Maple and the Parameterization of Orthogonal Wavelet Bases

Ivan W. Selesnick

October 24, 1997

1 Parameterization

Our purpose is educational — to illustrate with Maple the lattice parameterization of orthogonal wavelet bases, and to obtain parameterizations of the coefficients of wavelet (2-channel) filter banks in terms of angles. We will use the lattice parameterization of 2×2 unitary matrices. This lattice parameterization, described in [4, 5], offers the opportunity to design orthogonal wavelet filters via unconstrained optimization. Also see, for example, page 137 of [3] or page 160 of [1].

Let $H_0(z)$ and $H_1(z)$ be the two filters of a unitary 2-channel filter bank. (Usually $H_0(z)$ is lowpass, and $H_1(z)$ is highpass). The polyphase components of $H_0(z)$ will be denoted by $H_{00}(z)$ and $H_{01}(z)$. That is,

$$H_0(z) = H_{00}(z^2) + z^{-1} H_{01}(z^2).$$

The polyphase components give the even and odd indexed coefficients of $H_0(z)$ separately. Similarly, the polyphase components of $H_1(z)$ will be denoted by $H_{10}(z)$ and $H_{11}(z)$ so that

$$H_1(z) = H_{10}(z^2) + z^{-1} H_{11}(z^2).$$

The *polyphase* matrix of the 2-channel filter bank is then defined as

$$H_p(z) = \begin{bmatrix} H_{00}(z) & H_{01}(z) \\ H_{10}(z) & H_{11}(z) \end{bmatrix}.$$

Definition: A 2-channel filter bank is unitary iff $H_p^t(z^{-1}) H_p(z) = I$.

The important result from [4, 5], is that for unitary filter banks, the polyphase matrix $H_p(z)$ can be written as

$$H_p(z) = \begin{bmatrix} 1 & 0 \\ 0 & \pm 1 \end{bmatrix} \cdot R(\theta_K) \cdot \Lambda(z) \cdot R(\theta_{K-1}) \cdot \Lambda(z) \cdots \Lambda(z) \cdot R(\theta_0)$$

where

$$R(\theta) = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \qquad \text{and} \qquad \Lambda(z) = \begin{bmatrix} 1 & 0 \\ 0 & z^{-1} \end{bmatrix}.$$

By changing the sign of $H_1(z)$, if necessary, $H_p(z)$ can be written as

$$H_p(z) = R(\theta_K) \cdot \Lambda(z) \cdot R(\theta_{K-1}) \cdot \Lambda(z) \cdots \Lambda(z) \cdot R(\theta_0).$$

This *lattice* parameterization gives $H_p(z)$ as a function of angles.

Note that $H_p(z)$ is of degree K. Therefore the the filters $H_0(z)$ and $H_1(z)$ are of degree $2K+1$, (they are of length $N = 2K+2$).

To associate a wavelet basis with the filter bank, it is necessary that $H_0(z = -1) = 0$. This condition can be written very simply in terms of the angles $\theta_0, \ldots, \theta_K$. They must sum to $\pi/4$:

$$\sum_{k=0}^{K} \theta_k = \pi/4.$$

2 Maple

In the previous section, the polyphase matrix matrix was given in terms of angles. To obtain expressions for the coefficients of $H_0(z)$ in terms of angles, it is necessary to multiply out the matrix product. This is rather cumbersome by hand in general, as the process involves matrices the entries of which are polynomials. However, the use of Maple makes this process simple. So we take this opportunity to illustrate the use of Maple, and to clarify the procedure by which one obtains coefficients from angles.

Here is the Maple code to obtain a parameterization for $H_0(z)$ of length $N = 8$. The general process should be clear, given the above formulas, but we can hardly say that Maple code is transparent. A useful introduction to Maple is given in [2].

```
# Maple program for the parameterization of a
# length N scaling filter in terms of angles.

with(linalg):           # load linear algebra package

N := 8;                 # filter length
K := (N-2)/2;           # number of angles (N = 2K+2)
R := t -> matrix(2,2, [cos(t), sin(t), -sin(t), cos(t)]);
Lambda := matrix(2,2,[1, 0, 0, 1/z]);

Hp := R(t0):            # construct polyphase matrix
i := 'i':
for i from 1 to K do
        Hp := evalm(R(t.i) &* Lambda &* Hp):
od:
H00 := expand(Hp[1,1]): # extract H00 and H01
H01 := expand(Hp[1,2]): # expand polynomials

# display coefficients as functions of angles
lprint(t.K = Pi/4 - sum('t.i','i'=0..(K-1)));
i := 'i':
n := 1:
for i from 0 to K do
        lprint(h[n] = coeff(H00,z,-i));
        n := n + 1:
        lprint(h[n] = coeff(H01,z,-i));
        n := n + 1:
od:
```

This maple code gives the following:

```
t3 = 1/4*Pi-t0-t1-t2
h[1] = cos(t3)*cos(t2)*cos(t1)*cos(t0)
h[2] = cos(t3)*cos(t2)*cos(t1)*sin(t0)
h[3] = -cos(t3)*cos(t2)*sin(t1)*sin(t0)-cos(t3)*sin(t2)*sin(t1)*cos(t0)-
sin(t3)*sin(t2)*cos(t1)*cos(t0)
h[4] = cos(t3)*cos(t2)*sin(t1)*cos(t0)-cos(t3)*sin(t2)*sin(t1)*sin(t0)-
sin(t3)*sin(t2)*cos(t1)*sin(t0)
h[5] = -cos(t3)*sin(t2)*cos(t1)*sin(t0)+sin(t3)*sin(t2)*sin(t1)*sin(t0)-
sin(t3)*cos(t2)*sin(t1)*cos(t0)
h[6] = cos(t3)*sin(t2)*cos(t1)*cos(t0)-sin(t3)*sin(t2)*sin(t1)*cos(t0)-
sin(t3)*cos(t2)*sin(t1)*sin(t0)
h[7] = -sin(t3)*cos(t2)*cos(t1)*sin(t0)
h[8] = sin(t3)*cos(t2)*cos(t1)*cos(t0)
```

When $N = 6$, the Maple code gives

```
t2 = 1/4*Pi-t0-t1
h[1] = cos(t2)*cos(t1)*cos(t0)
h[2] = cos(t2)*cos(t1)*sin(t0)
h[3] = -cos(t2)*sin(t1)*sin(t0)-sin(t2)*sin(t1)*cos(t0)
h[4] = cos(t2)*sin(t1)*cos(t0)-sin(t2)*sin(t1)*sin(t0)
h[5] = -sin(t2)*cos(t1)*sin(t0)
h[6] = sin(t2)*cos(t1)*cos(t0)
```

Comparing this with, for example, the formulas for the length 6 case on page 66 of [1], the expressions appear somewhat different. However, with the right trigonometric identities, they must be the same.

The Maple code in this report is also available electronically. Also available is Matlab code for scaling filters of lengths 6, 8, and 10. If desired, code for the parameterization of longer scaling filters can be easily generated with exactly the same Maple code. See URL `http://www.dsp.rice.edu/~selesi/`.

References

[1] C. S. Burrus, R. A. Gopinath, and H. Guo. *Introduction to Wavelets and Wavelet Transforms.* Prentice Hall, 1997.

[2] R. M. Corless. *Essential Maple: An Introduction for Scientific Programmers.* Springer-Verlag, 1995.

[3] G. Strang and T. Nguyen. *Wavelets and Filter Banks.* Wellesley-Cambridge Press, 1996.

[4] P. P. Vaidyanathan. *Multirate Systems and Filter Banks.* Prentice Hall, 1992.

[5] P. P. Vaidyanathan and P.-Q. Hoang. Lattice structures for optimal design and robust implementation of two-channel perfect reconstruction QMF banks. *IEEE Trans. on Signal Processing*, 36(1):81–93, January 1988.