i,j}_{\langle m-\sigma n\rangle_{N^i}} x^j_n \\ &= \sum_{n=0}^{N^j-1} v^{i,j}_{\langle m-\sigma n+\delta-1\rangle_{N^i}} x^j_n \end{aligned} \tag{2.88}$$

$$\tag{2.89}$$

For each n, this computation is only valid for those $m \in [0, N^i - 1]$ where $v^{i,j}_{<m-\sigma m+\delta-1>^i_N}$ is defined, namely where

$$0 \leq \langle m - \sigma n + \delta - 1 \rangle^i_N \leq L^{i,j} - 1. \tag{2.90}$$

Let k and l be defined such that $k \leq m \leq l$ whenever (2.90) is satisfied. Then we can find k from the requirement

$$\begin{aligned} \langle k - \sigma n + \delta - 1 \rangle_{N^i} &= 0 \\ \Leftrightarrow k &= \langle \sigma n - \delta + 1 \rangle_{N^i} \end{aligned}$$

and the last row as $l = \langle k + L^{i,j} - 1 \rangle_{N^i}$. Letting

$$y^{i,j}_{k:l} = \left[y^{i,j}_k, y^{i,j}_{k+1}, \cdots, y^{i,j}_l \right]$$

then we can write the computation (2.88) compactly as

$$y^{i,j}_{k:l} = y^{i,j}_{k:l} + x^j_n v^{i,j}_{0:L^{i,j}-1}, \qquad n = 0, 1, \cdots, N^j - 1. \tag{2.91}$$

When $k > l$, the band is wrapped and (2.91) must be modified accordingly.

If the vector $\mathbf{x}$ is a wavelet spectrum then many of its elements are normally close to zero as described earlier. Therefore, we will design the algorithm to disregard computations involving elements in x^j where

$$|x^j_n| < \epsilon.$$

The algorithm is given below.

Algorithm: $\mathbf{y}^{i,j} = \mathbf{H}^{i,j}\mathbf{x}^j, \quad i \geq j$

For $n = 0$ *to* $N^j - 1$

if $|x^j_n| > \epsilon$ *then*

$k = \langle \sigma n - \delta + 1 \rangle_{N^i}$

$l = \langle k + L^{i,j} - 1 \rangle_{N^i}$

if $k < l$ *then*

$y^{i,j}_{k:l} = y^{i,j}_{k:l} + x^j_n v^{i,j}$

else (*wrap*)

$y^{i,j}_{0:l} = y^{i,j}_{0:l} + x^j_n v^{i,j}_{L^{i,j}-l:L-1}$

$y^{i,j}_{k:N^i-1} = y^{i,j}_{k:N^i-1} + x_n v^{i,j}_{0:L^{i,j}-l-l}$

end

end

end