Pollen Bases and Daubechies-Lagarias Algorithm in MATLAB

Brani Vidakovic

Daubechies-Lagarias algorithm provides fast tool for calculating various orthonormal wavelets. Matlab programs implementing the algorithm on Pollen type of wavelet bases are given and their usage is illustrated.

Pollen-Type Parameterizations of Wavelet Bases

Why Pollen? It generates a family possessing continuum many wavelet bases of various degrees of regularity.

It is possible to construct of a family of wavelets that contains continuum many different wavelet bases.

Let h be a wavelet filter of length $2N$. Pollen showed that there is a continuous mapping from $[0, 2\pi]^{N-1}$ to a set of "wavelet solutions" in the form of a sequence $h = \{h_0, h_1, \ldots, h_{2N-1}\}$.

Pollen representations of all wavelet solutions of lengths 4 ($N = 2$) and 6 ($N = 3$) are given in Tables 1 and 2.

A special case of Pollen's representation for $\varphi = \frac{\pi}{6}$ gives DAUB2 filter.

Pollen wavelets provide an interesting library of wavelets that comprises of continuum many different wavelet bases indexed by $\varphi \in [0, 2\pi]$.

Fig. (1) depicts wavelet and scaling functions for $\varphi = \frac{\pi}{4}$. Its low-pass coefficients are $h_0 = \frac{\sqrt{2}}{4}$, $h_1 = \frac{2+\sqrt{2}}{4}$, $h_2 = \frac{\sqrt{2}}{4}$, and $h_3 = \frac{-2+\sqrt{2}}{4}$, and the

Table 1: Pollen parameterization for $N = 2$ (four-tap filters). $[s = 2\sqrt{2}]$

n	h_n for $N = 2$
0	$(1 + \cos\varphi - \sin\varphi)/s$
1	$(1 + \cos\varphi + \sin\varphi)/s$
2	$(1 - \cos\varphi + \sin\varphi)/s$
3	$(1 - \cos\varphi - \sin\varphi)/s$

plots in Fig. 1 are obtained by point-to-point application of the Daubechies-Lagarias algorithm, described below.

Daubechies-Lagarias Algorithm

Why Daubechies-Lagarias? It provides a fast and economic calculation of the value of scaling and wavelets functions at the point. Alternative is Mallat's algorithm which will calculate much more than needed. Such evaluations are essential for wavelet density estimators, wavelet based classification, etc.

Except for the Haar wavelet, all compactly supported orthonormal families of wavelets (e.g., Daubechies, Symmlet, Coiflet, etc.) scaling and wavelet functions have no a closed form. A non-elegant solution is to have values of the mother and father wavelet given in a table. Evaluation of $\phi_{jk}(x)$ or $\psi_{jk}(x)$, for given x, then can be performed by interpolating the table values.

Based on Daubechies and Lagarias (1992) *local pyramidal algorithm* a solution is proposed. A brief theoretical description and MATLAB program are provided.

Table 2: Pollen parameterization for $N = 3$ (six-tap filters) $[s = 2\sqrt{2}]$.

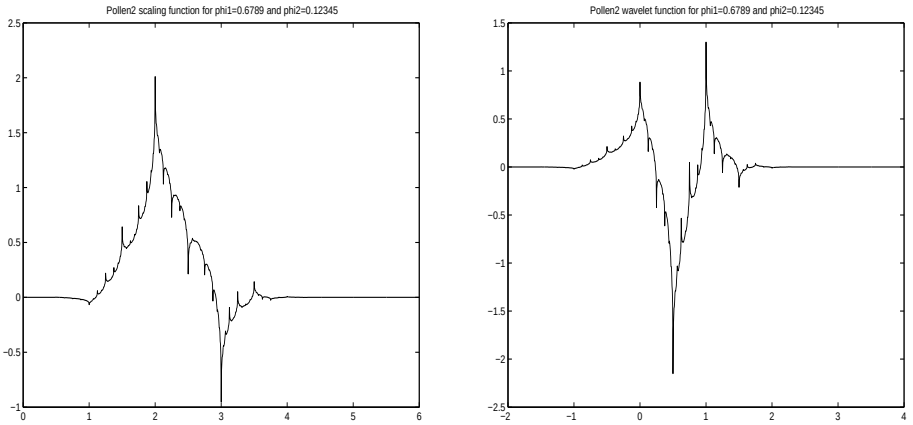
n	h_n for $N = 3$
0	$(1 + \cos\varphi_1 - \cos\varphi_2 - \cos\varphi_1\cos\varphi_2 + \sin\varphi_1 - \cos\varphi_2\sin\varphi_1 - \sin\varphi_2 + \cos\varphi_1\sin\varphi_2 - \sin\varphi_1\sin\varphi_2)/(2s)$
1	$(1 - \cos\varphi_1 + \cos\varphi_2 - \cos\varphi_1\cos\varphi_2 + \sin\varphi_1 + \cos\varphi_2\sin\varphi_1 - \sin\varphi_2 - \cos\varphi_1\sin\varphi_2 - \sin\varphi_1\sin\varphi_2)/(2s)$
2	$(1 + \cos\varphi_1\cos\varphi_2 + \cos\varphi_2\sin\varphi_1 - \cos\varphi_1\sin\varphi_2 + \sin\varphi_1\sin\varphi_2)/s$
3	$(1 + \cos\varphi_1\cos\varphi_2 - \cos\varphi_2\sin\varphi_1 + \cos\varphi_1\sin\varphi_2 + \sin\varphi_1\sin\varphi_2)/s$
4	$(1 - \cos\varphi_1 + \cos\varphi_2 - \cos\varphi_1\cos\varphi_2 - \sin\varphi_1 - \cos\varphi_2\sin\varphi_1 + \sin\varphi_2 + \cos\varphi_1\sin\varphi_2 - \sin\varphi_1\sin\varphi_2)/(2s)$
5	$(1 + \cos\varphi_1 - \cos\varphi_2 - \cos\varphi_1\cos\varphi_2 - \sin\varphi_1 + \cos\varphi_2\sin\varphi_1 + \sin\varphi_2 - \cos\varphi_1\sin\varphi_2 - \sin\varphi_1\sin\varphi_2)/(2s)$

Let ϕ be the scaling function of a compactly supported wavelet generating an orthogonal MRA. Suppose the support of ϕ is $[0, N]$. Let $x \in (0, 1)$, and let $dyad(x) = \{d_1, d_2, \dots d_n, \dots\}$ be the set of 0-1 digits in dyadic representation of x $(x = \sum_{j=1}^{\infty} d_j 2^{-j})$. By $dyad(x, n)$ we denote the subset of the first n digits from $dyad(x)$, i.e., $dyad(x, n) = \{d_1, d_2, \dots d_n\}$.

Let $h = (h_0, h_1, \dots, h_N)$ be the vector of wavelet filter coefficients. Define two $N \times N$ matrices as

$$T_0 = \sqrt{2}(h_{2i-j-1})_{1\le i,j\le N}, \text{and} \quad T_1 = \sqrt{2}(h_{2i-j})_{1\le i,j\le N}. \tag{1}$$

Then



(a) (b)

Figure 1: Pollen2 parameterization, $\varphi_1 = 0.67890, \varphi_2 = 0.12345$. Scaling [panel (a)] and wavelet [panel (b)] functions. The filter for this basis is: {-0.0153, 0.2469, 0.8404, 0.4675, -0.1180, -0.0073}.

Theorem 1 *(Daubechies and Lagarias, 1992.)*

$$\lim_{n\to\infty} T_{d_1} \cdot T_{d_2} \cdot \cdots \cdot T_{d_n} = \begin{bmatrix} \phi(x) & \phi(x) & \dots & \phi(x) \\ \phi(x+1) & \phi(x+1) & \dots & \phi(x+1) \\ \vdots & & & \\ \phi(x+N-1) & \phi(x+N-1) & \dots & \phi(x+N-1) \end{bmatrix}. \quad (2)$$

The convergence of $||T_{d_1} \cdot T_{d_2} \cdot \cdots \cdot T_{d_n} - T_{d_1} \cdot T_{d_2} \cdot \cdots \cdot T_{d_{n+m}}||$ to zero, for fixed m, is exponential and constructive, i.e., effective bounds, that decrease exponentially to 0, can be established.

Example: Consider the DAUB #2 case (**n**=2, N=3). The corresponding filter is $(\frac{1+\sqrt{3}}{4\sqrt{2}}, \frac{3-\sqrt{3}}{4\sqrt{2}}, \frac{3+\sqrt{3}}{4\sqrt{2}}, \frac{1-\sqrt{3}}{4\sqrt{2}})$. According to (1) the matrices T_0 and T_1

are given as:

$$T_0 = \begin{bmatrix} \frac{1+\sqrt{3}}{4} & 0 & 0 \\ \frac{3-\sqrt{3}}{4} & \frac{3+\sqrt{3}}{4} & \frac{1+\sqrt{3}}{4\sqrt{2}} \\ 0 & \frac{1-\sqrt{3}}{4} & \frac{3-\sqrt{3}}{4} \end{bmatrix}, \text{and} \qquad T_1 = \begin{bmatrix} \frac{3+\sqrt{3}}{4} & \frac{1+\sqrt{3}}{4} & 0 \\ \frac{1-\sqrt{3}}{4} & \frac{3-\sqrt{3}}{4} & \frac{3+\sqrt{3}}{4} \\ 0 & 0 & \frac{1-\sqrt{3}}{4} \end{bmatrix}.$$

If, for instance, $x = 0.45$, then $dyad(0.45, 20) = \{$ 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1 $\}$. The values $\phi(0.45), \phi(1.45)$, and $\phi(2.45)$ are calculated as

$$\prod_{i \in dyad(0.45,20)} T_i = \begin{bmatrix} 0.86480582 & 0.86480459 & 0.86480336 \\ 0.08641418 & 0.08641568 & 0.08641719 \\ 0.04878000 & 0.04877973 & 0.04877945 \end{bmatrix}.$$

The Daubechies and Lagarias algorithm gives only the values of the scaling function. However, most of the evaluation needed involves the mother wavelet. It turns out that another algorithm is unnecessary, due to Theorem 2.

Theorem 2 *Let x be arbitrary real number. Let the orthonormal MRA be given by wavelet filter coefficients $\{h_0, h_1, \ldots, h_N\}$, $N = 2n-1$. Define vector u with N components as:*

$$u(x) = \{(-1)^{1-[2x]} h_{i+1-[2x]}, i = 0, N-1\} \tag{3}$$

If for some i the index $i + 1 - [2x]$ is negative or larger than N then the corresponding component of u is zero.

Let the vector v be defined as

$$v(x, n) = \frac{1}{N} \mathbf{1}' \prod_{i \in dyad(\{2x\}, n)} T_i \tag{4}$$

Then

$$\psi(x) = \lim_{n\to\infty} u(x)'v(x,n). \tag{5}$$

Matlab Codes

Matlab functions `MakePollen1.m, MakePollen2.m, Phijk.m,` and `Psijk.m` should be placed in `Wavelab802/Orthogonal/` directory. Although the files are written in the spirit of wavelab m-files, no other wavelab-function is needed and all functions can run withouth `wavelab` installed.

1. This m-script demonstrates usage of Pollen family and Daubechies-Lagarias m-functions. It also produces Figure 1.

```
% Daubechies Lagarias for Pollen2 wavelets that produces Figure 1.
% ----------------------------------------
xxs = 0:0.005:6;
xxw = -2:0.005:4;
 yys =[];
 yyw =[];
 poll2 = MakePollen2(0.67890, 0.12345);
for i = 1:length(xxs)
    yys = [yys Phijk(xxs(i), 0, 0, poll2, 25)];
    yyw = [yyw Psijk(xxw(i), 0, 0, poll2, 25)];
end
figure(1)
plot(xxs, yys)
figure(2)
plot(xxw, yyw)
```

2. This m-function gives low pass wavelet filter from the one parameter Pollen family.

```
function h = MakePollen1(phi)
% One parameter Orthonormal Pollen Family
% y = MakePollen1(phi)
% Input: phi - angle in [0, 2 *pi)
% Output:  h - low pass ON wavelet filter
% phi=pi/6 corresponds to Daubechies 4 tap ON filter
s = 2 * sqrt(2);
h(1) = (1 + cos(phi) - sin(phi))/s;
```

```
h(2) = (1 + cos(phi) + sin(phi))/s;
h(3) = (1 - cos(phi) + sin(phi))/s;
h(4) = (1 - cos(phi) - sin(phi))/s;
% B. Vidakovic, 12/8/2002
```

3. This m-function gives low pass wavelet filter from the two parameter Pollen family. The spacial case $\varphi_1 = 0.67890$, and $\varphi_2 = 0.12345$ is plotted in Figure 1. The filter for this selection of parameters is: {-0.0153, 0.2469, 0.8404, 0.4675, -0.1180, -0.0073}.

```
function h = MakePollen2(phi1, phi2)
% Two parameter Orthonormal Pollen Family
% y = MakePollen2(phi1, phi2)
% Input: phi1, phi2 - angle in [0, 2 *pi)
% Output:  h - low pass ON wavelet filter of length 6
%
s = 2 * sqrt(2);

h( 1 ) = ( 1+cos(phi1) - cos( phi2 ) - cos( phi1) * cos( phi2) ...
       + sin( phi1 ) - cos( phi2) * sin( phi1) - sin( phi2 )  ...
       + cos( phi1) * sin( phi2) - sin( phi1) * sin( phi2 ) )/ (2*s);

h( 2 ) = ( 1-cos( phi1) + cos( phi2) - cos( phi1) * cos( phi2) ...
       + sin( phi1) + cos( phi2) * sin( phi1) - sin( phi2) ...
       - cos( phi1) * sin( phi2) - sin( phi1) * sin( phi2) )/ (2*s);

h( 3 ) = ( 1 + cos( phi1) * cos( phi2) + cos( phi2) * sin( phi1) ...
       - cos( phi1) * sin( phi2 ) + sin( phi1) *sin( phi2 ) )/s;

h( 4 ) = ( 1 + cos( phi1 ) * cos( phi2 ) - cos( phi2 ) * sin( phi1 ) ...
       + cos( phi1 )* sin( phi2 ) + sin( phi1)*  sin( phi2) )/s;

h( 5 ) = ( 1-cos( phi1) + cos( phi2 )- cos( phi1 )* cos( phi2 ) ...
       - sin( phi1 ) - cos( phi2 ) * sin( phi1 ) + sin( phi2 ) ...
       + cos( phi1 )* sin( phi2 )- sin( phi1 )* sin( phi2) ) /(2*s);

h( 6 ) = ( 1+cos( phi1 )- cos( phi2 )- cos( phi1 )* cos( phi2 ) ...
       - sin( phi1 ) + cos( phi2 ) * sin( phi1 ) + sin( phi2 ) ...
       - cos( phi1 ) * sin( phi2 ) - sin( phi1 )*  sin( phi2) )/ (2*s);
% B. Vidakovic, 12/8/2002
```

4. The functions `Phijk.m` and `Psijk.m` given below calculate the scaling (wavelet) functions at the point with preassigned precision. Functions `dec2bin.m`, `t0.m`, and `t1.m` are provided in each `Phijk.m` and `Psijk.m` for convenience.

```
function yy = Phijk(z, j, k, filter, n)
%------------------------------------------------------------
%  yy=Phijk(z, j, k, filter, n)
%  Evaluation of the scaling function corresponding to an Orthogonal
%           MRA by Daubechies-Lagarias Algorithm.
%  inputs:  z -- the argument
%           j -- scale
%           k -- shift
%           filter -- ON finite wavelet filter, might be an
%                     output of WaveLab's: MakeONFilter
%           n -- precision of approximation maesured by the number
%                of Daubechies-Lagarias steps (default n=20)
%-----------------------------------------------------------
%  output:  yy -- value of father wavelet (j,k) coresponding to
%                  'filter' at z.
%------------------------------------------------------------
% Example of use:
% > xx = 0:0.01:6;  yy=[];
% > for i=1:length(xx)
% > yy =[yy Phijk(x(i), 0, 1, MakeONFilter('Daubechies',4), 25)];
% > end
% > plot(x,yy)
%------------------------------------------------------------
    if (nargin == 4)
        n=20;
    end
    daun=length(filter)/2;
    N=length(filter)-1;
    x=(2^j)*z-k;
    if(x<=0|x>=N) yy=0;
else
    int=floor(x);
    dec=x-int;
    dy=dec2bin(dec,n);
    t0=t0(filter);
    t1=t1(filter);
    prod=eye(N);
    for i=1:n
            if dy(i)==1 prod=prod*t1;
            else prod=prod*t0;
            end
        end
    y=2^(j/2)*prod;
      yyy = mean(y');
      yy = yyy(int+1);
  end

  %--------------------functions needed---------------------------
  %---------
function a = dec2bin(x,n)
            a=[];
             for i = 1:n
```

```
            if(x <= 0.5) a=[a 0]; x=2*x;
             else a=[a 1]; x=2*x-1;
            end
           end
 %----------
 function t0 = t0(filter)
%
n = length(filter); nn = n - 1;
%
t0 = zeros(nn); for i = 1:nn
    for j= 1:nn
        if (2*i - j > 0 & 2*i - j <= n)
            t0(i,j) = sqrt(2) * filter( 2*i - j );
        end
    end
end
%------------------
function t1 = t1(filter)
%
n = length(filter); nn = n - 1;
%
t1 = zeros(nn); for i = 1:nn
    for j= 1:nn
        if (2*i -j+1 > 0 & 2*i - j+1 <= n)
            t1(i,j) = sqrt(2) * filter( 2*i - j+1 );
        end
    end
end
%---------------------- B. Vidakovic, 2002 --------------------
%---------------------------------------------------------------


function yy = Psijk(z, j, k, filt, n)
%---------------------------------------------------------------
%  yy=Psijk(z, j, k, filter, n)
%  Evaluation of the wavelet function corresponding to an Orthogonal
%           MRA by Daubechies-Lagarias Algorithm.
%  inputs:  z -- the argument
%           j -- scale
%           k -- shift
%           filter -- ON finite wavelet filter, might be an
%                     output of WaveLab's: MakeONFilter
%           n -- precision of approximation maesured by the number
%                of Daubechies-Lagarias steps (default n=20)
%-------------------------------------------------------
%  output:  yy -- value of mother wavelet (j,k) coresponding to
%                  'filter' at z.
%---------------------------------------------------------------
% Example of use:
% > xx = -1:0.01:5;  yy=[];
% > for i=1:length(xx)
% > yy =[yy Psijk(x(i), 0, 1, MakeONFilter('Daubechies',4), 25)];
% > end
```

```
% > plot(x,yy)
%---------------------------------------------------------------
    if (nargin == 4) n=20;
    end
    N=length(filt)-1;
    daun = (N+1)/2;
    x=(2^j)*z-k;
if(x<=daun-N|x>=daun) yy=0;
     %if(x<=0|x>=N) yy=0;
else
    twox = 2 * x;
    inti=floor(twox);
    dec=twox-inti;
    dy=dec2bin(dec,n);
    t0=t0(filt);
    t1=t1(filt);
    prod=eye(N);
    for i=1:n
            if dy(i)==1
                prod=prod*t1;
            else prod=prod*t0;
            end
        end
    uu=[];
    for i=1:N
        index = i + 1 - inti;
        if ( index > 0 & index < N + 2 )
             fi= (-1)^(-index)*filt(index);
        else fi=0;
        end
        uu =[uu   fi];
    end
   %----------------------------------------
   v = 1/N * ones(1,N) * prod' ;
    yy=2^(j/2)* uu * v';
   end

  %--------------------functions needed----------------------------
  %---------
  function a = dec2bin(t,n)
            a=[];
             for i = 1:n
             if(t <= 0.5) a=[a 0]; t=2*t;
              else a=[a 1]; t=2*t-1;
             end
            end
 %-----------
 function t0 = t0(filt)
%
n = length(filt); nn = n - 1;
%
t0 = zeros(nn); for i = 1:nn
```

```
    for j= 1:nn
        if (2*i - j > 0 & 2*i - j <= n)
            t0(i,j) = sqrt(2) * filt( 2*i - j );
        end
    end
end
%------------------
function t1 = t1(filt)
%
n = length(filt); nn = n - 1;
%
t1 = zeros(nn); for i = 1:nn
    for j= 1:nn
        if (2*i -j+1 > 0 & 2*i - j+1 <= n)
            t1(i,j) = sqrt(2) * filt( 2*i - j+1 );
        end
    end
end
%--------------------  B. Vidakovic, 2002   --------------------
%---------------------------------------------------------------
```

References

[1] Daubechies, I. and Lagarias, J. (1991). Two-scale difference equations I. Existence and global regularity of solutions, *SIAM J. Math. Anal.*,**22 (5)**, 1388–1410.

[2] Daubechies, I. and Lagarias, J. (1992). Two-scale difference equations II. Local regularity, infinite products of matrices and fractals, *SIAM J. Math. Anal.*,**23 (4)**, 1031–1079.

[3] Wavelab802, Matlab toolbox for wavelet analysis. Donoho, D. et al., Stanford University. `http://www-stat.stanford.edu/w̃avelab/`.

[4] Pollen, D. (1990). $SU_I(2, F[z, 1/z])$ for F a subfield of C, *J. Amer. Math. Soc.*,**3**, 611–624.

[5] Vidakovic, B. (1999). *Statistical Modeling by Wavelets.* Wiley, NY.