title: ‘TFDWT: Fast Discrete Wavelet Transform TensorFlow Layers’

tags:

- Discrete Wavelet Transform
- Deep Learning
- Python
- Tensorflow
- CUDA

authors:

- name: Kishore K. Tarafdar
  orcid: 0009-0001-4548-7639

- name: Vikram M. Gadre
  orcid: 0000-0001-8439-5625
  corresponding: true

affiliations:

- name: Department of Electrical Engineering, Indian Institute of Technology Bombay, India

date: 1 April 2025

bibliography: paper.bib

1 Summary

TFDWT is an open-source Python package that allows the creation of TensorFlow Layers for Fast Discrete Wavelet Transform (DWT) and Inverse Discrete Wavelet Transform (IDWT) in end-to-end backpropagation learning networks. These layers facilitate the construction of multilevel DWT filterbanks and Wavelet Packet Transform (WPT) filterbanks for a spatial-frequency representation of the inputs and features in shallow or deep networks. A multiresolution signal representation using a multi-rate discrete wavelet system creates enriched joint natural-frequency representations. The discrete wavelet system partitions the frequency plane into subbands using orthogonal dilated and translated lowpass scaling and highpass wavelet function. A realization of a fast discrete wavelet system is a two-band perfect reconstruction multi-rate filter bank with FIR filters corresponding to the impulse responses of the scaling and wavelet function with downsampling and upsampling operations. A filter bank for a higher dimensional input is a seamless extension by successive separable circular convolutions across each independent axis.

2 Statement of need

In machine or deep learning, an efficient multiresolution representation of data often helps to build economical and explainable models. The Wavelet toolbox [@misiti1996wavelet] by MathWorks is a proprietary software that has served the requirements for D-dimensional wavelet transforms in the MATLAB environment for a few decades. Several open-source packages are now available for 1D and 2D DWT in Python. Pywavelets [@lee2019pywavelets] is a D-dimensional wavelet transform library in Python that works with Numpy [@harris2020array] arrays. However, it is challenging to directly use Pywavelets with the symbolic tensors in TensorFlow [@developers2022tensorflow] layers and CUDA [@fatica2008cuda]. WaveTF [@versaci2021wavetf] is a solution for constructing 1D and 2D DWT layers in TensorFlow but is limited to only Haar and Db2 wavelets. The package tensorflow-wavelets [@tensorflow-wavelets-1.1.2] supports multiple wavelets, but it has a minor bug in perfect reconstruction due to the padding and boundary effects in processing the finite-length inputs. In Pytorch [@imambi2021pytorch], the pytorch-wavelets [@pytorch-wavelets-1.3.0] package allows the construction of 1D and 2D DWT layers.

For a D-dimensional wavelet $\boldsymbol{\psi} \in L^2(\mathbb{R})^{\mathrm{D}}$, a discrete wavelet system defined by $\left\{\boldsymbol{\psi}_{m,\boldsymbol{p}} : m \in \mathbb{Z}, \boldsymbol{p} \in \mathbb{Z}^{\mathrm{D}}\right\}$ forms an orthonormal basis in $\boldsymbol{\psi} \in L^2(\mathbb{R})^{\mathrm{D}}$, where $\boldsymbol{\psi}_{m,\boldsymbol{p}}(\boldsymbol{x}) := 2^m \psi(2^m \boldsymbol{x} - \boldsymbol{p})$. Then, by definition the DWT of $\boldsymbol{x} \in \mathbb{Z}^{\mathrm{D}}$ is $\boldsymbol{x} \mapsto (\langle \boldsymbol{x}, \psi_{m,\boldsymbol{p}} \rangle)_{m,\boldsymbol{p}}$, where m is the dilation parameter and $\boldsymbol{p}$ is the shift or translation parameter. This new TFDWT Python package is a simple standalone DWT and IDWT library with minimal dependencies that allow computation with symbolic tensors and CUDA. The package can also be seamlessly upgraded to separable higher dimensional DWT. The boundary effects are taken care of with cyclic convolutions instead of padding. The package supports orthogonal and biorthogonal wavelets of different families having impulse responses of diverse lengths. In this paper, we defined the platform-independent, underlying mathematics for realizing fast DWT and IDWT layers with filter bank structures having FIR filters, downsamplers and upsamplers. Although our realization of the discrete wavelet system is in TensorFlow 2, a seamless reproduction of the computations is possible in other deep learning frameworks.

3 Discrete wavelet system for sequences

A discrete wavelet system has a lowpass scaling function and a highpass wavelet. The realization of a discrete wavelet system with a continuous one-dimensional (D=1) mother wavelet $\psi \in L^2(\mathbb{R})$ is by using the impulse responses of the scaling function and the wavelet as Finite Impulse Response (FIR) filters $g\big[n\big]$ and $h\big[n\big]$. Figure [fig:wavelets] shows some orthogonal and biorthogonal wavelets and their scaling functions. Figure [fig:waveletsimpulse] shows their corresponding impulse responses. These FIR filters are the building blocks of a two-band perfect reconstruction filter bank for realizing fast discrete wavelet systems.

Figure [fig:2BPRFB] shows a two-band perfect reconstruction filter bank that operates on one-dimensional inputs, i.e., sequences in $l^2(\mathbb{Z})$. The analysis and synthesis bank in orthogonal wavelet systems have identical lowpass and highpass FIR filters but differ in biorthogonal wavelet systems. In biorthogonal wavelet filter banks, $\tilde{g}\left[n\right]$ and $\tilde{h}\left[n\right]$ are the lowpass and highpass filters of the synthesis bank. The only difference in the biorthogonal families like bior and rbio is the interchange of the analysis and synthesis scaling and wavelets functions, for example, in bior3.1 and rbio3.1.

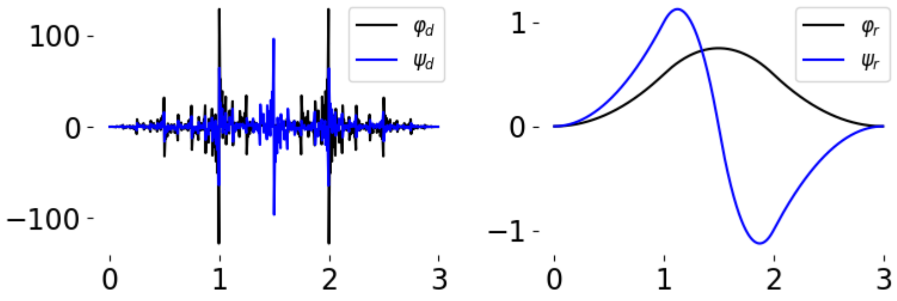


Figure 1: Wavelets and scaling functions of bior3.1 analysis (left) and synthesis (right).

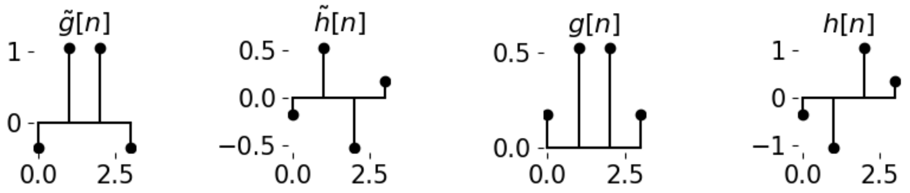


Figure 2: Impulse responses of different bior3.1 analysis (left two) and synthesis (right two) lowpass and highpass FIR filters.

3.1 Circular convolution operators

The four matrices in the two band perfect reconstruction filter bank in Figure \autoref{fig:2BPRFB} are — (i) $\boldsymbol{G}$ is lowpass analysis matrix, (ii) $\boldsymbol{H}$ is highpass analysis matrix, (iii) $\tilde{\boldsymbol{G}}$ is lowpass synthesis matrix and (iv) $\tilde{\boldsymbol{H}}$ is highpass synthesis matrix. These matrices are operators for circular convolution, constructed by circular shifts of the corresponding FIR filters $g\left[n-k\right]$, $h\left[n-k\right]$, $\tilde{g}\left[n-k\right]$ and $\tilde{h}\left[n-k\right]$, where $g = \tilde{g}$ and $h = \tilde{h}$ for orthogonal wavelets.

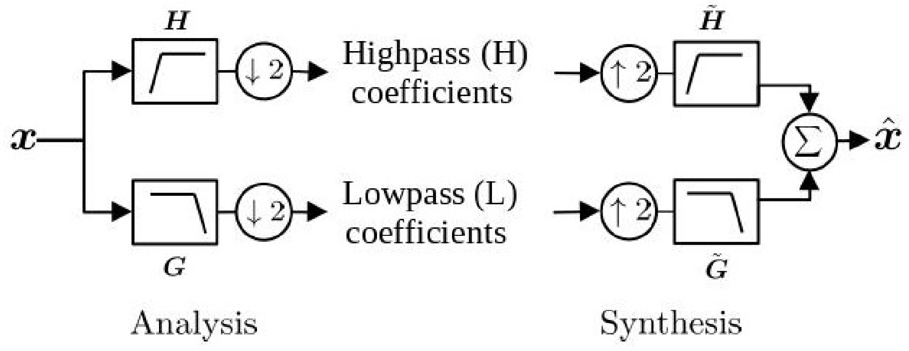


Figure 3: Two band perfect reconstruction filter bank.

3.1.1 Analysis matrix

For a lowpass analysis FIR filter of length L and a input sequence of length N, the circular convolution operator is a matrix $\boldsymbol{G}$ of shape $N \times N$. A downsampling by two $f_{(\downarrow 2)}$ of the output the convolution is equivalent to getting rid of the even rows of $\boldsymbol{G}$ to give $\frac{N}{2} \times N$ operator $f_{(\downarrow 2)}\boldsymbol{G}$. Similarly, for the highpass analysis FIR filter of same wavelet with convolution and downsampling is a $\frac{N}{2} \times N$ operator $f_{(\downarrow 2)}\boldsymbol{H}$. The analysis matrix $\boldsymbol{A}$ is a $N \times N$ decimated circular convolution operator formed by combining the lowpass and highpass decimated operators as given by equation [eq:AnalysisMatrix].

$$\boldsymbol{A} = \begin{bmatrix} f_{(\downarrow 2)}\boldsymbol{G} \\ f_{(\downarrow 2)}\boldsymbol{H} \end{bmatrix}_{N \times N}$$

3.1.2 Synthesis matrix

The synthesis matrix $\boldsymbol{S}$ is another $N \times N$ decimated circular convolution operator as given by equation [eq:SynthesisMatrix],

$$\boldsymbol{S} = \begin{bmatrix} f_{(\downarrow 2)}\tilde{\boldsymbol{G}} \\ f_{(\downarrow 2)}\tilde{\boldsymbol{H}} \end{bmatrix}^{T}_{N \times N}$$

where $\tilde{\boldsymbol{G}}$ and $\tilde{\boldsymbol{H}}$ are matrices formed by the lowpass and highpass synthesis FIR filters. Equation [eq:SynthesisMatrix] is a general representation for both orthogonal and biorthogonal wavelets families where for orthogonal wavelets, $\tilde{\boldsymbol{G}} = \boldsymbol{G}$, $\tilde{\boldsymbol{H}} = \boldsymbol{H}$ and thus $\boldsymbol{S} = \boldsymbol{A}^T$.

A two-band perfect reconstruction discrete wavelet system for one-dimensional inputs is given by the analysis equation [eq:DWToneSequence] and the synthesis equation [eq:IDWToneSequence],

$$\boldsymbol{q} = \text{DWT}\big(\boldsymbol{x}\big) \text{ or, } \boldsymbol{q} = \big(\boldsymbol{A}\boldsymbol{x}^T\big)^T$$

$$\boldsymbol{x} = \text{IDWT}\big(\boldsymbol{q}\big) \text{ or, } \boldsymbol{x} = \big(\boldsymbol{S}\boldsymbol{q}^T\big)^T$$

where \boldsymbol{A} and \boldsymbol{S} are analysis and synthesis matrices as defined in equations [eq:AnalysisMatrix] and [eq:SynthesisMatrix], \boldsymbol{x} is a input sequence and \boldsymbol{q} has a distinct lowpass and a highpass subband.

Example 1. Given, a sequence $\boldsymbol{x} \in \mathbb{R}^8$ or $N = 8$ and FIR filters length $L = 6$.

$$\text{LPF \& downsampling } f_{(\downarrow 2)}\boldsymbol{G} = \begin{bmatrix} g_1 & g_0 & 0 & 0 & g_5 & g_4 & g_3 & g_2 \\ g_3 & g_2 & g_1 & g_0 & 0 & 0 & g_5 & g_4 \\ g_5 & g_4 & g_3 & g_2 & g_1 & g_0 & 0 & 0 \\ 0 & 0 & g_5 & g_4 & g_3 & g_2 & g_1 & g_0 \end{bmatrix}_{\frac{N}{2}\times N}$$

$$\text{HPF \& downsampling } f_{(\downarrow 2)}\boldsymbol{H} = \begin{bmatrix} h_1 & h_0 & 0 & 0 & h_5 & h_4 & h_3 & h_2 \\ h_3 & h_2 & h_1 & h_0 & 0 & 0 & h_5 & h_4 \\ h_5 & h_4 & h_3 & h_2 & h_1 & h_0 & 0 & 0 \\ 0 & 0 & h_5 & h_4 & h_3 & h_2 & h_1 & h_0 \end{bmatrix}_{\frac{N}{2}\times N}$$

$$\text{Analysis matrix is } \boldsymbol{A} = \begin{bmatrix} g_1 & g_0 & 0 & 0 & g_5 & g_4 & g_3 & g_2 \\ g_3 & g_2 & g_1 & g_0 & 0 & 0 & g_5 & g_4 \\ g_5 & g_4 & g_3 & g_2 & g_1 & g_0 & 0 & 0 \\ 0 & 0 & g_5 & g_4 & g_3 & g_2 & g_1 & g_0 \\ h_1 & h_0 & 0 & 0 & h_5 & h_4 & h_3 & h_2 \\ h_3 & h_2 & h_1 & h_0 & 0 & 0 & h_5 & h_4 \\ h_5 & h_4 & h_3 & h_2 & h_1 & h_0 & 0 & 0 \\ 0 & 0 & h_5 & h_4 & h_3 & h_2 & h_1 & h_0 \end{bmatrix}_{N\times N}$$

Similarly,

$$\text{Synthesis matrix is } \boldsymbol{S} = \begin{bmatrix} \tilde{g}_1 & \tilde{g}_0 & 0 & 0 & \tilde{g}_5 & \tilde{g}_4 & \tilde{g}_3 & \tilde{g}_2 \\ \tilde{g}_3 & \tilde{g}_2 & \tilde{g}_1 & \tilde{g}_0 & 0 & 0 & \tilde{g}_5 & \tilde{g}_4 \\ \tilde{g}_5 & \tilde{g}_4 & \tilde{g}_3 & \tilde{g}_2 & \tilde{g}_1 & \tilde{g}_0 & 0 & 0 \\ 0 & 0 & \tilde{g}_5 & \tilde{g}_4 & \tilde{g}_3 & \tilde{g}_2 & \tilde{g}_1 & \tilde{g}_0 \\ \tilde{h}_1 & \tilde{h}_0 & 0 & 0 & \tilde{h}_5 & \tilde{h}_4 & \tilde{h}_3 & \tilde{h}_2 \\ \tilde{h}_3 & \tilde{h}_2 & \tilde{h}_1 & \tilde{h}_0 & 0 & 0 & \tilde{h}_5 & \tilde{h}_4 \\ \tilde{h}_5 & \tilde{h}_4 & \tilde{h}_3 & \tilde{h}_2 & \tilde{h}_1 & \tilde{h}_0 & 0 & 0 \\ 0 & 0 & \tilde{h}_5 & \tilde{h}_4 & \tilde{h}_3 & \tilde{h}_2 & \tilde{h}_1 & \tilde{h}_0 \end{bmatrix}_{N\times N}$$

The DWT of $\boldsymbol{x}$ produces subbbands,

$$\boldsymbol{q} = \text{DWT}\big(\boldsymbol{x}\big) \text{ or, } \boldsymbol{q = Ax}$$

Perfect reconstruction,

$$\boldsymbol{x} = \text{IDWT}\big(\boldsymbol{q}\big) \text{ or, } \boldsymbol{x} = \boldsymbol{S}\boldsymbol{q}$$

3.2 DWT 1D layer

A DWT 1D layer takes batched, multichannel sequences of shape (batch, length, channels) as input and computes subbands of each sequence using equation [eq:DWToneSequence]. The shape of the DWT 1D layer output after grouping the lowpass and highpass subbands of each sequence is $(\text{batch}, \text{length}/2, 2 \times \text{channels})$.

3.3 IDWT 1D layer

An IDWT 1D layer takes batched, multichannel subbands of shape $(\text{batch}, \text{length}/2, 2 \times \text{channels})$. as input and reconstructs each batched, multichannel sequence using equation [eq:IDWToneSequence]. The shape of the IDWT 1D layer output is (batch, length, channels).

4 Higher dimensional discrete wavelet systems

In sequences (1D), the DWT 1D applies to the only independent variable. To achieve high-dimensional DWT, the DWT 1D needs to be applied separably to all the independent variables one after the other. For example, the DWT of an image (2D) is a row-wise DWT 1D followed by a column-wise DWT 1D. Similarly, the reconstruction is column-wise IDWT 1D followed by a row-wise IDWT 1D.

4.1 Two-dimensional discrete wavelet system

The pixel values in an image are a function of two independent spatial axes. A DWT 2D filter bank is a separable transform with row-wise filtering followed by column-wise filtering that yields four subbands - LL, LH, HL and HH. A two-dimensional discrete wavelet system is given by,

$$\boldsymbol{q} = \text{DWT}\big(\boldsymbol{x}\big) := \boldsymbol{A}(\boldsymbol{A}\boldsymbol{x}_{021})_{021}^T$$

$$\boldsymbol{x} = \text{IDWT}\big(\boldsymbol{q}\big) := \boldsymbol{S}(\boldsymbol{S}\left[\boldsymbol{A}(\boldsymbol{A}\boldsymbol{x}_{021}^T)_{021}^T\right]_{021}^T)_{021}^T$$

where, $\boldsymbol{A}$ and $\boldsymbol{S}$ are analysis and synthesis matrices as defined in equations [eq:AnalysisMatrix] and [eq:SynthesisMatrix]. Figure [fig:subdband2d] shows an $N \times N$ image and its four localized spatial-frequency subbands after DWT. Here, the low-frequency band is LL, and the other three are high-frequency subbands representing horizontal, vertical and diagonal features. Figure [fig:2BPRFB-2] illustrates a separable 2D DWT perfect reconstruction filter bank. The 2D layers operate on batched, multichannel tensors of shape (batch, height, width, channels), where each image is of shape height and width. Figure [fig:2DDWTlayer] illustrates input, output and perfect reconstruction by DWT 2D and IDWT 2D layers. The Multiresolution Encoder-Decoder Convolutional Neural Network in [@tarafdar2025multiresolution] uses these layers.

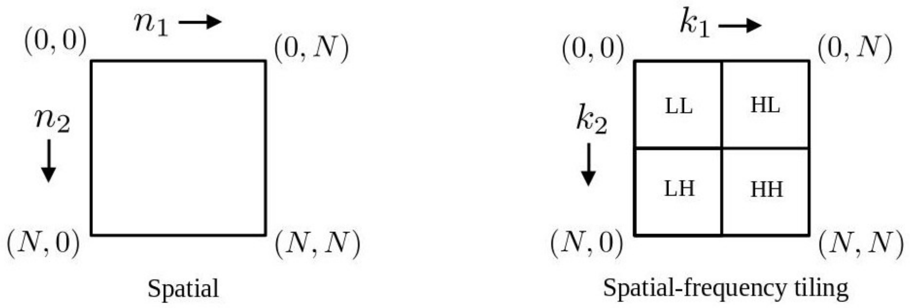


Figure 4: An image of shape (left) and spatial-frequency tiled subbands (right) after a level-1 DWT 2D decomposition.

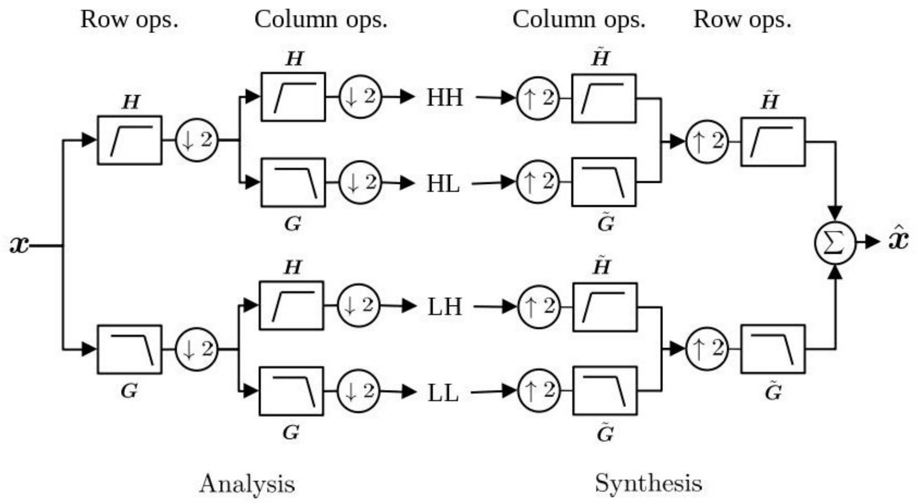


Figure 5: Separable DWT 2D perfect reconstruction filter bank.

4.1.1 DWT 2D layer

A DWT 2D layer operates on input tensors of shape (batch, height, width, channels) and produces an output of shape $(\text{batch}, \text{height}/2, \text{width}/2, 4 \times \text{channels})$ as described in Algorithm [algo:DWT2D].

1. Input $\boldsymbol{X}$ of shape
   (batch, height, width, channels).
2. Generate analysis matrix $\boldsymbol{A}$ using height
   and width of input.
3. For each batched channel $\boldsymbol{x}_c \in \boldsymbol{X}$ of
   shape (batch, height, width):
   (omitting suffix c in $\boldsymbol{x}$ below for simplicity of
   notation)
   1. Batched row wise DWT
      $\boldsymbol{A}\boldsymbol{x}_{021}^T := \text{Einsum}\big(ij, bjk \to bik\big)$
   2. Batched column wise DWT,
      $\boldsymbol{A}(\boldsymbol{A}\boldsymbol{x}_{021}^T)_{021}^T := \text{Einsum}\big(ij, bjk \to bik\big)$

      or, equivalently, DWT of a batched channel $\boldsymbol{x}$ is,

      $\boldsymbol{q}_c = \text{DWT}\big(\boldsymbol{x}\big)$ or, $\boldsymbol{q}_c = \boldsymbol{A}(\boldsymbol{A}\boldsymbol{x}_{021})_{021}^T$
4. Stacking for all c channels,
   $\boldsymbol{Q} := \big(\boldsymbol{q}_c\big)_{\forall c}$ to a
   shape (batch, height, width, channels).
5. Group subbands and return an output
   $\boldsymbol{Q}^{(\text{grouped})}$ of shape
   $(\text{batch}, \text{height}/2, \text{width}/2, 4 \times \text{channels})$

[]{#algo:DWT2D label="algo:DWT2D"}

4.1.2 IDWT 2D layer

An IDWT 3D layer operates on input tensors of shape $(\text{batch}, \text{height}/2, \text{width}/2, 4\times$ channels) and produces an output of shape (batch, height, width, channels) as described in [Algo:IDWT2DLayer].

1. Input $\boldsymbol{Q}^{(\text{grouped})}$ of shape
   $(\text{batch}, \text{height}/2, \text{width}/2, 4 \times \text{channels})$
2. Ungroup the subbands to get $\boldsymbol{Q}$ of shape
   (batch, height, width, channels)
3. Generate synthesis matrix $\boldsymbol{S}$ using height
   and width of $\boldsymbol{Q}$.

4. For each batched channel $\boldsymbol{q}_c \in \boldsymbol{Q}$ of shape (batch, height, width): (omitting suffix c in $\boldsymbol{q}$ below for simplicity of notation)

    1. Batched row wise IDWT,

       $\boldsymbol{S}\boldsymbol{q}^T_{021} := \mathrm{Einsum}\big(ij, bjk \to bik\big)$

    2. Batched column wise IDWT,

       $\boldsymbol{S}(\boldsymbol{S}\boldsymbol{q}^T_{021})^T_{021} := \mathrm{Einsum}\big(ij, bjk \to bik\big)$

       or equivalently, a perfect reconstruction,

       $\boldsymbol{x} = \mathrm{IDWT}\big(\boldsymbol{q}\big)$ or, $\boldsymbol{x} = \boldsymbol{S}(\boldsymbol{S}\boldsymbol{q}^T_{021})^T_{021}$

       where, $\boldsymbol{S} = \boldsymbol{A}^T$ for orthogonal wavelets

5. Layer output (perfect reconstruction) $\boldsymbol{X} := \big(\boldsymbol{x}_c\big)_{\forall c}$ is of shape (batch, height, width, channels)

[]{#Algo:IDWT2DLayer label="Algo:IDWT2DLayer"}

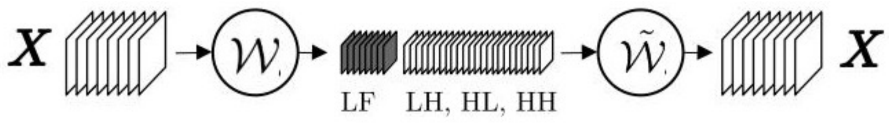


Figure 6: DWT decomposition and perfect reconstruction of a tensor , where is DWT 2D layer and is IDWT 2D layer.

4.2 Three-dimensional discrete wavelet system

A three-dimensional (3D) discrete wavelet system for a 3D input $\boldsymbol{x}$ is given by,

$$\boldsymbol{q} = \mathrm{DWT}\big(\boldsymbol{x}\big) := \big[\boldsymbol{A}(\boldsymbol{A}(\boldsymbol{A}\boldsymbol{x}^T_{0213})^T_{0213})^T_{0132}\big]^{\,T}_{\,0132}$$

$$\boldsymbol{x} = \mathrm{IDWT}\big(\boldsymbol{q}\big) := (\boldsymbol{S}(\boldsymbol{S}(\boldsymbol{S}\,\big[\boldsymbol{A}(\boldsymbol{A}(\boldsymbol{A}\boldsymbol{x}^T_{0213})^T_{0213})^T_{0132}\big]^{\,T}_{\,0132}{}^{T}_{0132})^T_{0132})^T_{0213})_{0213}$$

where, $\boldsymbol{A}$ and $\boldsymbol{S}$ are analysis and synthesis matrices as defined in equations [eq:AnalysisMatrix] and [eq:SynthesisMatrix]. The DWT 3D and IDWT 3D layers operate on batched, multichannel tensors of shape $\big(\text{batch, height, width, depth, channels}\big)$, where each cube is transformed by equations [eq:Analysis3D] and [eq:Synthesis3D].

4.2.1 DWT 3D layer

A DWT 3D layer operates on input tensors of shape (batch, height, width, depth, channels) and produces an output of shape $(\text{batch}, \text{height}/2, \text{width}/2, \text{depth}/2, 8 \times \text{channels})$
as described in Algorithm [Algo:DWT3Dlayer].

1. Input $\boldsymbol{X}$ of shape
   (batch, height, width, depth, channels).
2. Generate analysis matrix $\boldsymbol{A}$ using height,
   width and depthof input.
3. For each batched channel $\boldsymbol{x}_c \in \boldsymbol{X}$ of
   shape (batch, height, width, depth):
   (omitting suffix c in $\boldsymbol{x}$ below for simplicity of
   notation)
   1. Row-wise operations,

      $\boldsymbol{A}\boldsymbol{x}_{0213}^{T} := \text{Einsum}\big(ij, bjkl \rightarrow bikl\big)$
   2. Column-wise operations,

      $\boldsymbol{A}\left(\boldsymbol{A}\boldsymbol{x}_{0213}^{T}\right)_{0213}^{T} := \text{Einsum}\big(ij, bjkl \rightarrow bikl\big)$
   3. Depth-wise operations,

      $\boldsymbol{A}(\boldsymbol{A}(\boldsymbol{A}\boldsymbol{x}_{0213}^{T})_{0213}^{T})_{0132}^{T} := \text{Einsum}\big(ik, bjkl \rightarrow bjil\big)$

      Therefore the forward transform of $\boldsymbol{x}$ yield DWT analysis coefficients

      $\boldsymbol{q}_c := \text{DWT}\,(\boldsymbol{x}) = \big[\boldsymbol{A}(\boldsymbol{A}(\boldsymbol{A}\boldsymbol{x}_{0213}^{T})_{0213}^{T})_{0132}^{T}\big]_{0132}^{T}$
4. Stacking for all c channels,
   $\boldsymbol{Q} := \big(\boldsymbol{q}_c\big)_{\forall c}$ to a
   shape (batch, height, width, depth, channels).
5. Group subbands and return an output
   $\boldsymbol{Q}^{(\text{grouped})}$ of shape
   $(\text{batch}, \text{height}/2, \text{width}/2, \text{depth}/2, 8 \times \text{channels})$.

[]{#Algo:DWT3Dlayer label="Algo:DWT3Dlayer"}

4.2.2 IDWT 3D layer

An IDWT 3D layer operates on input tensors of shape $(\text{batch}, \text{height}/2, \text{width}/2, \text{depth}/2, 8 \times$ channels) and produces an output of shape (batch, height, width, depth, channels)
as described in Algorithm [Algo:IDWT3Dlayer].

---

1. Input $\boldsymbol{Q}^{(\text{grouped})}$ of shape
   $(\text{batch}, \text{height}/2, \text{width}/2, \text{depth}/2, 8 \times \text{channels})$.$

2. Ungroup to get $\boldsymbol{Q}$ of shape (batch, height, width, depth, channels)

3. Generate synthesis matrix $\boldsymbol{S}$ using height, width and depth of $\boldsymbol{Q}$.

4. For each batched channel $\boldsymbol{q}_c \in \boldsymbol{Q}$ of shape (batch, height, width, depth): (omitting suffix c in $\boldsymbol{q}$ below for simplicity of notation)

    1. Row-wise operations, $\boldsymbol{S}\boldsymbol{q}^T_{0132} := \text{Einsum}\big(ik, bjkl \to bjil\big)$

    2. Column-wise operations,

       $\boldsymbol{S}(\boldsymbol{S}\boldsymbol{q}^T_{0132})_{0132} := \text{Einsum}\big(ij, bjkl \to bikl\big)$

    3. Depth-wise operations,

       $\boldsymbol{S}(\boldsymbol{S}(\boldsymbol{S}\boldsymbol{q}^T_{0132})^T_{0132})^T_{0213} := \text{Einsum}\big(ij, bjkl \to bikl\big)$

       or equivalently, a perfect reconstruction,

       $\boldsymbol{x}_c := \text{IDWT}\,(\boldsymbol{q}) = \big[\boldsymbol{S}(\boldsymbol{S}(\boldsymbol{S}\boldsymbol{q}^T_{0132})^T_{0132})^T_{0213}\big]^T_{0213}$

       where, $\boldsymbol{S} = \boldsymbol{A}^T$ for orthogonal wavelets.

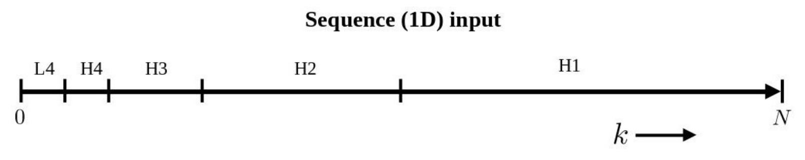
5. Layer output (perfect reconstruction) $\boldsymbol{X} := \big(\boldsymbol{x}_c\big)_{\forall c}$ is of shape (batch, height, width, depth, channels)

[]{#Algo:IDWT3Dlayer label="Algo:IDWT3Dlayer"}

In general, a seamless realization of fast D-dimensional DWT and IDWT is possible by extending the above separable method to all the independent N axes one after the other. The number of subbands will be equal to 2^{D} for a D dimensional DWT. For example, sequences ($\text{D} = 1$) yield two subbands, images ($\text{D} = 2$) yield four subbands, three-dimensional inputs ($\text{D} = 3$) with voxels yield eight subbands etc.

5 Multilevel wavelet filter banks

The above-discussed DWT and IDWT layers are building blocks in constructing multilevel DWT filter banks. Figure [fig:MultilevelDWTtiling] shows the partitioning of the 1D frequency axis and tiling of the 2D frequency plane using a level-4 1D and 2D DWT. The multilevel DWT successively decomposes the low-frequency feature. If the high-frequency features are also decomposed successively, then we get a Wavelet Packet Transform (WPT) filter bank.



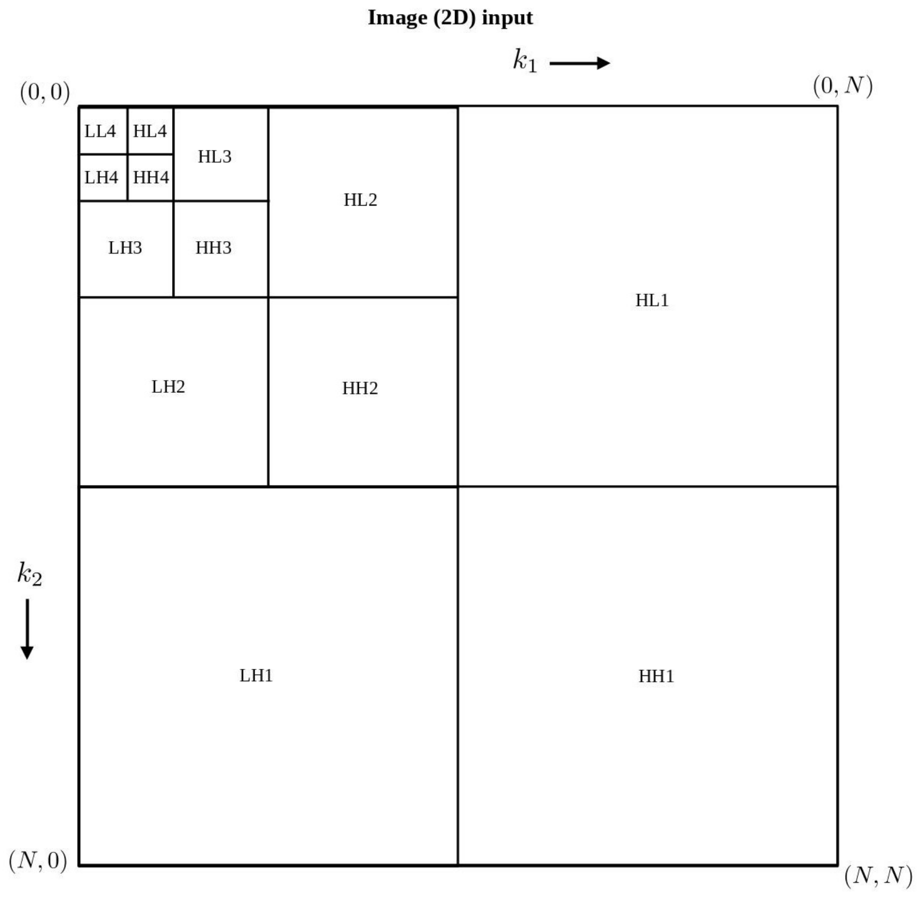


Figure 7: Spatial-frequency tiling by DWT level 4 decomposition of a sequence of length N (top) and image of shape Nimes N (bottom).

References