Transform Methods & Signal Processing

lecture 11

Matthew Roughan
<matthew.roughan@adelaide.edu.au>

Discipline of Applied Mathematics
School of Mathematical Sciences
University of Adelaide

December 4, 2009

2D Wavelets

Generalizing Wavelets to 2D is not quite as simple as generalizing the Fourier transform to higher dimensions.

We extend Wavelets to 2D

Transform Methods & Signal Processing (APP MTH 4043): lecture 11 – p.1/27

Transform Methods & Signal Processing (APP MTH 4043): lecture 11 – p.2/27

2D

Simple extension (for separable wavelet bases)

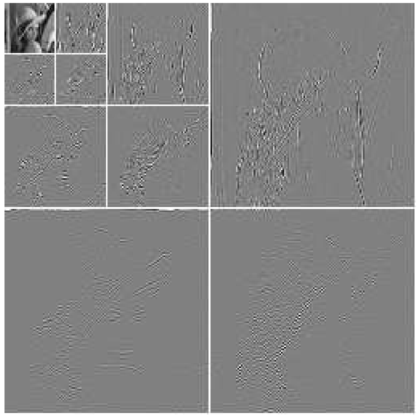


Figure 2.23 from Mallat (p.310), made using WaveLab http://www-stat.stanford.edu/~wavelab/

```
% file:       lena.m, (c) Matthew Roughan, Mon Aug 14 2006
%
Image = ReadImage('Lenna');
figure(1)
imagesc(Image);
colormap(gray);
opts = struct('height',8, 'Color', 'gray');
axis image
axis('off')
set(gca,'DataAspectRatio', [1 1 1]);
exportfig(gcf,sprintf('Plots/lena.eps'), opts, 'format', 'eps');

[n,J] = quadlength(Image);
qmf = MakeONFilter('Daubechies',8);
L = 5;
wc = FWT2_PO(Image,L,qmf);

figure(3)
Display2dProjV(wc,L,qmf);
axis image
axis('off')
set(gca,'DataAspectRatio', [1 1 1]);
exportfig(gcf,sprintf('Plots/lena_approx.eps'), opts, 'format', 'eps');

% taken from wt07fig26 (Mallat Fig 7.26)
wc2 = wc;
avg = wc(1:2^L,1:2^L);
wc2(1:2^L,1:2^L) = 1-(avg./max(max(abs(avg))));
wc2(1:2^L,2^L) = zeros(2^L,1);
wc2(2^L,1:2^L) = zeros(1,2^L);
```

Transform Methods & Signal Processing (APP MTH 4043): lecture 11 – p.3/27

Separable wavelet bases

Take any orthonormal wavelet basis $\{\psi_{n,j}\}_{n,j\in\mathbb{Z}}$ of $L^2(\mathbb{R})$, then a separable wavelet basis for $L^2(\mathbb{R}^2)$ is

$$\{\psi_{n_1,j_1}\psi_{n_2,j_2}\}_{n_1,n_2,j_1,j_2\in\mathbb{Z}}$$

- but basis above mixes resolutions at different scales j_1 and j_2
- separable MRAs lead to constructions that are products of functions dilated to the same scale
- can construct non-separable bases, but used less often
- build approximation spaces $V_j^2 = V_j \otimes V_j$ such that these are separable, i.e., basis looks like $\phi^2_{n_1,n_2,j}(x_1,x_2) = \phi_{n_1,j}(x_1)\phi_{n_2,j}(x_2)$

Transform Methods & Signal Processing (APP MTH 4043): lecture 11 – p.4/27

2D scaling functions

Scaling functions

$$\phi^2_{n_1,n_2,j}(x_1,x_2) = \phi_{n_1,j}(x_1)\phi_{n_2,j}(x_2) = \frac{1}{2^j}\phi\left(\frac{x_1}{2^j} - n_1\right)\phi\left(\frac{x_2}{2^j} - n_2\right)$$

Approximation $\hat{f}_j = \sum_{n_1,n_2,j} \left\langle f, \phi^2_{n_1,n_2,j} \right\rangle \phi^2_{n_1,n_2,j}$ where

$$\left\langle f, \phi^2_{n_1,n_2,j} \right\rangle = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x_1,x_2)\phi^2_{n_1,n_2,j}(x_1,x_2)\, dx_1\, dx_2$$

which separates into two integrals if f separates.

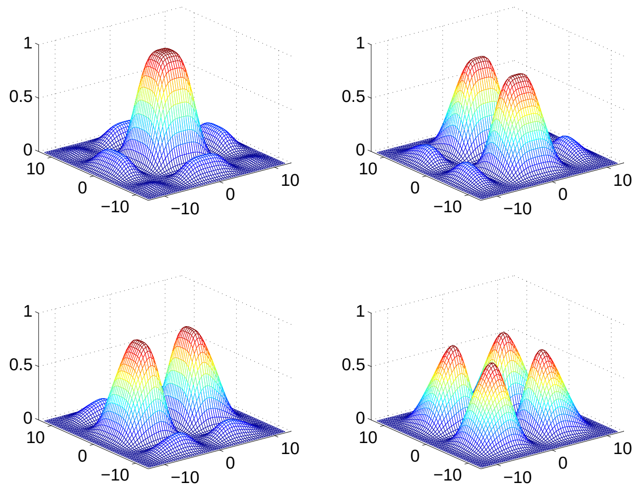
2D Wavelets

We get 3 mother wavelets

$$\begin{aligned} \psi^1(x_1,x_2) &= \phi(x_1)\psi(x_2) \\ \psi^2(x_1,x_2) &= \psi(x_1)\phi(x_2) \\ \psi^3(x_1,x_2) &= \psi(x_1)\psi(x_2) \end{aligned}$$

from which we derive the wavelets by dilation and 2D translations.

2D Wavelet filter spectrum



2D MRA tree

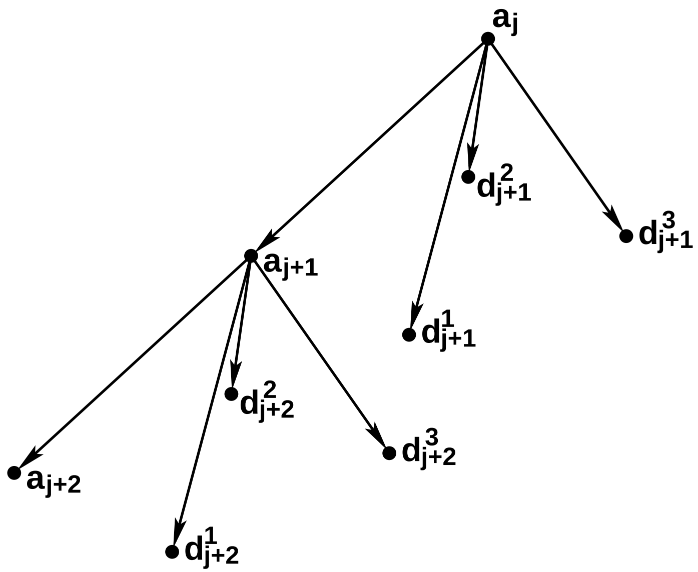


Figure 2.24 from Mallat (p.308), made using WaveLab `http://www-stat.stanford.edu/~wavelab/`

2D Wavelet Filters

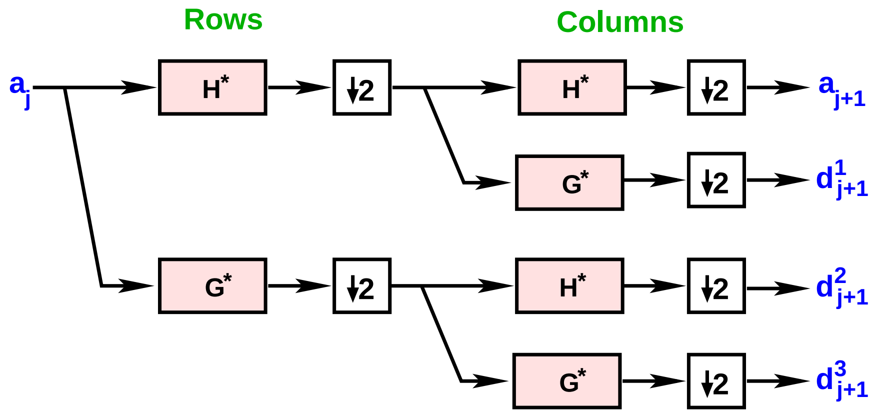
$$a_j(n_1,n_2) = \langle f, \phi_{n1,n2,j} \rangle \quad \text{and} \quad d_j^k(n_1,n_2) = \langle f, \psi^k_{n1,n2,j} \rangle$$

one step of the decomposition takes the form

$$a_{j+1}(n_1,n_2) = \left[a_j * \overline{hh}\right](2n_1,2n_2)$$
$$d^1_{j+1}(n_1,n_2) = \left[a_j * \overline{h}\overline{g}\right](2n_1,2n_2)$$
$$d^2_{j+1}(n_1,n_2) = \left[a_j * \overline{g}\overline{h}\right](2n_1,2n_2)$$
$$d^3_{j+1}(n_1,n_2) = \left[a_j * \overline{gg}\right](2n_1,2n_2)$$

- Notation $\overline{h}(n) = h(-n)$
- Product hg means $[hg](n_1,n_2) = h(n_1)g(n_2)$
- 2D convolution can be performed as two 1D convolutions (downsample between doing rows and columns)

2D Wavelet Block Diagram



Transform Methods & Signal Processing (APP MTH 4043): lecture 11 – p.9/27

Transform Methods & Signal Processing (APP MTH 4043): lecture 11 – p.10/27

2D Wavelet Reconstruction

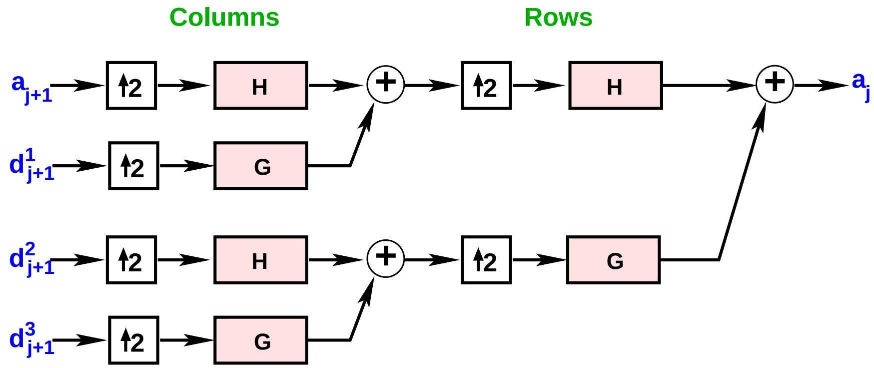
Representation $\{a_J, \{d_j^1, d_j^2, d_j^3\}_{L<j\leq j}\}$

- define upsampled 2D image $\breve{y}(n_1, n_2)$ made by inserting rows and columns of zeros in between existing rows and columns

Reconstruction

$$a_j = \breve{a}_{j+1} * hh + \breve{d}^1{}_{j+1} * hg + \breve{d}^2{}_{j+1} * gh + \breve{d}^3{}_{j+1} * gg$$

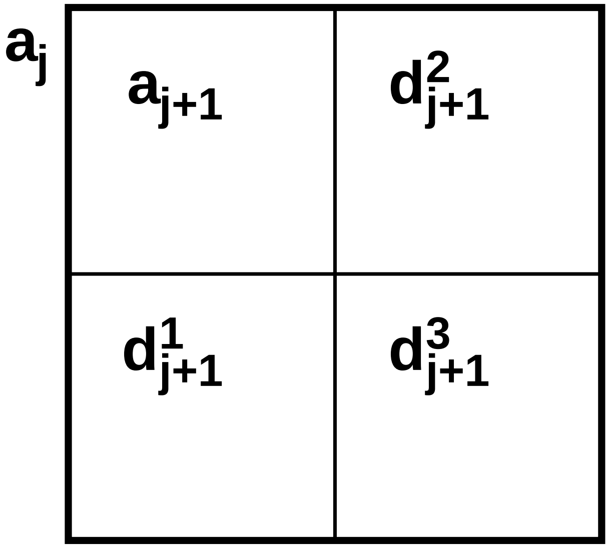
2D Wavelet Block Diagram



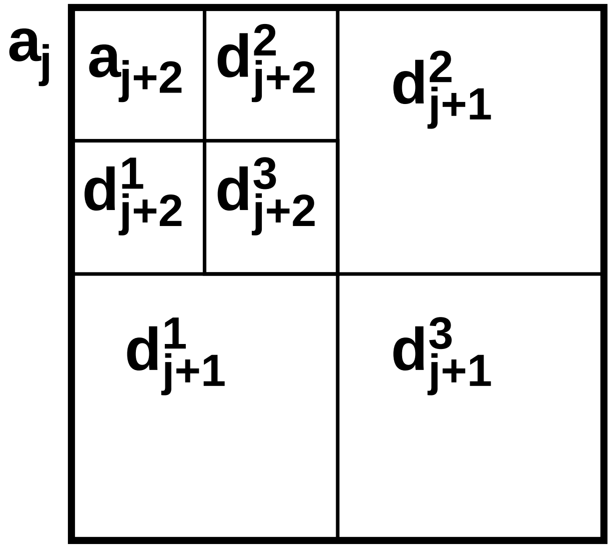
Transform Methods & Signal Processing (APP MTH 4043): lecture 11 – p.11/27

Transform Methods & Signal Processing (APP MTH 4043): lecture 11 – p.12/27

2D Layout



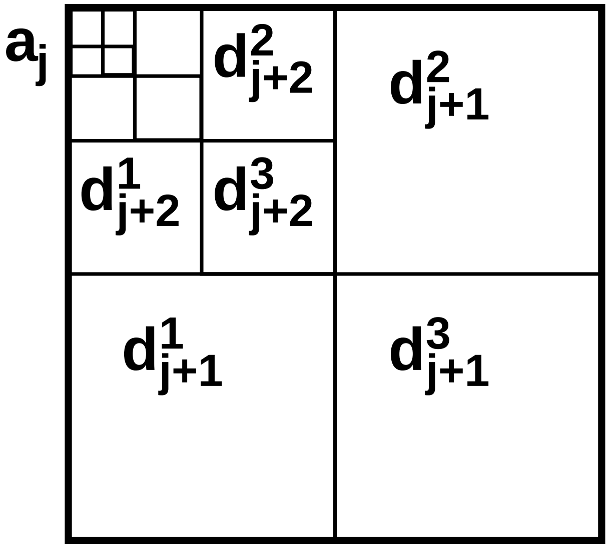
2D Layout



Transform Methods & Signal Processing (APP MTH 4043): lecture 11 – p.13/27

Transform Methods & Signal Processing (APP MTH 4043): lecture 11 – p.14/27

2D Layout



2D

Simple extension (for separable wavelet bases)

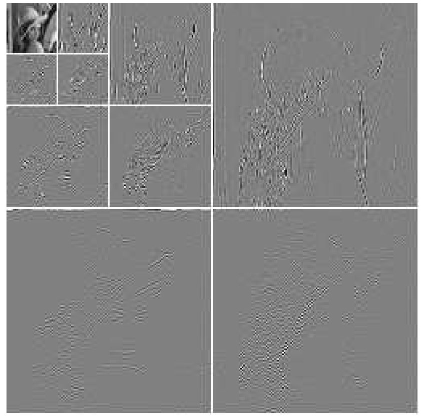
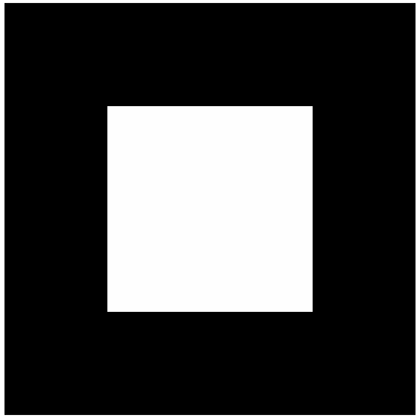


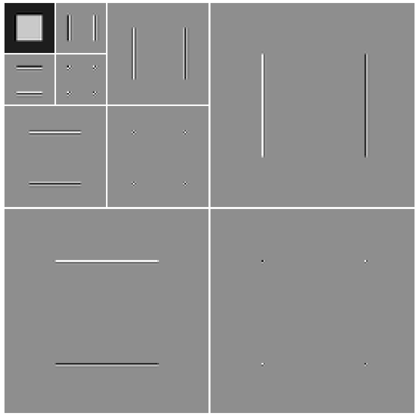
Figure 2.23 from Mallat (p.310), made using WaveLab http://www-stat.stanford.edu/~wavelab/

```
% file:       lena.m, (c) Matthew Roughan, Mon Aug 14 2006
%
Image = ReadImage('Lenna');
figure(1)
imagesc(Image);
colormap(gray);
opts = struct('height',8, 'Color', 'gray');
axis image
axis('off')
set(gca,'DataAspectRatio', [1 1 1]);
exportfig(gcf,sprintf('Plots/lena.eps'), opts, 'format', 'eps');

[n,J] = quadlength(Image);
qmf = MakeONFilter('Daubechies',8);
L = 5;
wc = FWT2_PO(Image,L,qmf);

figure(3)
Display2dProjV(wc,L,qmf);
axis image
axis('off')
set(gca,'DataAspectRatio', [1 1 1]);
exportfig(gcf,sprintf('Plots/lena_approx.eps'), opts, 'format', 'eps');

% taken from wt07fig26 (Mallat Fig 7.26)
wc2 = wc;
avg = wc(1:2^L,1:2^L);
wc2(1:2^L,1:2^L) = 1-(avg./max(max(abs(avg))));
wc2(1:2^L,2^L) = zeros(2^L,1);
wc2(2^L,1:2^L) = zeros(1,2^L);
```

2D

Simple extension (for separable wavelet bases)





Applications

Figure 2.23 from Mallat (p.310), made using WaveLab `http://www-stat.stanford.edu/~wavelab/`

```
% file:       wavelet_box.m, (c) Matthew Roughan, Mon Aug 14 2006
%
N = 256;
Image = MakeImage('Square',N);
figure(1)
imagesc(Image);
colormap(gray);
opts = struct('height',8, 'Color', 'gray');
axis image
axis('off')
set(gca,'DataAspectRatio', [1 1 1]);
exportfig(gcf,sprintf('Plots/wavelet_box.eps'), opts, 'format', 'eps');

[n,J] = quadlength(Image);
qmf = MakeONFilter('Daubechies',8);
L = 5;
wc = FWT2_PO(Image,L,qmf);

figure(3)
Display2dProjV(wc,L,qmf);
axis image
axis('off')
set(gca,'DataAspectRatio', [1 1 1]);
exportfig(gcf,sprintf('Plots/wavelet_box_approx.eps'), opts, 'format', 'eps');

% taken from wt07fig26 (Mallat Fig 7.26)
wc2 = wc;
avg = wc(1:2^L,1:2^L);
wc2(1:2^L,1:2^L) = 1-(avg./max(max(abs(avg))));
wc2(1:2^L,2^L) = zeros(2^L,1);
```

Compression: FBI fingerprints

- FBI have $\sim$ 30 million fingerprints
  - actually more like 200 million (repeats etc)
  - 30-50,000 more per day
- 1993 started converting from ink on cards (transmitted by fax) to digital storage
- 500 pixels per inch resolution. 256 grey levels (8 bits)
  - one fingerprint, 700,000 pixels, and 6 MB storage
  - 200 TB for whole database
  - have to transmit cards (3 hours on slow modem)
- compression is needed

Compression: FBI fingerprints

Basic idea, quantize in the transform space.

- use Wavelet transform (in 2D)
- steps
  - wavelet transform
  - quantize coefficients
  - entropy encoding
- called WSQ (Wavelet Scalar Quantization)

Links:
http://www.c3.lanl.gov/~brislawn/FBI/FBI.html
http://www.amara.com/IEEEwave/IW_fbi.html
ftp://ftp.c3.lanl.gov/pub/misc/WSQ/FBI_WSQ_FAQ

Transform Methods & Signal Processing (APP MTH 4043): lecture 11 – p.19/27

Transform Methods & Signal Processing (APP MTH 4043): lecture 11 – p.20/27

Compression: FBI fingerprints

Compression: FBI fingerprints

Original, file size 589,824 bytes.

JPEG, file size 45853 bytes, compression ratio 12.9.

http://www.c3.lanl.gov/~brislawn/FBI/FBI.html

Compression: FBI fingerprints

Original, file size 589,824 bytes.

Wavelets, compression ratio 12.9.

http://www.c3.lanl.gov/~brislawn/FBI/FBI.html

Compression

Compression

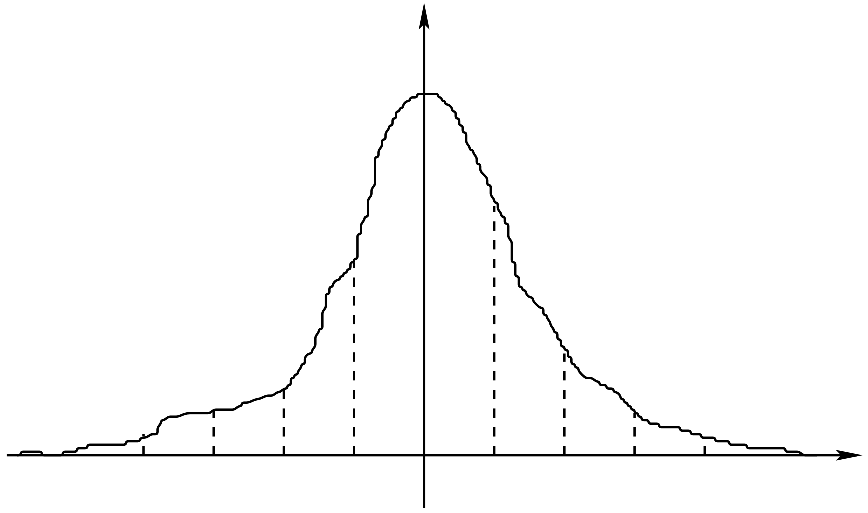
- Simplest version – just perform algorithm above
- more general: quantize wavelet coefficients by a fixed step

$$\hat{d}(j,k) = Q\text{sign}(d(j,k)) \left\lfloor \frac{|d(j,k)|}{Q} \right\rfloor$$

- more general: use a quantization table
- Inverse Wavelet Transform of $\{\hat{d}(j,k)\}_{j,k}$

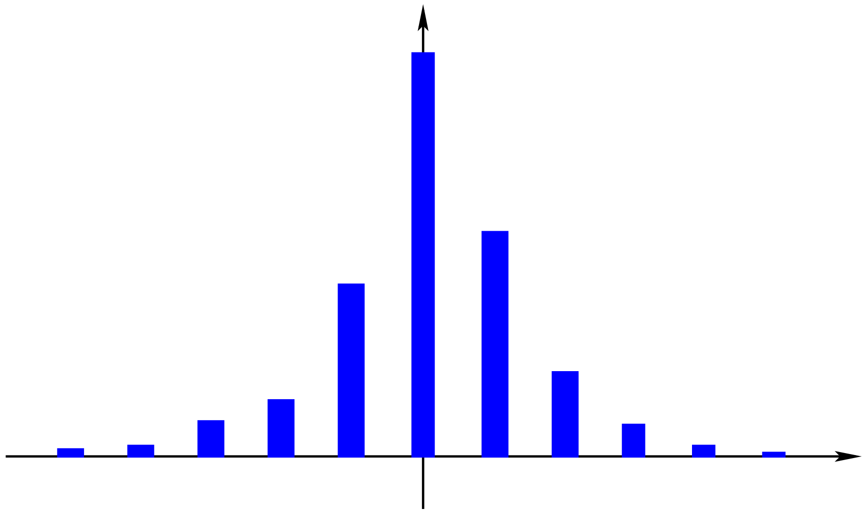
Quantization

We start with some continuous distribution



Quantization

Then quantize the distribution into a number of levels



JPEG 2000

JPEG 2000 uses wavelets rather than the DCT

Links:
http://www.jpeg.org/jpeg2000/
http://www.gvsu.edu/math/wavelets/student_work/Miljour/

Some comparisons
http://www.imagepower.com/technology/jpeg2000/compare/
http://www.levien.com/gimp/jpeg2000/comparison.html
http://ai.fri.uni-lj.si/~aleks/jpeg/artifacts.htm
http://www.fnordware.com/j2k/jp2samples.html