
The following is based on book manuscript: B. Han, Framelets and Wavelets: Algorithms, Analysis and Applications.

To introduce a discrete framelet transform, we need some definitions and notation. By \( l(\mathbb{Z}) \) we denote the linear space of all sequences \( v = \{v(k)\}_{k \in \mathbb{Z}} : \mathbb{Z} \to \mathbb{C} \) of complex numbers on \( \mathbb{Z} \). One-dimensional discrete input data or signal is often treated as an element in \( l(\mathbb{Z}) \). Similarly, by \( l_0(\mathbb{Z}) \) we denote the linear space of all sequences \( u = \{u(k)\}_{k \in \mathbb{Z}} : \mathbb{Z} \to \mathbb{C} \) on \( \mathbb{Z} \) such that \( \{k \in \mathbb{Z} : u(k) \neq 0\} \) is a finite set. An element in \( l_0(\mathbb{Z}) \) is often regarded as a finite-impulse-response (FIR) filter (also called a finitely supported mask in the literature of wavelet analysis). In this book we often use \( u \) for a general filter and \( v \) for a general signal or data. It is often convenient to use the formal Fourier series (or symbol) \( \hat{v} \) of a sequence \( v = \{v(k)\}_{k \in \mathbb{Z}} \), which is defined as follows:

\[
\hat{v}(\xi) := \sum_{k \in \mathbb{Z}} v(k)e^{-ik\xi}, \quad \xi \in \mathbb{R},
\]

where \( i \) in this book always denotes the imaginary unit. For \( v \in l_0(\mathbb{Z}) \), \( \hat{v} \) is a \( 2\pi \)-periodic trigonometric polynomial.

A discrete framelet transform can be described using two linear operators—the subdivision operator and the transition operator. For a filter \( u \in l_0(\mathbb{Z}) \) and \( v \in l(\mathbb{Z}) \), the subdivision operator \( \mathcal{S}_u : l(\mathbb{Z}) \to l(\mathbb{Z}) \) is defined to be

\[
[\mathcal{S}_u][v](n) := 2 \sum_{k \in \mathbb{Z}} v(k)u(n - 2k), \quad n \in \mathbb{Z}
\]

and the transition operator \( \mathcal{T}_v : l(\mathbb{Z}) \to l(\mathbb{Z}) \) is defined to be

\[
[\mathcal{T}_v][v](n) := 2 \sum_{k \in \mathbb{Z}} v(k)u(n - 2k), \quad n \in \mathbb{Z}.
\]

The transition operator plays the role of coarsening and frequency-separating the data to lower resolution levels; while the subdivision operator plays the role of refining and predicting the data to higher resolution levels.

In terms of Fourier series, the subdivision operator \( \mathcal{S}_u \) in (2) and the transition operator \( \mathcal{T}_v \) in (3) can be equivalently rewritten as

\[
\mathcal{S}_u v(\xi) = \hat{v}(2\xi)\hat{u}(\xi), \quad \xi \in \mathbb{R}
\]

and

\[
\mathcal{T}_v v(\xi) = \hat{v}(\xi/2)\overline{\hat{u}(\xi/2)} + \hat{v}(\xi/2 + \pi)\overline{\hat{u}(\xi/2 + \pi)}, \quad \xi \in \mathbb{R}
\]

for \( u, v \in l_0(\mathbb{Z}) \), where \( \overline{\cdot} \) denotes the complex conjugate of a complex number \( c \in \mathbb{C} \).

Let \( \tilde{a}, \tilde{b}_1, \ldots, \tilde{b}_s \) be filters for decomposition. For a positive integer \( J \), a \( J \)-level discrete framelet decomposition is given by

\[
v_j := \frac{\sqrt{2}}{2} \mathcal{S}_a v_{j-1}, \quad w_{\ell,j} := \frac{\sqrt{2}}{2} \mathcal{T}_{\tilde{b}_\ell} v_{j-1}, \quad \ell = 1, \ldots, s, \quad j = 1, \ldots, J,
\]

where \( v_0 : \mathbb{Z} \to \mathbb{C} \) is an input signal. The filter \( \tilde{a} \) is often called a dual low-pass filter and the filters \( \tilde{b}_1, \ldots, \tilde{b}_s \) are called dual high-pass filters. After a \( J \)-level discrete framelet
decomposition, the original input signal \( v_0 \) is decomposed into one sequence \( v_J \) of low-pass framelet coefficients and \( s \) \( J \) sequences \( w_{\ell,j} \) of high-pass framelet coefficients for \( \ell = 1, \ldots, s \) and \( j = 1, \ldots, J \). Such framelet coefficients are often processed for various purposes. One of the most commonly employed operations is thresholding so that the low-pass framelet coefficients \( v_J \) and high-pass framelet coefficients \( w_{\ell,j} \) become \( \hat{v}_j \) and \( \hat{w}_{\ell,j} \), respectively. More precisely, \( \hat{w}_{\ell,j}(k) = \eta(w_{\ell,j}(k)), k \in \mathbb{Z} \), where \( \eta : \mathbb{C} \to \mathbb{C} \) is a thresholding function. For example, for a given threshold value \( \lambda > 0 \), the hard thresholding function \( \eta^\text{hard}_\lambda \) and soft-thresholding function \( \eta^\text{soft}_\lambda \) are defined to be

\[
\eta^\text{hard}_\lambda(z) = \begin{cases} 
    z, & \text{if } |z| \geq \lambda; \\
    0, & \text{otherwise}
\end{cases}
\]

\[
\eta^\text{soft}_\lambda(z) = \begin{cases} 
    z - \epsilon \left\lfloor \frac{z}{\lambda} \right\rfloor, & \text{if } |z| \geq \lambda; \\
    0, & \text{otherwise}
\end{cases}
\]

Another commonly employed operation is quantization, which can be applied after or without thresholding. For example, for a given quantization level \( q > 0 \), the quantization function \( \mathcal{Q} : \mathbb{R} \to q\mathbb{Z} \) is defined to be \( \mathcal{Q}(x) := q \left\lfloor \frac{x}{q} + \frac{1}{2} \right\rfloor, x \in \mathbb{R} \), where \( \lfloor \cdot \rfloor \) is the floor function such that \( \lfloor x \rfloor = n \) if \( n \leq x < n + 1 \) for an integer \( n \).

Let \( a, b_1, \ldots, b_s \) be filters for reconstruction. Now a \( J \)-level discrete framelet reconstruction is

\[
\hat{v}_{j-1} := \frac{\sqrt{2}}{2} \mathcal{R}_a \hat{v}_j + \sqrt{2} \sum_{\ell=1}^s \mathcal{R}_{b_\ell} \hat{w}_{\ell,j}, \quad j = J, \ldots, 1.
\]

The filter \( a \) is often called a primal low-pass filter and the filters \( b_1, \ldots, b_s \) are called primal high-pass filters.

We say that \( \{ \{ \hat{a}; \hat{b}_1, \ldots, \hat{b}_s \}, \{ a; b_1, \ldots, b_s \} \} \) is a dual framelet filter bank if it satisfies the perfect reconstruction condition:

\[
\begin{bmatrix}
    \hat{a}(\xi) & \hat{b}_1(\xi) & \cdots & \hat{b}_s(\xi) \\
    \hat{a}(\xi + \pi) & \hat{b}_1(\xi + \pi) & \cdots & \hat{b}_s(\xi + \pi)
\end{bmatrix}^* \begin{bmatrix}
    \hat{a}(\xi) & \hat{b}_1(\xi) & \cdots & \hat{b}_s(\xi) \\
    \hat{a}(\xi + \pi) & \hat{b}_1(\xi + \pi) & \cdots & \hat{b}_s(\xi + \pi)
\end{bmatrix} = I_2,
\]

\{ \{ a; b_1, \ldots, b_s \} \} \) is called a tight framelet filter bank if \( \{ \{ a; b_1, \ldots, b_s \}, \{ a; b_1, \ldots, b_s \} \} \) is a dual framelet filter bank.
If \( s = 1 \), a dual framelet filter bank \( \{ \{ a; b \} \} \) is called a biorthogonal wavelet filter bank. If \( s = 1 \), a tight framelet filter bank \( \{ a; b \} \) is called an orthogonal wavelet filter bank.

In the following, let us provide a few examples to illustrate various types of filter banks. For a filter \( u = \{ u(k) \}_{k \in \mathbb{Z}} \) such that \( u(k) = 0 \) for all \( k \in \mathbb{Z} \setminus \{ m, n \} \) and \( u(m)u(n) \neq 0 \), we denote by \( \text{fsupp}(u) := \{ m, n \} \) as its filter support. To list the filter \( u \), we shall adopt the following notation throughout the book:

\[
  u = \{ u(m), u(m+1), \ldots, u(-1), u(0), u(1), \ldots, u(n-1), u(n) \}_{m,n},
\]

where we underlined and boldfaced the number \( u(0) \) to indicate its position at the origin.

**Example 1** \( \{ a; b \} \) is an orthogonal wavelet filter bank (called the Haar orthogonal wavelet filter bank), where

\[
  a = \{ \frac{1}{2}, \frac{1}{2} \}_{[0,1]}, \quad b = \{ -\frac{1}{2}, \frac{1}{2} \}_{[0,1]}.
\]

**Example 2** \( \{ \{ \tilde{a}; \tilde{b} \} \} \) is a biorthogonal wavelet filter bank, where

\[
  \tilde{a} = \{ -\frac{1}{8}, \frac{1}{4}, -\frac{1}{2}, -\frac{1}{4} \}_{-2,2}, \quad \tilde{b} = \{ -\frac{1}{4}, \frac{1}{2}, -\frac{1}{4} \}_{[0,2]},
\]

\[
  a = \{ \frac{1}{4}, \frac{1}{4} \}_{[-1,1]}, \quad b = \{ -\frac{1}{4}, \frac{1}{4}, -\frac{1}{4}, -\frac{1}{4} \}_{[-1,3]}.
\]

**Example 3** \( \{ a; b_1, b_2 \} \) is a tight framelet filter bank, where

\[
  a = \{ \frac{1}{4}, \frac{1}{4} \}_{[-1,1]}, \quad b_1 = \{ -\frac{\sqrt{7}}{7}, \frac{\sqrt{7}}{7} \}_{[-1,1]}, \quad b_2 = \{ -\frac{1}{4}, \frac{1}{4} \}_{[-1,1]}.
\]

**Example 4** \( \{ \{ \tilde{a}; \tilde{b}_1, \tilde{b}_2 \} \} \) is a dual framelet filter bank, where

\[
  \tilde{a} = \{ \frac{1}{4}, \frac{1}{2} \}_{[0,1]}, \quad \tilde{b}_1 = \{ -\frac{1}{2}, \frac{1}{2} \}_{[-1,0]}, \quad \tilde{b}_2 = \{ -\frac{1}{2}, \frac{1}{2} \}_{[0,1]}
\]

and

\[
  a = \{ \frac{1}{8}, \frac{3}{8}, \frac{1}{8}, \frac{1}{8} \}_{[-1,2]}, \quad b_1 = \{ -\frac{1}{4}, \frac{3}{4} \}_{[-1,0]}, \quad b_2 = \{ -\frac{1}{8}, \frac{3}{8}, \frac{3}{8}, \frac{1}{8} \}_{[-1,2]}.
\]

At the end of this section, we illustrate a one-level discrete framelet transform using the Haar orthogonal wavelet filter bank in (11). Let

\[
  v = \{ 1, 0, -1, -1, -4, 60, 58, 56 \}_{[0,7]}
\]

be a test input signal. Note that

\[
  [\mathcal{T}_a v](n) = v(2n+1) + v(2n), \quad [\mathcal{T}_b v](n) = v(2n+1) - v(2n), \quad n \in \mathbb{Z}.
\]

Therefore, we have the wavelet coefficients:

\[
  w_0 = \frac{\sqrt{7}}{7} \{ 1, -2, 56, 114 \}_{[0,3]}, \quad w_1 = \frac{\sqrt{2}}{2} \{ -1, 0, 64, -2 \}_{[0,3]}.
\]
On the other hand, we have
\[ [\mathcal{S}_w \hat{w}_0](2n) = \hat{w}_0(n), \quad [\mathcal{S}_w \hat{w}_0](2n + 1) = \hat{w}_0(n), \quad n \in \mathbb{Z} \]
and
\[ [\mathcal{S}_h \hat{w}_1](2n) = -\hat{w}_1(n), \quad [\mathcal{S}_h \hat{w}_1](2n + 1) = \hat{w}_1(n), \quad n \in \mathbb{Z}. \]
Hence, we have
\[ \sqrt{2} \mathcal{S}_w w_0 = \frac{1}{2} \{ 1, 1, -2, 56, 56, 114, 114 \} \|_{[0, \pi]}, \]
\[ \sqrt{2} \mathcal{S}_h w_1 = \frac{1}{2} \{ 1, -1, 0, 0, -64, 64, 2, -2 \} \|_{[0, \pi]}. \]
Clearly, we have the perfect reconstruction of the original input signal \( v \):
\[ \sqrt{2} \mathcal{S}_w w_0 + \sqrt{2} \mathcal{S}_h w_1 = \{ 1, 0, -1, -1, -4, 60, 58, 56 \} \|_{[0, \pi]} = v \]
and the following energy-preserving identity
\[ \| w_0 \|_{l_2(\mathbb{Z})}^2 + \| w_1 \|_{l_2(\mathbb{Z})}^2 = \frac{10119}{2} = \| v \|_{l_2(\mathbb{Z})}^2. \]

Next, let us describe how to efficiently implement discrete framelet/wavelet transform.

The subdivision operator and the transition operator in applications are often implemented through the widely used convolution operation in mathematics and engineering. For \( u \in l_0(\mathbb{Z}) \) and \( v \in l(\mathbb{Z}) \), the convolution \( u * v \) is defined to be
\[ (u * v)(n) := \sum_{k \in \mathbb{Z}} u(k) v(n - k), \quad n \in \mathbb{Z}. \]
(13)
Note that \( \hat{u} \hat{v}(\xi) = \hat{u}(\xi) \hat{v}(\xi) \). To implement the subdivision and transition operators using the convolution operation, we also need the upsampling and downsampling operators on sequences in \( l(\mathbb{Z}) \). The downsampling (or decimation) operator \( \downarrow d : l(\mathbb{Z}) \rightarrow l(\mathbb{Z}) \) and the upsampling operator \( \uparrow d : l(\mathbb{Z}) \rightarrow l(\mathbb{Z}) \) with a sampling factor \( d \in \mathbb{Z} \setminus \{0\} \) are given by
\[ [v \downarrow d](n) := v(dn) \quad \text{and} \quad [v \uparrow d](n) := \begin{cases} v(n/d), & \text{if } n/d \text{ is an integer;} \\ 0, & \text{otherwise,} \end{cases} \]
(14)
for \( n \in \mathbb{Z} \). For a sequence \( v = \{ v(k) \}_{k \in \mathbb{Z}} \), we denote its complex conjugate sequence reflected about the origin by \( v^* \), which is defined to be
\[ v^*(k) := v(-k), \quad k \in \mathbb{Z}. \]
Note that \( \hat{v^*}(\xi) = \overline{\hat{v}(\xi)} \). Now the subdivision operator \( \mathcal{S}_u \) in (2) and the transition operator \( \mathcal{S}_v \) in (3) can be equivalently expressed as follows:
\[ \mathcal{S}_u v = 2(v \uparrow 2) * u \quad \text{and} \quad \mathcal{S}_u v = 2(v * u^*) \downarrow 2. \]
(15)
For \( u = \{ u(k) \}_{k \in \mathbb{Z}} \) and \( \gamma \in \mathbb{Z} \), we define the associated coset sequence \( u[\gamma] \) of \( u \) at the coset \( \gamma + 2\mathbb{Z} \) by
\[
\hat{u}[\gamma](\xi) := \sum_{k \in \mathbb{Z}} u(\gamma + 2k)e^{-ik\xi}, \text{ i.e., } u[\gamma] = u(\gamma + \cdot) \downarrow 2 = \{ u(\gamma + 2k) \}_{k \in \mathbb{Z}}. \quad (16)
\]
Using the coset sequences of \( u \), we can rewrite (15) as
\[
[Su][0] \downarrow 2 = 2v \ast u[0], \quad [Su][1] \downarrow 2 = 2v \ast u[1],
\]
\[
S u v \downarrow 2 = 2 (v[0] \ast (u[0])^\ast + v[1] \ast (u[1])^\ast). \quad (17)
\]

Figure 2: Diagram of a two-level discrete framelet transform employing filter banks \( \{ \tilde{a}; \tilde{b}_1, \ldots, \tilde{b}_s \} \) and \( \{ a; b_1, \ldots, b_s \} \). Note that \( \sqrt{2} \downarrow 2 \tilde{b}_\ell v = \sqrt{2}(v \ast \tilde{b}_\ell^\ast) \downarrow 2 \) and \( \sqrt{2} \downarrow 2 b_\ell v = \sqrt{2}(v \uparrow 2) \ast \tilde{b}_\ell \) for \( \ell = 1, \ldots, s \).