

1 Matlab correction

Note

This correction illustrates the wavelet decomposition in 1D or 2D.

1.1 1D signals

Two functions are required: a function that loops over the different scales and calls the second function that performs the single step wavelet decomposition. The Haar wavelet is illustrated in Fig.1.

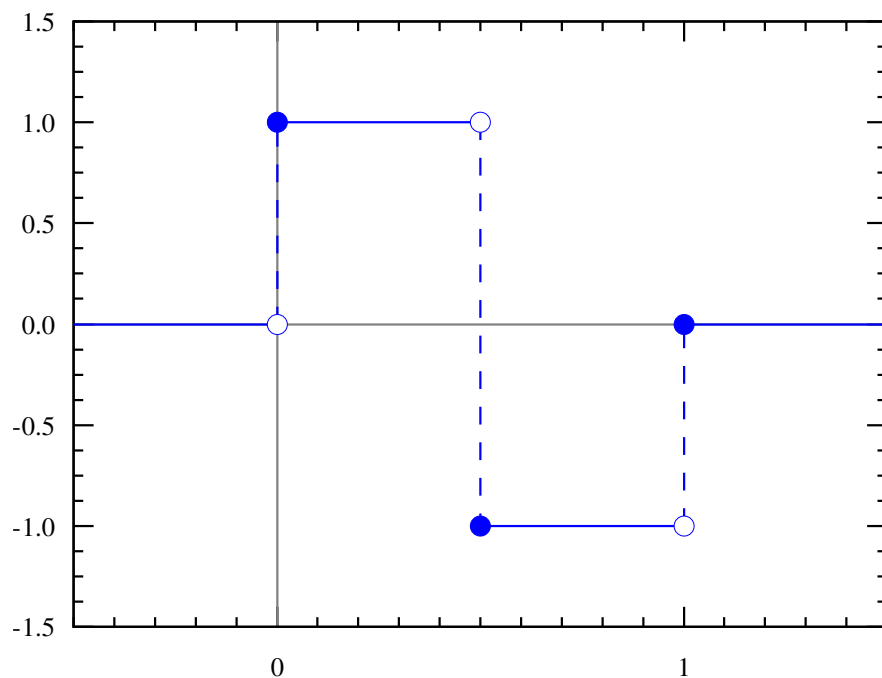


Figure 1: Haar wavelets. From wikipedia, author Omegatron.

1.1.1 Simple 1D decomposition



```
1 function C = simpleWaveDec(signal , nb_scales)
  % wavelet decomposition of <signal> into <nb_scales> scales
  3 % This function uses Haar wavelets for demonstration purposes.

  5 % Haar Wavelets filters for decomposition and reconstruction
    ld = [1 1];
  7 hd = [-1 1];
```



```

9 % transformation
  C=cell(nb_scales+1, 1);
11 A = signal; % approximation
  for i=1:nb_scales
13     [A, D] = waveSingleDec(A, ld, hd);
        % get the coefficients
15     C{i}=D;
  end
17 C{nb_scales+1}=A;

```



```

1 % function single step wavelet decomposition
  function [A, D]=waveSingleDec(signal, ld, hd)
3 % 1D wavelet decomposition into
  % A: approximation vector
5 % D: detail vector
  % ld: low pass filter
7 % hd: high pass filter

9 % convolution
  A = conv(signal, ld, 'same');
11 D = conv(signal, hd, 'same');

13 % subsampling
  A = A(1:2:end);
15 D = D(1:2:end);

```

1.1.2 Simple 1D reconstruction

The reconstruction starts from the highest scale and computes the approximation signal with the given details.



```

1 function A = simpleWaveRec(C)
  % wavelet simple reconstruction function of a 1D signal
3 % C: Wavelet coefficients
  %
5 % The Haar wavelet is used
  ld = [1 1];
7 hd = [-1 1];
  lr = ld/2;
9 hr = -hd/2;

11 nb_scales = length(C)-1;
  A = C{nb_scales+1};
13 for i=nb_scales:-1:1

```



```
A = waveSingleRec(A, C{i}, lr, hr);  
15 end
```



```
1 function approx = waveSingleRec(a, d, lr, hr)  
  % 1D wavelet reconstruction at one scale  
3 % a: vector of approximation  
  % d: vector of details  
5 % lr: low pass filter defined by wavelet  
  % hr: high pass filter defined by wavelet  
7 %  
  % This is Mallat algorithm.  
9 % NB: to avoid side effects, the convolution function does not use the '  
    ↪ same' option  
  
11 approx = zeros(1, length(a)*2);  
    approx(1:2:end) = a;  
13 approx = conv(approx, lr);  
  
15 detail = zeros(1, length(a)*2);  
    detail(1:2:end) = d;  
17 detail = conv(detail, hr);  
  
19 approx = approx + detail;  
    approx = approx(1:length(a)*2);
```

1.1.3 Results

```

Command window
>> signal = [4;8;2;3;5;18;19;20];
2 >> C = simpleWaveDec(signal,3)
C =
4      [4x1 double]
      [2x1 double]
6      [          -45]
      [           79]
8
>> for i=1:4, C{i}, end
10 ans =
      -4
12      -1
      -13
14      -1
16 ans =
       7
18     -16
20 ans =
     -45
22
24 ans =
      79

```

1.2 2D signals

1.2.1 Decomposition



```

%% 2D simple wavelet decomposition
2 function [LcLrA, HcLrA, LcHrA, HcHrA] = decWave2D(image, ld, hd)
   % wavelet decomposition of a 2D image into four new images.
4   % The image is supposed to be square, the size of it is a power of 2 in
      ↳ the x and y dimensions.

6   % We manipulate doubles
   image = double(image);
8
   %% Decomposition on rows
10  sx=size(image, 1);
   sy=size(image, 2);
12
   LrA = zeros(sx, sy/2);
14  HrA = zeros(sx, sy/2);
   for i=1:sx

```



```

16 [A, D]= waveSingleDec(image(i,:), ld, hd);
    LrA(i,:)= A;
18 HrA(i,:)= D;
    end
20

22 %% Decomposition on cols
    LcLrA = zeros(sx/2, sy/2);
24 HcLrA = zeros(sx/2, sy/2);
    LcHrA = zeros(sx/2, sy/2);
26 HcHrA = zeros(sx/2, sy/2);
    for j=1:sy/2
28 [A, D]= waveSingleDec(LrA(:,j), ld, hd);
        LcLrA(:,j) = A;
30 HcLrA(:,j) = D;

32 [A, D]= waveSingleDec(HrA(:,j), ld, hd);
        LcHrA(:,j) = A;
34 HcHrA(:,j) = D;
    end
36

38 %% Display result
    figure();
    subplot(2, 2, 1); imshow(LcLrA, []);
40 subplot(2, 2, 2); imshow(HcLrA, []);
    subplot(2, 2, 3); imshow(LcHrA, []);
42 subplot(2, 2, 4); imshow(HcHrA, []);

```



```

function C = simpleImageDec(image, nb_scales)
2 % wavelet decomposition of <image> into <nb_scales> scales
  % This function uses Haar wavelets for demonstration purposes.
4
  % Haar Wavelets filters for decomposition and reconstruction
6 ld = [1 1];
  hd = [-1 1];
8
  % transformation
10 C=cell(nb_scales+1, 1);
  A = image; % approximation
12
  coeffs = cell(3,1);
14 for i=1:nb_scales
    [A, HcLrA, LcHrA, HcHrA] = decWave2D(A, ld, hd);
16 coeffs{1} = HcLrA;
    coeffs{2} = LcHrA;
18 coeffs{3} = HcHrA;
    % set the coefficients
20 C{i}=coeffs;

```



```

end
22 C{nb_scales+1} = A;

```

1.2.2 2D reconstruction



```

%% 2D simple wavelet reconstruction
2 function A = recWave2D(LcLrA, HcLrA, LcHrA, HcHrA, lr, hr)
% Reconstruction of an image from lr and hr filters and from the wavelet
% decomposition.
4 % A: resulting (reconstructed) image
%
6 % NB: This algorithm supposes the number of pixels in x and y dimensions
% is
% a power of 2.
8
[ sx, sy ] = size(LcLrA);
10
%% Allocate temporary matrices
12 LrA = zeros(sx*2, sy);
HrA = zeros(sx*2, sy);
14 A = zeros(sx*2, sy*2);

16 %% Reconstruct from cols
for j=1:sy,
18     LrA(:,j) = waveSingleRec(LcLrA(:,j), HcLrA(:,j), lr, hr);
HrA(:,j) = waveSingleRec(LcHrA(:,j), HcHrA(:,j), lr, hr);
20 end

22 %% Reconstruct from rows
for i=1:sx*2,
24     A(i,:) = waveSingleRec(LrA(i,:), HrA(i,:), lr, hr);
end
26

%% Display reconstructed image
28 figure();
imshow(A, []);

```



```

1 function A = simpleImageRec(C)
% wavelet reconstruction of an image described by the wavelet
% coefficients C
3
% The Haar wavelet is used
5 ld = [1 1];
hd = [-1 1];

```



```

7 lr = ld/2;
  hr = -hd/2;

9
  nb_scales = length(C)-1;
11 A = C{nb_scales+1};
  for i=nb_scales:-1:1
13     A = recWave2D(A, C{i}{1}, C{i}{2}, C{i}{3}, lr, hr);
  end

```

1.2.3 Results

The illustration Fig. 2 is obtained by the following code. The useful functions are presented below.



```

1 >> I = imread('lena256.png');
  >> C = simpleImageDec(I, 3);
3 >> A = displayImageDec(C);
  >> imwrite(uint8(255*A), 'lena_3lvl.png');

```



```

function [ A ] = displayImageDec( C )
2 % Construct a single image from a wavelet decomposition
  % C: the decomposition
4 % A: the entire illustration image

6 n_scales = length(C)-1;
  [n, m] = size(C{1}{1});
8 A = zeros(2*n, 2*m);

10 prev = C{n_scales+1};
  for s=n_scales:-1:1
12     ns = n / 2^(s-2);
     ms = m / 2^(s-2);
14     A(1:ns, 1:ms) = imdec2im(prev, C{s});
     prev = A(1:ns, 1:ms);
16 end

18 end

```



```

function A = imdec2im(LcLrA, lvlC)
2 % constructs a single image from:
  % LcLrA: the approximation image

```



```

4 % lvlC: the wavelet decomposition at one level
%
6 % for display purposes

8 HcLrA=lvlC{1};
  LcHrA=lvlC{2};
10 HcHrA=lvlC{3};
   [n, m] = size(HcLrA);
12
   A = zeros(2*n, 2*m);
14
   % approximation image can be with high values when using Haar
   %   ↪ coefficients
16 A(1:n, 1:m) = LcLrA / max(LcLrA(:));

18 % details are low, and can be negative
   A(1:n, m+1:2*m) = imadjust(HcLrA, stretchlim(HcLrA), [0 1]);
20 A(n+1:2*n, 1:m) = imadjust(LcHrA, stretchlim(LcHrA), [0 1]);
   A(n+1:2*n, m+1:2*m) = imadjust(HcHrA, stretchlim(HcHrA), [0 1]);
22
   imshow(A)

```

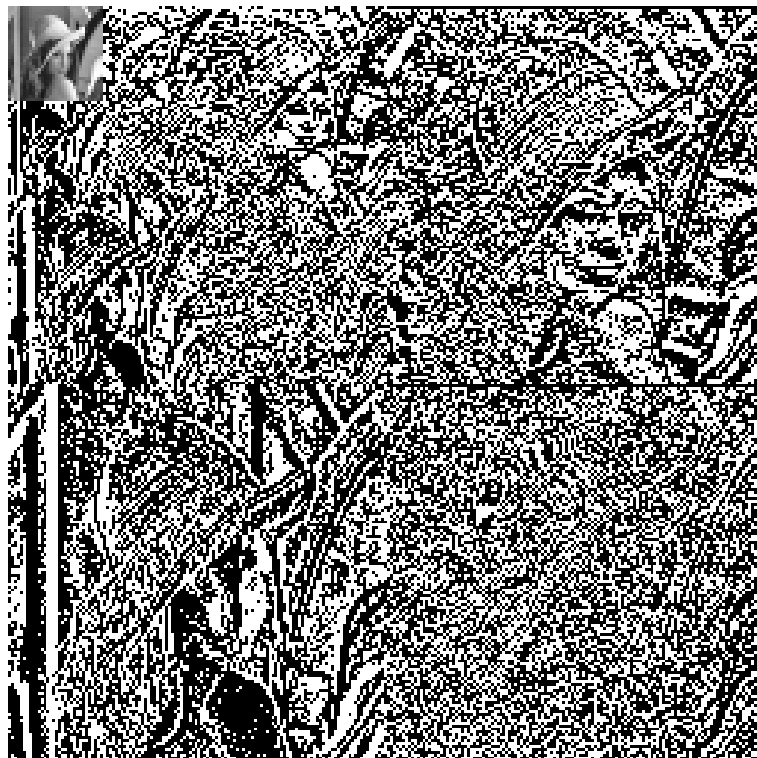


Figure 2: (Haar) Wavelet decomposition of the Lena image.

1.3 Matlab functions

1.3.1 1D

Example on a sample function:



```
1 % signal
  t=0:.001:10;
3 signal = sin(2*pi*3*t)+.2*sin(2*pi*50*t);

5 % parameters
  wavelet = 'db1'; % Daubechies wavelets
7 lvl = 5; % decomposition level

9 % Decomposition
  [C, S] = wavedec(signal, lvl, wavelet);
11
  % Reconstruction
13 srec = waverec(C, S, wavelet);
```

1.3.2 2D



```
1 % read an image
  I = imread('Couche_18.png');
3 lvl = 8;

5 % decomposition
  [C, S] = wavedec2(I, lvl, 'db4');
7
  % threshold after lvl
9 newC = wthcoef2('a', C, S);

11 % reconstruction
  I2 = waverec2(newC, S, 'db4');
13
  % display result
15 imshow(I, []);

17 figure(); imshow(I2, []);
```