

# Wavelet Filter Transforms in Detail

Wei ZHU and M. Victor WICKERHAUSER  
Washington University in St. Louis, Missouri

`victor@math.wustl.edu`

`http://www.math.wustl.edu/~victor`

FFT 2008

The Norbert Wiener Center

February 22nd, 2008

## Main Ideas

Goal: implement discrete wavelet transforms efficiently

- **Lifting:** reduce the number of arithmetic operations.
- **Nearest-neighbors:** reduce the number of memory accesses.
- **Symmetry:** simplify the implementation.
- **Choice:** maximize some utility.

Starting point: Sweldens' and Daubechies' lifting factorization.

## Discrete Wavelet Transforms

**Signal:**  $u \in \ell^2$ , in practice finitely supported or periodic.

**Analysis filters:** linear maps  $\tilde{H}, \tilde{G} : \ell^2 \rightarrow \ell^2$ ; convolution and subsampling.

**Discrete wavelet transform:**

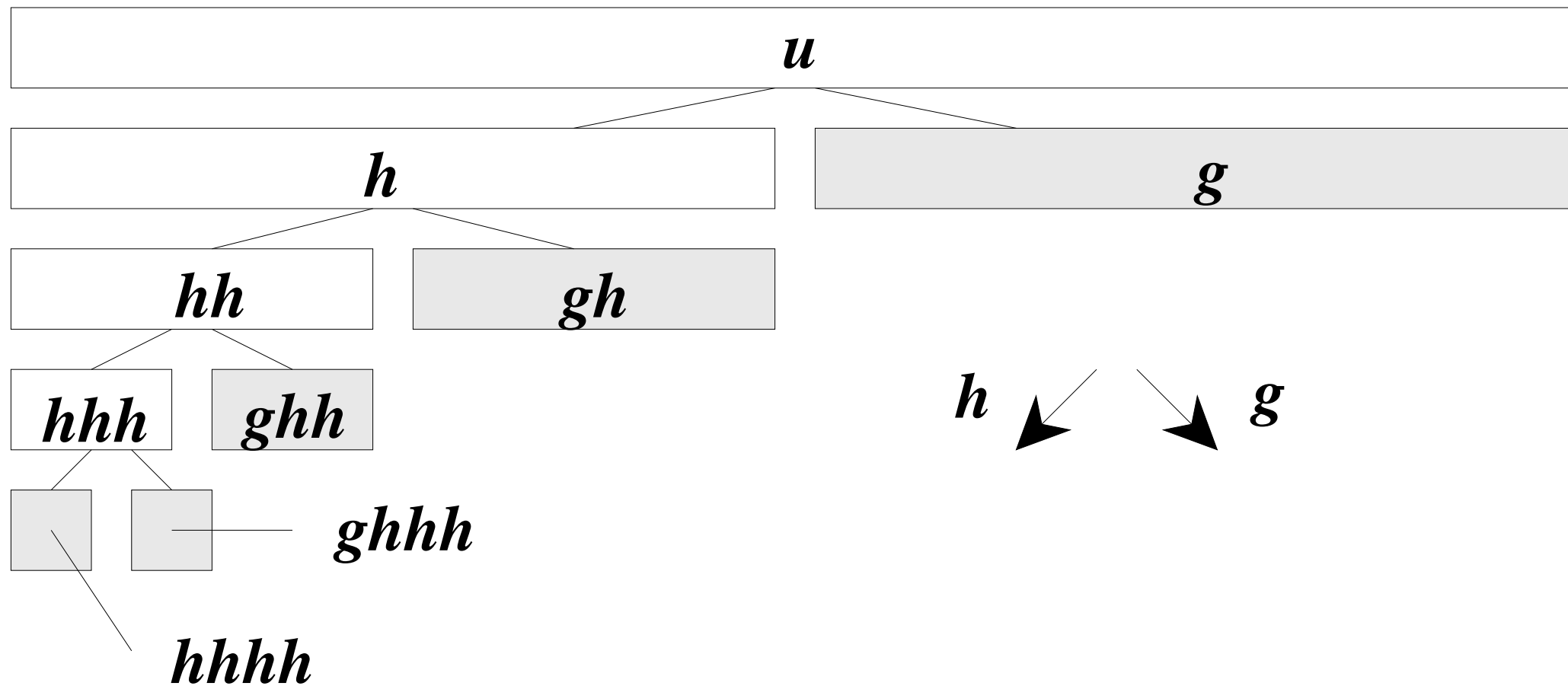
$$u \mapsto \{\tilde{H}^J u; \tilde{G}\tilde{H}^{J-1}u, \tilde{G}\tilde{H}^{J-2}u, \dots, \tilde{G}\tilde{H}u, \tilde{G}u\}.$$

**Synthesis filters:** linear maps  $H, G : \ell^2 \rightarrow \ell^2$ , related to  $\tilde{H}, \tilde{G}$ .

**Wavelet reconstruction:**

$$\begin{aligned} u &= G^* \tilde{G}u + H^* \tilde{H}u \\ &= G^* \tilde{G}u + H^* (G^* \tilde{G}\tilde{H}u + H^* \tilde{H}^2u) \\ &= \dots \\ &= G^* \tilde{G}u + H^* (G^* \tilde{G}\tilde{H}u + H^* (\dots + H^* (G^* \tilde{G}\tilde{H}^{J-1}u + H^* \tilde{H}^J u) \dots)). \end{aligned}$$

## Discrete Wavelet Transforms II



Use adjoints  $h^*, g^*$  to reconstruct  $u$ , moving up and adding.

## Filters, Adjoints, and Conjugates

A (*finite*) *filter*  $F : \ell^2 \rightarrow \ell^2$  is a linear transformation determined by a (finitely-supported) absolutely summable sequence  $f = \{f_n : n \in \mathbf{Z}\}$ :

$$Fx_m = \sum_n f_{2m-n} x_n, \quad m \in \mathbf{Z}.$$

The *adjoint filter*  $F^*$  determined by the same sequence  $f$  is

$$F^*x_n = \sum_m \bar{f}_{2m-n} x_m, \quad n \in \mathbf{Z}.$$

Thus  $\langle Fx, y \rangle = \langle x, F^*y \rangle$  for all  $x, y \in \ell^2$ , using the inner product in  $\ell^2$ .

The *conjugate filter*  $\dot{F}$  of  $F$  has sequence  $\dot{f} = \{\dot{f}_n : n \in \mathbf{Z}\}$  defined by

$$\dot{f}_n = (-1)^n f_{1-n} \quad \Rightarrow \quad f_n = (-1)^{1-n} \dot{f}_{1-n}, \quad \Rightarrow \quad \dot{\dot{F}} = -F$$

If  $F$  is finite then  $\dot{F}$  is also finite, with the same support length.

## Orthogonality, Biorthogonality, and Perfect Reconstruction

Filter  $H$  is called *orthogonal* if it and its conjugate filter  $G = \dot{H}$  satisfy

$$HH^* = Id; \quad GG^* = Id; \quad GH^* = HG^* = 0; \quad H^*H + G^*G = Id.$$

Filters  $H, G$  form a *perfect reconstruction pair* if they and their conjugates  $\tilde{H} = \dot{G}$  and  $\tilde{G} = \dot{H}$  satisfy

$$\tilde{H}H^* = Id; \quad \tilde{G}G^* = Id; \quad \tilde{G}H^* + \tilde{H}G^* = 0; \quad H^*\tilde{H} + G^*\tilde{G} = Id.$$

These are also called *biorthogonality conditions*. Since  $\dot{\dot{H}} = -H$  and  $\dot{\dot{G}} = -G$ ,

$$H\tilde{H}^* = Id; \quad G\tilde{G}^* = Id; \quad G\tilde{H}^* + H\tilde{G}^* = 0; \quad \tilde{H}^*H + \tilde{G}^*G = Id.$$

Thus  $(\tilde{H}, \tilde{G}) = (\dot{G}, \dot{H})$  form a perfect reconstruction pair as well.

EG: If  $H$  is an orthogonal filter, then  $(H, \dot{H})$  is a perfect reconstruction pair.

Call  $H$  a *perfect reconstruction filter* if there is a filter  $G$  such that  $(H, G)$  is a perfect reconstruction filter pair.

## Perfect Reconstruction Conditions for Sequences

Suppose  $H \leftrightarrow h$  has conjugate  $G \leftrightarrow g$ .

Orthogonality:

$$\sum_k h(k)\bar{h}(k+2n) = \mathbf{1}(n) = \sum_k g(k)\bar{g}(k+2n);$$

$$\sum_k g(k)\bar{h}(k+2n) = 0 = \sum_k h(k)\bar{g}(k+2n);$$

and

$$\sum_k h(2k+m)\bar{h}(2k+n) + \sum_k g(2k+m)\bar{g}(2k+n) = \mathbf{1}(n-m).$$

Here

$$\mathbf{1}(n) = \begin{cases} 1, & \text{if } n = 0, \\ 0, & \text{otherwise.} \end{cases} \quad n, m \in \mathbf{Z}.$$

**Exercise:** find the conditions for biorthogonal perfect reconstruction pairs.

## Z Transforms

$Z$  transform of a sequence  $x = \{x_n \in \mathbf{C} : n \in \mathbf{Z}\} \in \ell^2$ :

$$x(z) = \sum_n x_n z^{-n}, \quad \text{with} \quad \begin{array}{l} \text{even part } x_e(z) \stackrel{\text{def}}{=} \sum_n x_{2n} z^{-n}; \\ \text{odd part } x_o(z) \stackrel{\text{def}}{=} \sum_n x_{2n+1} z^{-n}. \end{array}$$

Recover the  $Z$ -transform of  $x$  from the even and odd parts  $x_e, x_o$ :

$$x(z) = x_e(z^2) + z^{-1}x_o(z^2).$$

Get the even and odd parts from the  $Z$ -transform:

$$x_e(z^2) = \frac{x(z) + x(-z)}{2}, \quad x_o(z^2) = \frac{x(z) - x(-z)}{2z^{-1}}.$$

## Laurent Polynomials

If  $p \in \ell^2$  is finitely-supported, then its  $Z$  transform  $p(z)$  is a *Laurent polynomial*:

$$p(z) = \sum_{n=a}^b p_n z^{-n}, \quad a \leq b, \quad a, b \in \mathbf{Z}.$$

If  $p \neq 0$ , define its *degree* by  $\deg p = b - a$ .

Laurent polynomials form the commutative ring  $\mathbf{C}[z, z^{-1}]$  with multiplicative identity 1.

Element  $p \neq 0$  is called a *unit*

$\iff p$  has a multiplicative inverse

$\iff p$  is a *monomial*  $p(z) = Kz^n$

$\iff \deg p = 0$ .

Then  $p^{-1}(z) = K^{-1}z^{-n}$ .

## Conjugates and $Z$ Transforms

If  $\dot{F}$  is the conjugate of filter  $F$ , then

$$\begin{aligned} \dot{f}(z) &= -z^{-1}f(-z^{-1}), & \dot{f}_e(z) &= f_o(z^{-1}), \\ & & \dot{f}_o(z) &= -f_e(z^{-1}). \end{aligned}$$

**Remark:** it is possible to generalize to the  $M$ -conjugate for fixed  $M \in \mathbf{Z}$ :

$$\dot{f}_n = (-1)^n f_{2M+1-n} \quad \Rightarrow \quad f_n = (-1)^{1-n} \dot{f}_{2M+1-n}, \quad \Rightarrow \quad \dot{\dot{F}} = -F$$

For the  $M$ -conjugate of filter  $F$ , compute

$$\begin{aligned} \dot{f}(z) &= -z^{-2M-1}f(-z^{-1}), & \dot{f}_e(z) &= z^{-2M}f_o(z^{-1}), \\ & & \dot{f}_o(z) &= -z^{-2M}f_e(z^{-1}). \end{aligned}$$

## Filter and Adjoint Filter Action on $Z$ Transforms

$$\begin{aligned}
 Fx(z) &= \sum_m Fx_m z^{-m} = \sum_m \sum_n f_{2m-n} x_n z^{-m} \\
 &= \sum_m \sum_n f_{2m-2n} x_{2n} z^{-m} + \sum_m \sum_n f_{2m-2n-1} x_{2n+1} z^{-m} \\
 &= \left( \sum_m f_{2m} z^{-m} \right) \left( \sum_n x_{2n} z^{-n} \right) + \left( \sum_m f_{2m-1} z^{-m} \right) \left( \sum_n x_{2n+1} z^{-n} \right) \\
 &= \left( \sum_m f_{2m} z^{-m} \right) \left( \sum_n x_{2n} z^{-n} \right) + z^{-1} \left( \sum_m f_{2m+1} z^{-m} \right) \left( \sum_n x_{2n+1} z^{-n} \right) \\
 &= f_e(z) x_e(z) + z^{-1} f_o(z) x_o(z);
 \end{aligned}$$

$$\begin{aligned}
 F^* x(z) &= \sum_n F^* x_n z^{-n} = \sum_n \sum_m \bar{f}_{2m-n} x_m z^{-n} \\
 &= \sum_m \sum_n \bar{f}_n x_m z^{-n-2m} = \left( \sum_n \bar{f}_n z^{-n} \right) \left( \sum_m x_m z^{-2m} \right) \\
 &= \bar{f}(z) x(z^2),
 \end{aligned}$$

## Filter and Adjoint Filter Action on $Z$ Transforms II

Alternate “correlation” definition of (finite) filter and adjoint:

$$Fx_m = \sum_n f_{2m+n}x_n, \quad m \in \mathbf{Z}; \quad F^*x_n = \sum_m \bar{f}_{2m+n}x_m, \quad n \in \mathbf{Z}.$$

$$\begin{aligned} Fx(z) &= \sum_m Fx_m z^{-m} = \sum_m \sum_n f_{2m+n}x_n z^{-m} \\ &= \sum_m \sum_n f_{2m+2n}x_{2n}z^{-m} + \sum_m \sum_n f_{2m+2n+1}x_{2n+1}z^{-m} \\ &= \left( \sum_m f_{2m}z^{-m} \right) \left( \sum_n x_{2n}z^{-n} \right) + \left( \sum_m f_{2m+1}z^{-m} \right) \left( \sum_n x_{2n+1}z^{-n} \right) \\ &= f_e(z)x_e(z) + f_o(z)x_o(z), \end{aligned}$$

$$\begin{aligned} F^*x(z) &= \sum_n F^*x_n z^{-n} = \sum_n \sum_m \bar{f}_{2m+n}x_m z^{-n} \\ &= \sum_m \sum_n \bar{f}_n x_m z^{-n+2m} = \left( \sum_n \bar{f}_n z^{-n} \right) \left( \sum_m x_m z^{2m} \right) \\ &= \bar{f}(z)x(z^{-2}), \end{aligned}$$

## Perfect Reconstruction in Terms of $Z$ Transforms

Perfect reconstruction conditions for filters  $H, G, \tilde{H} = \dot{G}, \tilde{G} = \dot{H}$ , in terms of their  $Z$  transforms:

$$h(z)\tilde{h}(z^{-1}) + g(z)\tilde{g}(z^{-1}) = 1; \quad h(z)\tilde{h}(-z^{-1}) + g(z)\tilde{g}(-z^{-1}) = 0.$$

In terms of the even and odd parts:

$$h_e(z)\tilde{h}_e(z^{-1}) + g_e(z)\tilde{g}_e(z^{-1}) = 1; \quad h_e(z)\tilde{h}_o(z^{-1}) + g_e(z)\tilde{g}_o(z^{-1}) = 0;$$

$$h_o(z)\tilde{h}_o(z^{-1}) + g_o(z)\tilde{g}_o(z^{-1}) = 1; \quad h_o(z)\tilde{h}_e(z^{-1}) + g_o(z)\tilde{g}_e(z^{-1}) = 0$$

## The Polyphase Representation

The *polyphase matrix* of a pair  $H, G$  of finite filters and their conjugates  $\tilde{H} = \dot{G}$ ,  $\tilde{G} = \dot{H}$ , is

$$P(z) = \begin{bmatrix} h_e(z) & g_e(z) \\ h_o(z) & g_o(z) \end{bmatrix}, \quad \tilde{P}(z) = \begin{bmatrix} \tilde{h}_e(z) & \tilde{g}_e(z) \\ \tilde{h}_o(z) & \tilde{g}_o(z) \end{bmatrix}$$

$P$  and  $\tilde{P}$  belong to  $\text{Mat}(2 \times 2, \mathbf{C}[z, z^{-1}])$ , the ring of  $2 \times 2$  matrices over the Laurent polynomials.

The perfect reconstruction condition for even and odd parts is equivalent to:

$$P(z)\tilde{P}(z^{-1})^t = Id.$$

In practice,  $Id$  may be replaced by any diagonal matrix that is invertible in  $\text{Mat}(2 \times 2, \mathbf{C}[z, z^{-1}])$ , the  $2 \times 2$  matrices over the Laurent polynomials.

## Matrices of Laurent Polynomials

Matrix ring  $\text{Mat}(2 \times 2, \mathbf{C}[z, z^{-1}])$  has elements:

$$M(z) = \begin{bmatrix} a(z) & b(z) \\ c(z) & d(z) \end{bmatrix}, \quad a, b, c, d \in \mathbf{C}[z, z^{-1}].$$

$M$  is invertible iff  $\det M \in \mathbf{C}[z, z^{-1}]$  is invertible, namely is a monomial  $Kz^n$ .

Then

$$M^{-1}(z) = K^{-1}z^{-n} \begin{bmatrix} d(z) & -b(z) \\ -c(z) & a(z) \end{bmatrix}.$$

Say that a Laurent polynomial  $h(z)$  has a *complement*  $g = g(z)$  if their polyphase matrix is invertible.

Finite filter  $H$  is part of a perfect reconstruction pair iff its  $Z$ -transform has a complement. This reduces part of filter design to algebra.

## Division for Laurent Polynomials

$\mathbb{C}[z, z^{-1}]$  is a Euclidean domain: for  $a, b \in \mathbb{C}[z, z^{-1}]$  with  $\deg a \geq \deg b \geq 0$ ,

**Lemma 1** *There exists a quotient  $q$  and a remainder  $r$  with  $\deg r < \deg b$  so that*

$$a(z) = q(z)b(z) + r(z).$$

*Note that  $\deg q = \deg a - \deg b$ .*

Write  $q = a/b$  and  $r = a \% b$ , as in  $\mathbb{C}$ , but note that neither  $q$  nor  $r$  are unique.

**Lemma 2** *There are at most  $2^{1+\deg a - \deg b}$  different ways to divide  $a(z)/b(z)$ , among which at most  $2 + \deg a - \deg b$  quotients are different.*

**Example:** Let  $a(z) = 2z^{-1} + 4 + z$  and  $b(z) = 1 + z$ , then

$$a(z) = (2z^{-1} + 1)b(z) + 1 \quad (\text{symmetric division})$$

$$a(z) = (3z^{-1} + 1)b(z) + (-z^{-1}) \quad (\text{right division})$$

$$a(z) = (2z^{-1} + 2)b(z) + (-z) \quad (\text{left division})$$

Note: division is a generalization of Gaussian elimination.

## Greatest Common Divisors for Laurent Polynomials

Write  $b|a$  ( $b$  divides  $a$ ) if  $a = qb + 0$  for some  $q$ . Thus  $b|a \Rightarrow \deg b \leq \deg a$ .

Say that  $d$  is a *common divisor* of  $a$  and  $b$  if  $d|a$  and  $d|b$ .

Say that a common divisor  $d$  is a *greatest common divisor* of  $a$  and  $b$  if every common divisor  $c$  of  $a$  and  $b$  also divides  $d$ .

**Lemma 3** *If  $d_1$  and  $d_2$  are greatest common divisors for  $a$  and  $b$ , then  $d_1 = ud_2$  for some unit  $u \in \mathbf{C}[z, z^{-1}]$ .*

**Theorem 4** *Every pair  $a, b \in \mathbf{C}[z, z^{-1}]$ , not both zero, has a greatest common divisor that is unique up to multiplication by a unit.*

Denote this set of greatest common divisors by  $\gcd(a, b)$ .

Say that  $a, b$  are *coprime* if  $\gcd(a, b)$  is contained in the set of units.

## Euclidean Algorithm for Laurent Polynomials

Assume  $a, b$  are Laurent polynomials with  $\deg a \geq \deg b \geq 0$ .

Put  $a_0 \stackrel{\text{def}}{=} a$  and  $b_0 \stackrel{\text{def}}{=} b$ , and define  $a_k, b_k$  recursively:

$$a_{k+1} = b_k; \quad b_{k+1} = a_k - q_k b_k, \quad k = 0, 1, 2, \dots,$$

where  $q_k$  is one of the possible quotients  $a_k/b_k$ . It thus determines  $b_{k+1}$  as the corresponding one of the possible remainders  $a_k \% b_k$ .

**Lemma 5** *Let  $n$  be the smallest positive integer for which  $b_n = 0$ . Then  $a_n \in \gcd(a, b)$ .*

Note that  $a_n$  is determined only up to a unit, defined by the sequence of quotients  $q_0, \dots, q_{n-1}$ .

**Theorem 6** *Laurent polynomial  $h$  has a complement  $g$  if and only if  $h_e$  and  $h_o$  are coprime.*

Proof: apply the Euclidean Algorithm to find the polyphase matrix.

## The Euclidean Algorithm as Matrix Factorization

Write the recursion in matrix form:

$$\begin{bmatrix} a_{k+1}(z) \\ b_{k+1}(z) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & -q_k(z) \end{bmatrix} \begin{bmatrix} a_k(z) \\ b_k(z) \end{bmatrix}, \quad \Rightarrow \begin{bmatrix} a_n(z) \\ 0 \end{bmatrix} = \prod_{k=1}^n \begin{bmatrix} 0 & 1 \\ 1 & -q_{n-k}(z) \end{bmatrix} \begin{bmatrix} a(z) \\ b(z) \end{bmatrix}.$$

Inverting the product of matrices gives

$$\begin{bmatrix} a(z) \\ b(z) \end{bmatrix} = (-1)^n \prod_{k=0}^{n-1} \begin{bmatrix} q_k(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} a_n(z) \\ 0 \end{bmatrix}$$

If  $n$  is odd, absorb the unit  $(-1)^n$  term into  $a_n$ .

Put  $a = h_e$  and  $b = h_o$  and assume  $\gcd(h_e, h_o) = Kz^m$ ,  $K \neq 0$ . Define  $g_e, g_o$  by

$$P(z) \stackrel{\text{def}}{=} \begin{bmatrix} h_e(z) & g_e(z) \\ h_o(z) & g_o(z) \end{bmatrix} = \prod_{k=0}^{n-1} \begin{bmatrix} q_k(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} Kz^m & 0 \\ 0 & K^{-1}z^{-m} \end{bmatrix}.$$

Then  $P(z)$  is evidently invertible. Get  $\tilde{h}, \tilde{g}$  from  $\tilde{P}(z^{-1})^t = P(z)^{-1}$ .

## Factorization into Lifting Steps

**Theorem 7 (Daubechies and Sweldens)** *For every perfect reconstruction finite filter pair  $(H, G)$  with polyphase matrix  $P$ , there exist finitely many Laurent polynomials  $s_i(z)$  and  $t_i(z)$ ,  $1 \leq i \leq m < \infty$ , and a non-zero constant  $K$  such that*

$$P(z) = \prod_{i=1}^m \begin{bmatrix} 1 & s_i(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ t_i(z) & 1 \end{bmatrix} \begin{bmatrix} K & 0 \\ 0 & 1/K \end{bmatrix}.$$

Unit upper triangular is called *Prediction*:  $u_e \leftarrow u_e + Su_o$ .

Unit lower triangular is called *Updating*:  $u_o \leftarrow u_o + Tu_e$ .

Last diagonal matrix is called *Scaling*:  $u_e \leftarrow Ku_e$ ,  $u_o \leftarrow K^{-1}u_o$ .

Since  $u_e, u_o$  may be stored as disjoint arrays, this transform can be performed in place, without extra memory for temporary results.

## Convert Factors into Lifting Steps

Proof through a series of observations. First:

$$\begin{bmatrix} q_k(z) & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & q_k(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ q_k(z) & 1 \end{bmatrix}.$$

The *flip* matrices  $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$  cancel if Predict and Update steps alternate.

Factor a leftover flip matrix into lifting steps in a number of ways:

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

## Factoring Shift Matrices Into Lifting Steps

$$\begin{aligned} \begin{bmatrix} z & 0 \\ 0 & z^{-1} \end{bmatrix} &= \begin{bmatrix} 1 & -z \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ z^{-1} & 1 \end{bmatrix} \begin{bmatrix} 1 & 1-z \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1-z & 1 \end{bmatrix} \begin{bmatrix} 1 & z^{-1} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -z & 1 \end{bmatrix} \end{aligned}$$

$$\begin{bmatrix} z^{-1} & 0 \\ 0 & z \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ z & 1 \end{bmatrix} \begin{bmatrix} 1 & -z^{-1} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -1+z & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}.$$

Other factorizations exist, but at most 5 lifting steps are needed per shift.

Factor  $\begin{bmatrix} z^m & 0 \\ 0 & z^{-m} \end{bmatrix}$  or  $\begin{bmatrix} z^{-m} & 0 \\ 0 & z^m \end{bmatrix}$  into  $5m$  lifting steps

## Nearest Neighbor Factorization

The lifting factorization

$$P(z) = \prod_{k=1}^n \begin{bmatrix} 1 & s_k(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ t_k(z) & 1 \end{bmatrix} \begin{bmatrix} K & 0 \\ 0 & 1/K \end{bmatrix}.$$

uses only *nearest neighbors* if it satisfies the following conditions:

$$\begin{aligned} s_k(z) &= \alpha_k + \beta_k z^{-1}, \\ t_k(z) &= \gamma_k z + \delta_k, \end{aligned}$$

with  $\alpha_k, \beta_k, \gamma_k, \delta_k \in \mathbf{C}$ .

Nearest neighbor predict step:  $u_{2k} \leftarrow u_{2k} + \alpha_k u_{2k-1} + \beta_k u_{2k+1}$ .

Nearest neighbor update step:  $u_{2k+1} \leftarrow u_{2k+1} + \gamma_k u_{2k} + \delta_k u_{2k+2}$ .

## PRO: Easy Symmetric Extension for Nearest Neighbors

Assume  $u \in \ell^2$  is finitely supported in the index interval  $[0, N - 1]$ .

Big  $|u(0)|$  or  $|u(N - 1)|$  causes problems for filters.

Big  $|u(N - 1) - u(0)|$  causes problems for periodized filters.

If  $H, G$  are symmetric, use symmetric extension: Define

$$u(n) = \begin{cases} u(-n) & , \text{ if } -N < n < 0; \\ u(n) = u(2N - 1 - n), & \text{ if } N < n < 2N \end{cases}$$

and treat  $u$  as  $2N - 2$ -periodic. Several other extensions are possible, depending on the symmetry type of  $H, G$ .

Symmetric extension nearest neighbor predict step:

$$u_{2k} \leftarrow u_{2k} + \begin{cases} \alpha(u_{2k-1} + u_{2k+1}), & \text{ if } 2k \neq 0; \\ 2\alpha u_{2k+1}, & \text{ if } 2k = 0. \end{cases}$$

Symmetric extension nearest neighbor update step:

$$u_{2k+1} \leftarrow u_{2k+1} + \begin{cases} \gamma(u_{2k} + u_{2k+2}), & \text{ if } 2k + 1 \neq N - 1; \\ 2\gamma u_{N-2}, & \text{ if } 2k + 1 = N - 1. \end{cases}$$

## Example

Not all perfect reconstruction filters give nearest neighbor factorization directly.

Let

$$h(z) = \frac{1}{\sqrt{2}}(1 + z^{-9}) \quad g(z) = \frac{1}{\sqrt{2}}(-z^8 + z^{-1}).$$

This is similar to the Haar orthogonal filter pair.

Then

$$h_e(z) = \frac{1}{\sqrt{2}} \quad g_e(z) = \frac{1}{\sqrt{2}}z^4$$

$$h_o(z) = \frac{1}{\sqrt{2}}z^{-4} \quad g_o(z) = \frac{1}{\sqrt{2}}$$

The ordinary lifting factorization gives:

$$P(z) = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & z^4 \\ z^{-4} & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ z^{-4} & 1 \end{bmatrix} \begin{bmatrix} 1 & -\frac{1}{2}z^4 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 \\ 0 & \sqrt{2} \end{bmatrix}$$

This is not a nearest-neighbor factorization.

## Factoring To Degree 1

Nearest neighbor lifting steps have off-diagonal terms  $s(z), t(z)$  with  $\deg s \leq 1$  and  $\deg t \leq 1$ .

Obtain this condition by decomposition:

$$\begin{bmatrix} 1 & s_1(z) + s_2(z) \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & s_1(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & s_2(z) \\ 0 & 1 \end{bmatrix},$$

$$\begin{bmatrix} 1 & 0 \\ t_1(z) + t_2(z) & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ t_1(z) & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ t_2(z) & 1 \end{bmatrix}.$$

But note that this may create multiple successive predict factors and multiple successive update factors.

## Partial Long Division

Obtain degree 1 lifting factors at the Euclidean algorithm stage:

**Theorem 8** *Assume  $a$  and  $b$  are two coprime nonzero Laurent polynomials. Then there exist Laurent polynomials  $q_1, \dots, q_n$  with  $\deg q_k \leq 1$  for all  $k = 1, \dots, n$ , such that*

$$\begin{bmatrix} a(z) \\ b(z) \end{bmatrix} = \prod_{k=1}^n \begin{bmatrix} q_k(z) & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} K \\ 0 \end{bmatrix},$$

where  $n \leq 2(\deg a + \deg b + 1)$ .

But note that the degree condition is not enough to guarantee that the factorization gives a nearest neighbor filter transform.

From Degree 1 to Nearest Neighbor

$$\begin{bmatrix} 1 & z^{2m}(\alpha z^{-1} + \beta) \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} z^m & 0 \\ 0 & z^{-m} \end{bmatrix} \begin{bmatrix} 1 & \alpha z^{-1} + \beta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} z^{-m} & 0 \\ 0 & z^m \end{bmatrix},$$

$$\begin{bmatrix} 1 & 0 \\ z^{2m}(\gamma + \delta z) & 1 \end{bmatrix} = \begin{bmatrix} z^{-m} & 0 \\ 0 & z^m \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \gamma + \delta z & 1 \end{bmatrix} \begin{bmatrix} z^m & 0 \\ 0 & z^{-m} \end{bmatrix},$$

where  $m$  is any integer and  $\alpha, \beta, \gamma, \delta$  are constants.

Then factor the  $z^m$  shifts into at most  $5m$  nearest neighbor lifting steps.

## Condition Number for Filter Bank

If  $P(z)$  is the polyphase matrix of a perfect reconstruction filter pair, then

$$\text{cond}(P) \stackrel{\text{def}}{=} \frac{\sup \left\{ \sqrt{\lambda_{\max}(P(z)^* P(z))} : |z| = 1 \right\}}{\inf \left\{ \sqrt{\lambda_{\min}(P(z)^* P(z))} : |z| = 1 \right\}}.$$

If  $P = P_1 \cdots P_n$ , then  $\text{cond}(P) \leq \text{cond}(P_1) \cdots \text{cond}(P_n)$ .

Assume the worst case (equality) and estimate the condition number for three implementations of the filter bank:

1. the original polyphase matrix  $P$ ,
2. the usual (shortest) lifting factorization of  $P$ , and
3. the nearest-neighbor lifting factorization of  $P$ .

## CON: Condition Numbers for Nearest Neighbor Factorization

Filter	Cond of $P(z)$	Cond of Lifting	Cond of N-N
9-7	1.32	205	205
D4	1	77	77
D6	1	76	76
Cubic B-spline	4	56	56
CDF-1-1	1	8.59	8.59
CDF-1-3	1.28	8.72	3100
CDF-1-5	1.42	6.25	1200
CDF-2-2	2	8.59	8.59
CDF-2-4	2	99	1900
CDF-3-1	4	643	643
CDF-3-3	4	723	3200
CDF-4-2	8	111	111
CDF-4-4	8	113	2800

## PRO: Choose an Optimizing Lifting Factorization

Choose the lifting steps to obtain a nearest neighbor algorithm.

Choose the sequence of quotients  $\{q_k(z) : k = 0, 1, \dots, n - 1\}$  to minimize condition number.

Choose the filter indexing to minimize the number of  $z^m$  shift matrices.

## References

- C. Brislawn. “Classification of Nonexpansive Symmetric Extension Transforms for Multirate Filter Banks.” *Applied and Computational Harmonic Analysis*, 3:4(1996),337–357.
- I. Daubechies and W. Sweldens. “Factoring Wavelet Transforms into Lifting Steps.” *Fourier Analysis and Applications* 4:3(1998),245–267.
- A. Jensen and A. la Cour-Harbo. *Ripples in Mathematics: The Discrete Wavelet Transform*. Springer-Verlag, Berlin, 2001. ISBN 3-540-41662-5.
- G. Strang and T. Nguyen. *Wavelets and Filter Banks*. Wellesley–Cambridge Press, Wellesley, Massachusetts, 1996. ISBN 0-9614088-7-1.
- M. V. Wickerhauser. *Mathematics for Multimedia*. Elsevier/Academic Press, San Diego, California, 2003. ISBN 0-12-748451-5.