Lecture notes of Image Compression and Video Compression

4. Introduction to Wavelet

Topics

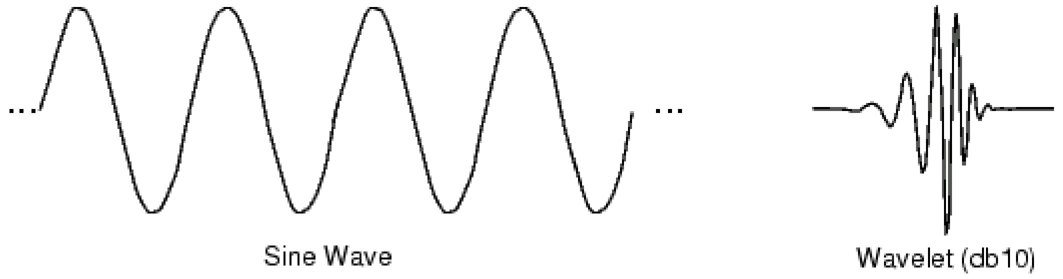
- Introduction to Image Compression
- Transform Coding
- Subband Coding, Filter Banks
- Introduction to Wavelet Transform
- Haar, SPIHT, EZW
- Motion Compensation
- Wireless Video Compression

Contents

Wavelet Definition

“The wavelet transform is a tool that cuts up data, functions or operators into different frequency components, and then studies each component with a resolution matched to its scale”

---- Dr. Ingrid Daubechies, Lucent, Princeton U

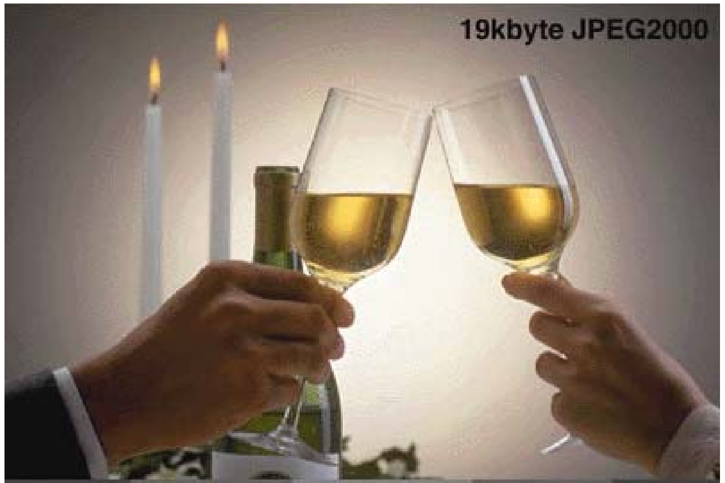


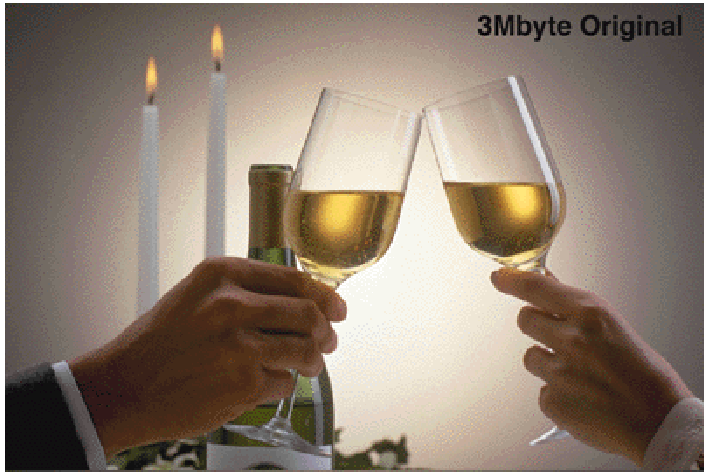
Wavelet Coding Methods

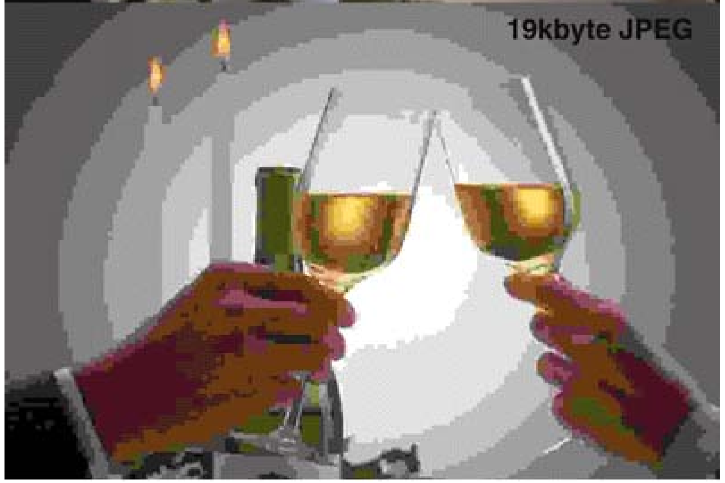
- EZW - Shapiro, 1993
  - Embedded Zerotree coding.
- SPIHT - Said and Pearlman, 1996
  - Set Partitioning in Hierarchical Trees coding. Also uses “zerotrees”.
- ECECOW - Wu, 1997
  - Uses arithmetic coding with context.
- EBCOT – Taubman, 2000
  - Uses arithmetic coding with different context.
- JPEG 2000 – new standard based largely on EBCOT
- GTW – Hong, Ladner 2000
  - Uses group testing which is closely related to Golomb codes
- UWIC - Ladner, Askew, Barney 2003
  - Like GTW but uses arithmetic coding

Comparison of Wavelet Based JPEG 2000 and DCT Based JPEG

- JPEG2000 image shows almost no quality loss from current JPEG, even at 158:1 compression.







Introduction to Wavelets

- *"... the new computational paradigm [wavelets] eventually may swallow Fourier transform methods..."*

- *" ...a new approach to data crunching that, if successful, could spawn new computer architectures and bring about commercial realization of such difficult data-compression tasks as sending images over telephone lines. "*

---- from "New-wave number crunching" C. Brown, Electronic Engineering Times, 11/5/90.

Early History of Wavelet Theory

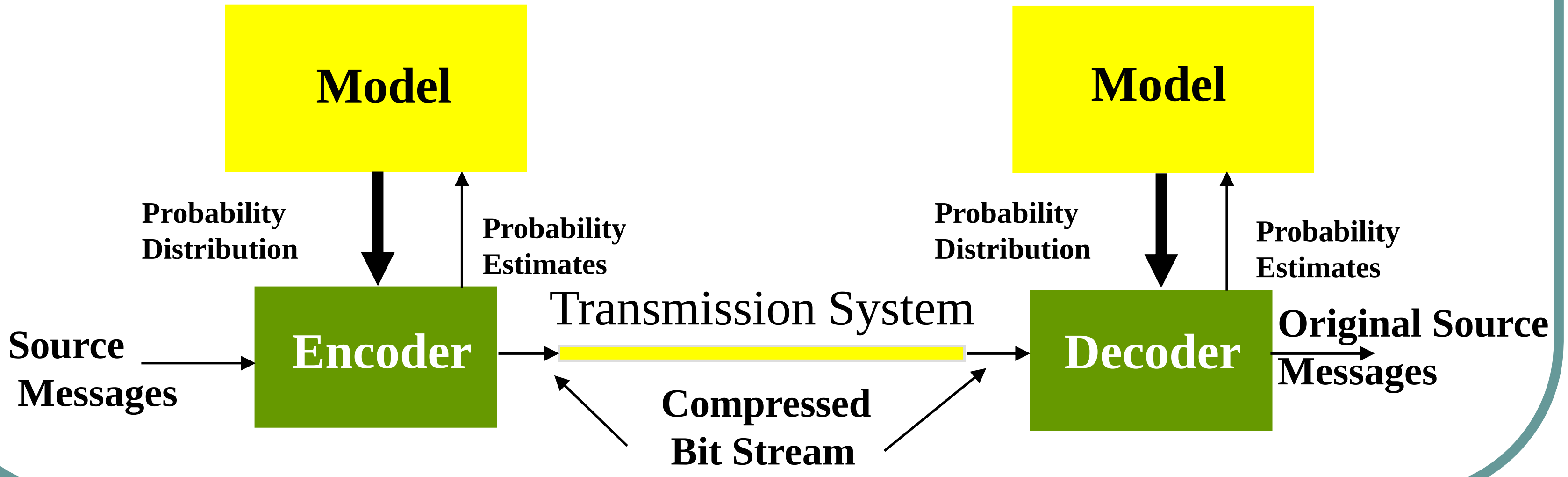
- Roots found in a variety of disciplines
  - Mathematics, Signal Processing, Computer Vision, Physics.
- 1910 Haar basis
  - First wavelet.
- 1946 The Gabor transform
  - Short time Fourier transform with Gaussian window function.
- 1964 Calderon's work on singular integral operators
  - Contains the continuous wavelet transform.
- 1971 A. Rosenfeld and M. Thurston
  - Multi-resolution techniques invented in machine vision
  - Multi-resolution schemes inherent in the wavelet transform.
- 1976 A. Croiser, D. Estaban, C. Galand
  - Quadrature mirror filter banks for speech coding
  - Digital implementation of wavelets.

Recent History of Wavelets

- 1984 J. Morlet and A. Grossman
  - “Invent“ term wavelets
  - Apply them to the analysis of seismic signals
- 1985 Meyer
  - tried to prove that no orthogonal wavelet other than Haar exists, found one by trial and error!
- 1987 Mallat
  - Developed multiresolution theory, DWT, wavelet construction techniques (but still noncompact)
- 1988 I. Daubechies
  - Found compact, orthogonal wavelets with arbitrary number of vanishing moments!
- 2001 wavelet based JPEG2000 finalized.

Model and Prediction

- Compression is **PREDICTION**.
- There are many **decomposition** approaches to modeling the signal.
  - Every signal is a function.
  - Modeling is function representation/approximation.



Methods of Function Approximation

- Sequence of samples
  - Time domain
- Pyramid (hierarchical)
- Polynomial
- Piecewise polynomials
  - Finite element method
- Fourier Transform
  - Frequency domain
  - Sinusoids of various frequencies
- Wavelet Transform
  - Time/frequency domain

The Fourier Transform

- Analysis, forward transform:

$$F(u) = \int f(t) e^{-j2\pi ut} dt$$

- Synthesis, inverse transform:

$$f(t) = \int F(u) e^{j2\pi ut} du$$

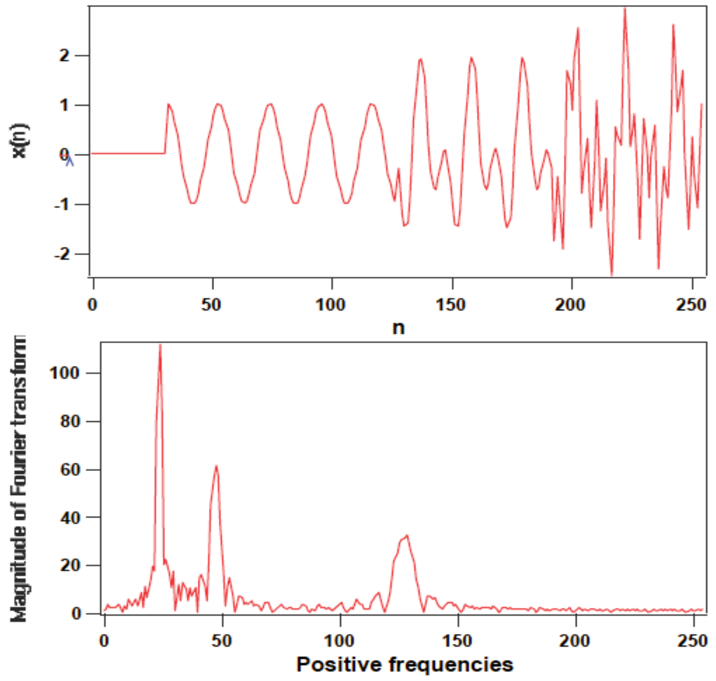
- Forward transform decomposes f(t) into sinusoids.
  - F(u) represents how much of the sinusoid with frequency u is in f(t).
- Inverse transform synthesizes f(t) from sinusoids, weighted by F(u).

The Fourier Transform Properties

- Linear Transform.
- Analysis (decomposition) of signals into sines and cosines has physical significance
  - tones, vibrations.
- Fast algorithms exist.
  - The fast Fourier transform requires O(nlogn) computations.

Problems With the Fourier Transform

- Fourier transform well-suited for stationary signals - signals that do not vary with time. This model does not fit real signals well.
- For time-varying signals or signals with abrupt transitions, the Fourier transform does not provide information on when transitions occur.



Problems With the Fourier Transform

- Fourier transform is a “global” analysis. A small perturbation of the function at any one point on the time-axis influences all points on the frequency-axis and vise versa.
- If a signal is received correctly for hours and gets corrupted for only a few second, it totally destroys the signal.
- The lack of time information makes Fourier transform error prone.

Problems With the Fourier Transform

- A qualitative explanation of why Fourier transform fails to capture time information is the fact that the set of basis functions (sines and cosines) are infinitely long and the transform picks up the frequencies regardless of where it appears in the signal.
- Need a better way to represent functions that are localized in both time and frequency.

Uncertainty Principle --- Preliminaries for the STFT

- The time and frequency domains are complimentary.
  - If one is local, the other is global.
  - For an impulse signal, which assumes a constant value for a very brief period of time, the frequency spectrum is infinite.
  - If a sinusoidal signal extends over infinite time, its frequency spectrum is a single vertical line at the given frequency.
- We can always localize a signal or a frequency but we cannot do that simultaneously.
  - If the signal has a short duration, its band of frequency is wide and vice versa.

Uncertainty Principle

- Heisenberg's uncertainty principle was enunciated in the context of quantum physics which stated that the position and the momentum of a particle cannot be precisely determined simultaneously.
- This principle is also applicable to signal processing.

Uncertainty Principle -- In Signal Processing

Let *g*(*t*) be a function with the property .

$$\int_{-\infty}^{\infty} \left| g(t)^2 \right| dt = 1$$

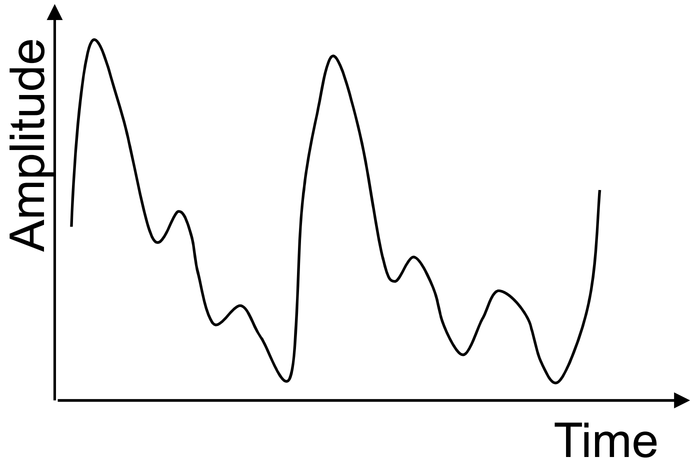
Then

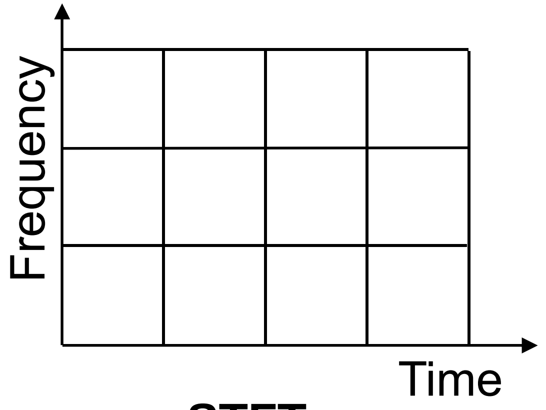
$$\left(\int_{-\infty}^{\infty} (t - t_m)^2 \left| g(t) \right|^2 dt\right)\left(\int_{-\infty}^{\infty} (f - f_m)^2 \left| G(f) \right|^2 dt\right) \geq \frac{1}{16\Pi^2}$$

where t_m, f_m denote average values of *t* and *f* ,and *G*(*f*) is the Fourier transform of *g*(*t*).

Gabor's Proposal Short-time Fourier Transform.

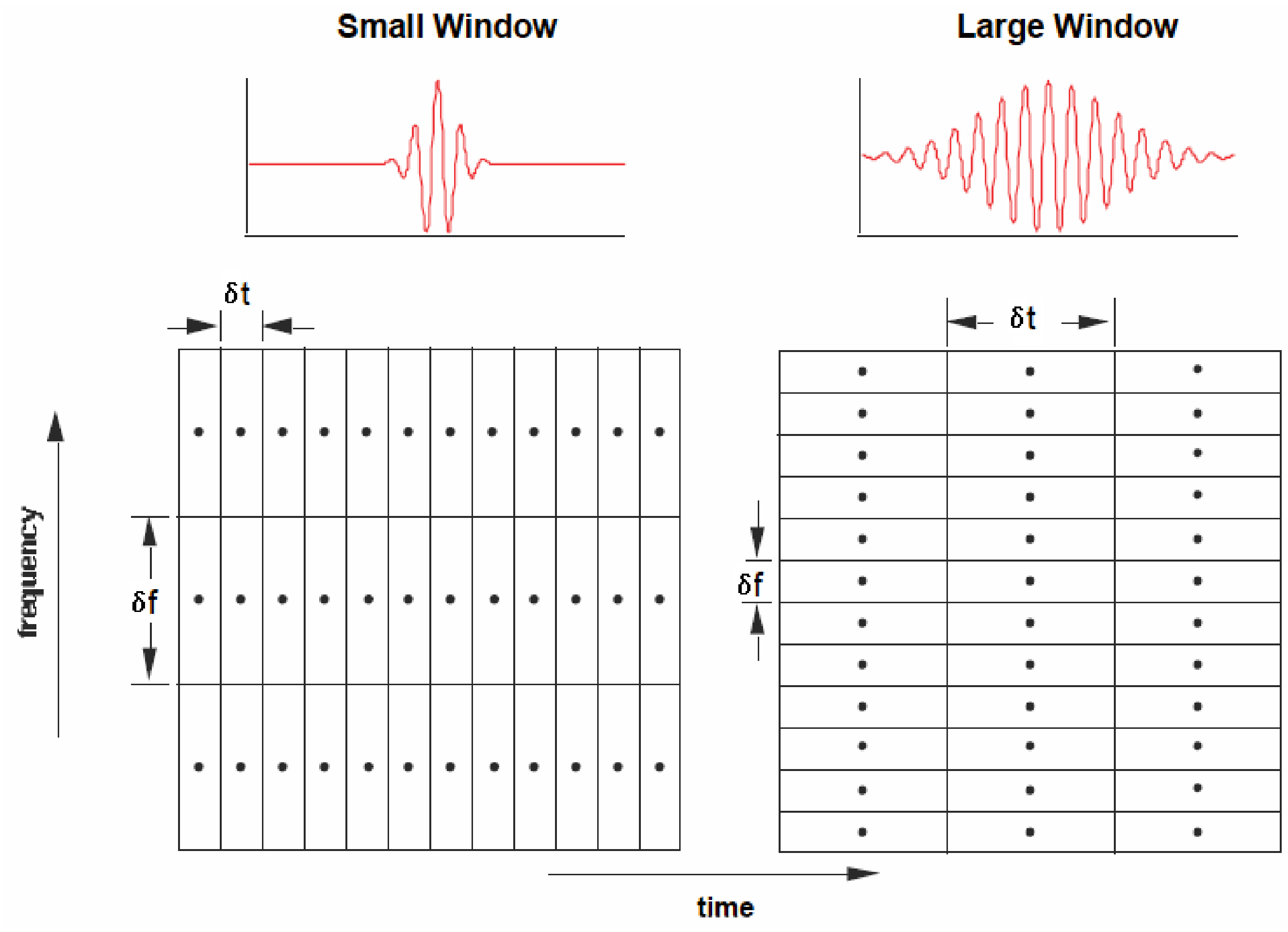
- The STFT is an attempt to alleviate the problems with FT.
- It takes a non-stationary signal and breaks it down into "windows" of signals for a specified short period of time and does Fourier transform on the window by considering the signal to consist of repeated windows over time.





STFT

The Short-time Fourier Transform Time-frequency Resolution



The Short-time Fourier Transform

- Analysis:

$$STFT(\tau,u)=\int f(t)w^*(t-\tau)e^{-j2\pi ut}dt$$

- Synthesis:

$$f(t)=\int STFT(\tau,u)w(t-\tau)e^{j2\pi ut}d\tau du$$

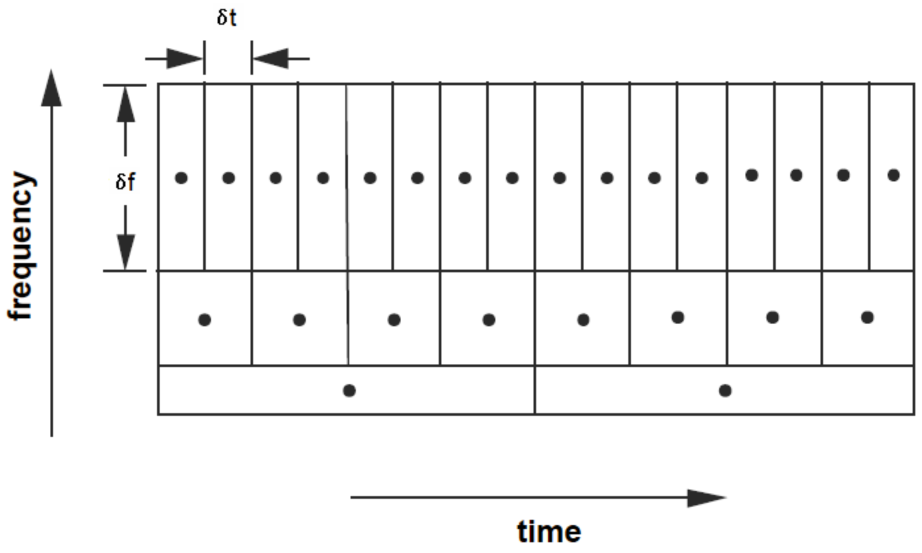
- where w(t) is a window function localized in time and frequency.
- Analysis functions are sinusoids windowed by w(t).
- Common window functions
  - Gaussian (Gabor), Hamming, Hanning, Kaiser.

The Short-time Fourier Transform Properties

- Linear transform.
- Time resolution (Δt) and frequency resolution (Δu) are determined by w(t), and remain fixed regardless of τor u.
- Biggest disadvantage:
  - since Δt and Δu are fixed by choice of w(t), need to know a priori what w(t) will work for the desired application.

Basic Idea of the Wavelet Transform -- Time-frequency Resolution

- Basic idea:
  - Δt, Δu vary as a function of scale (scale = 1/frequency).



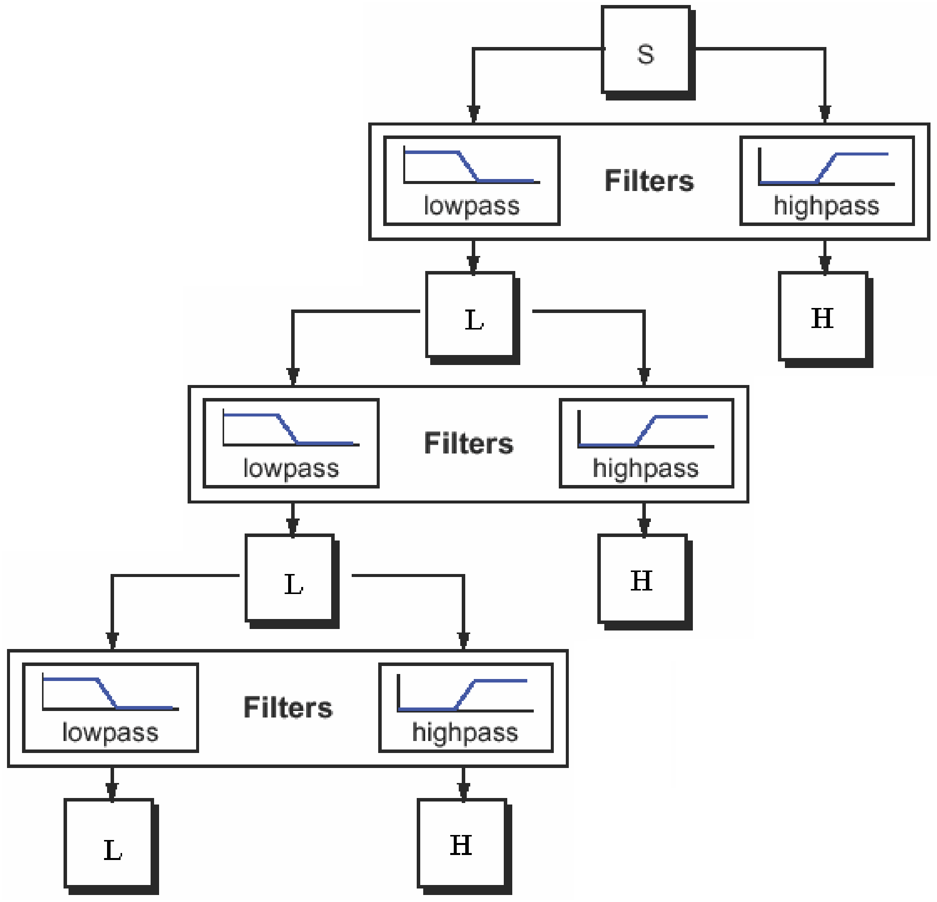
One-dimensional Haar Wavelet Transform

- Given input value {1, 2, 3, 4, 5, 6, 7, 8} (resolution 8)
- Step #1 (resolution 4)
  - Output Low Frequency {1.5, 3.5, 5.5, 7.5} - averages. The averages can be used to approximate the function with some loss of information. To recapture the lost information, we need the differences – the so-called detailed coefficients.
  - Output High Frequency {-0.5, -0.5, -0.5, -0.5} – **detail coefficients**.

  Now, Average +detail= first coefficient and Average-detail= second coefficient of the pair.
- Step #2 (resolution 2)
  - Refine Low frequency output in Step #1
    - L: {2.5, 6.5}- average
    - H: {-1, -1} - detail
- Step #3 (resolution1)
  - Refine Low frequency output in Step #2
    - L: {4.5} -average
    - H: {-2} - detail
    - Transmit { 4.5, -2, -1, -1, -0.5, -0.5, -0.5, -0.5}. No information has been lost or gained by this process. We can reconstruct the original image from this vector by adding and subtracting the detail coefficients. The vector has 8 values, as in the original sequence, but except for the first coefficient, all have small magnitudes. This vector is called the ***wavelet transform*** or the **wavelet decomposition** of the original sequence. As we will see soon,we have used the **Haar basis** for this one-dimensional transfor.

Sub-band Interpretation of Wavelet Transform

- The computation of the wavelet transform used recursive averaging and differencing coefficients. It behaves like a **filter bank**.
- Recursive application of a two-band filter bank to the lowpass band of the previous stage.



Properties of the Transform

- We can reconstruct the vector to any resolution by recursively adding and subtracting the detail coefficients.
- In practice, a large number of detail coefficients become very small. We can simply drop or truncate them. The reconstruction will be “lossy” but for most image applications this approach is good enough.

Function and Vector Space

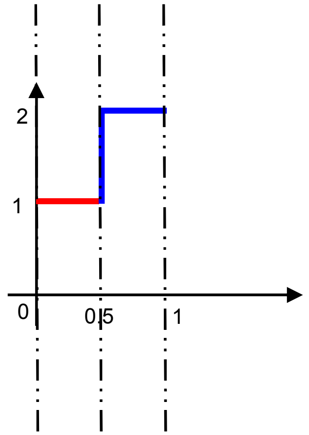
- Instead of thinking the pixels as coefficients, we may think of them as functions. For example, we can think of a one-pixel image to be a function that is constant over the entire interval [0,1). Since addition and multiplication operations are well defined for such functions, we can think of such a function to be a vector in the vector space V^0.
- A 2-pixel image is then a function with two constant pieces over intervals [0,1/2) and [1/2,1). The vector space containing all such functions is denoted as V^1 and so on .
- The vector space V^j include all piecewise constant functions defined on the interval [0,1) with constant pieces over each of 2^j equal sized subintervals. Thus, a one-dimensional image with resolution 2^j is an element of the vector space V^j.
- These vector spaces are nested.

Function and Vector Space

- Every vector space V^j is also contained in V^{j+1}, because a piece-wise constant function with 2^j intervals can be thought of as a piecewise constant function with 2^{j+1} intervals.
- So, $V^1 \subset V^2$
- In general, we have

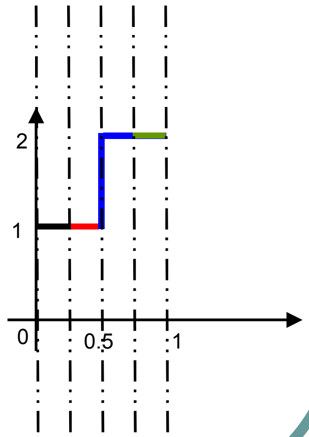
$$V^0 \subset V^1 \subset V^2 \subset \cdots$$

$$f(x) = \begin{cases} 1 & 0 < x < 0.5 \\ 2 & 0.5 < x < 1 \\ 0 & else \end{cases}$$



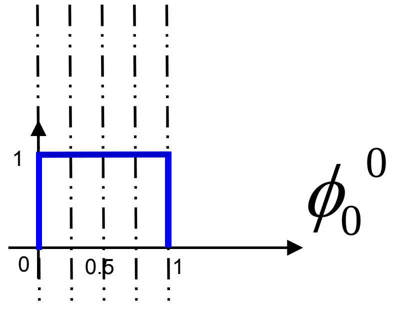
$\|$ Same function!

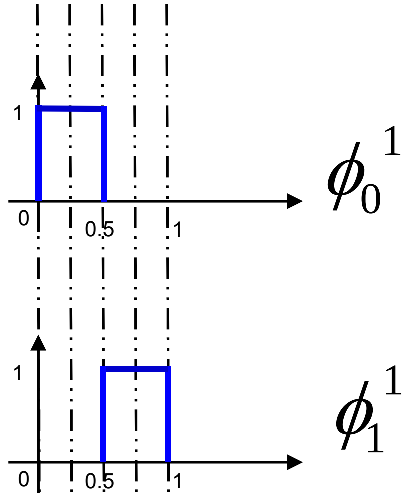
$$f(x) = \begin{cases} 1 & 0 < x < 0.25 \\ 1 & 0.25 < x < 0.5 \\ 2 & 0.5 < x < 0.75 \\ 2 & 0.75 < x < 1 \\ 0 & else \end{cases}$$

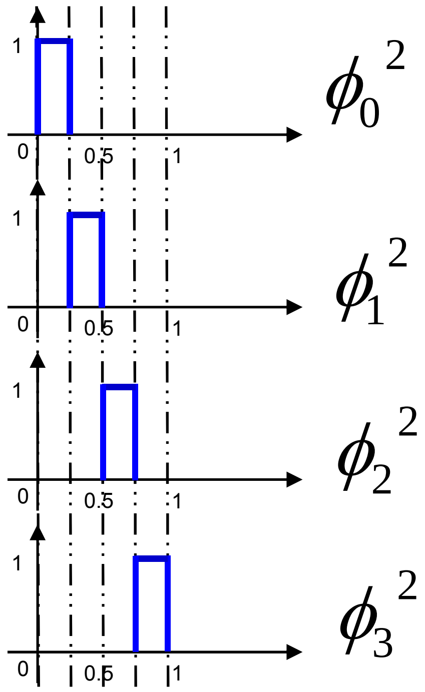


Scaling Function

- The basis function for the space V^j is called scaling function ϕ.







Basis function for V^0 Basis function for V^1 Basis function for V^2

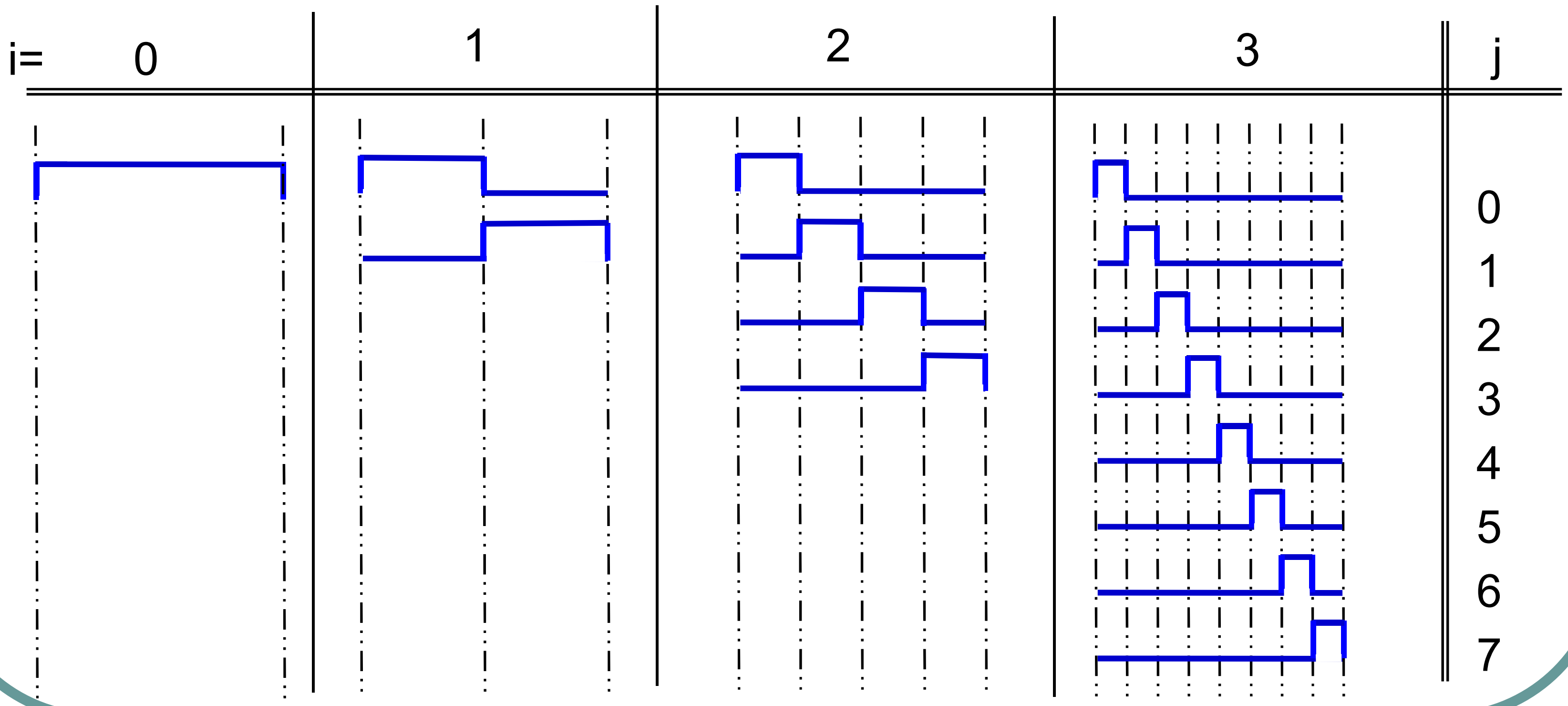
Scaling Function

- In general, a vector space is defined on the set of scaled and translated basis function $\phi_i^j(x) = \phi(2^j x - i), \qquad i = 0, 1, \cdots, 2^j - 1$ where
$$\phi(x) = \begin{cases} 1 & 0 \le x \le 1 \\ 0 & else \end{cases}$$
- The interval on which $\phi_i^j(x)$ equals ‘1’ is called support. For example, the support of $\phi_1^2(x)$ is [1/4, ½).
- Note that $\phi_0^0(x) = \phi(x)$

Scaling Function

- Example of the Haar scaling function

$$\phi_i^j(x) = \phi(2^j x - i), \qquad i = 0, 1, \cdots, 2^j - 1$$



Scaling Functions

- Any sequence of pixels of resolution 2^j can be explained as a linear combination of the box basis function in V^j. For example, for j = 2, an image *f*(*x*)=[9,7,3,3] can be explained as:

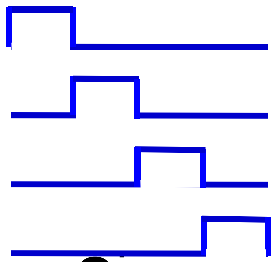
$$f(x) = c_0^{\,2}\phi_0^{\,2}(x) + c_1^{\,2}\phi_1^{\,2}(x)$$
$$+ c_2^{\,2}\phi_2^{\,2}(x) + c_3^{\,2}\phi_3^{\,2}(x)$$
$$\text{where} \quad c_0^{\,2} = 9, \quad c_1^{\,2} = 7,$$
$$c_2^{\,2} = 3, \quad c_3^{\,2} = 3$$

f(x) = 9 *

+ 7 *

+ 3 *

+ 3 *

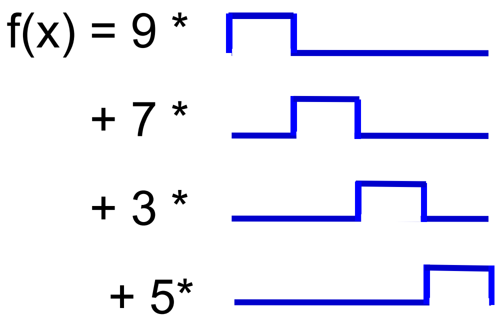


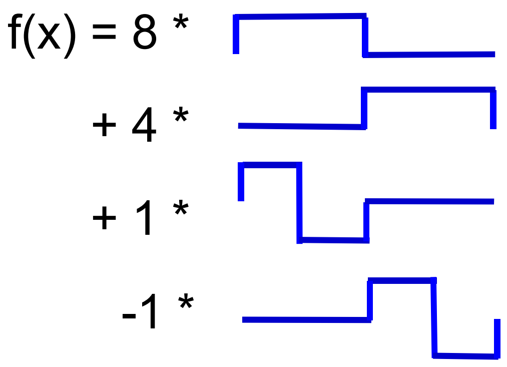
- In general, any function with resolution 2^j can be written as: $$f(x) = \sum_{i=0}^{2^j-1} c_i^{\,j}\phi_i^{\,j}(x)$$

The above is equivalent to piece-wise linear approximation.
It does not capture the notion of high or low frequency components

Averaging and Detailing Functions

- As we know, by using averaging and detail coefficient technique, we can explain the same sequence as [8,4,1,-1].
- To recover the original sequence, we can add and subtract each detail component with the average. Thus, the set of Haar scaling functions form a basis set for any one-dimensional function.
- For a given resolution j, the Haar scaling function cannot represent the details in resolution *j*+1. For this we define a set of Haar wavelet functions W^j. Informally, W^j represent the parts of a function at resolution *j*+1 that cannot be represented in resolution *j*.





A More Formal Definition of Wavelets

- W^j is the space of all functions in V^{j+1} that are orthogonal to all functions in V^j.
- A set of linearly independent basis functions $\psi_i^j(x)$ spanning W^j are called **wavelets.**

The wavelets have the two properties:

1. The basis functions $\psi_i^j(x)$ of W^j, together with the functions $\phi_i^j(x)$ of V^j, form a basis for V^{j+1}.
2. Every basis function $\psi_i^j(x)$ of W^j is orthogonal to every basis function $\phi_i^j(x)$ of V^j.

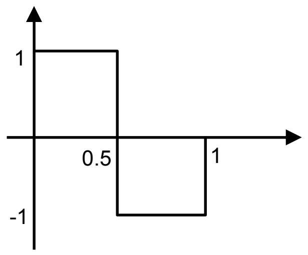
Haar Wavelet

- The Haar wavelet can be defined in general as:

$$\psi_i^j(x) = \psi(2^j x - i), \qquad i = 0, 1, \cdots, 2^j - 1$$

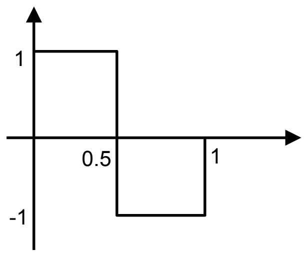
where

$$\psi(x) = \begin{cases} 1 & 0 \le x < 0.5 \\ -1 & 0.5 \le x < 1 \\ 0 & else \end{cases}$$

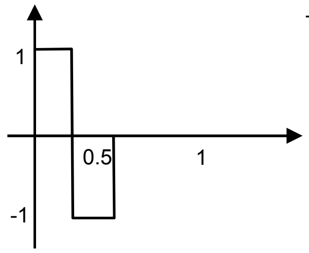


1D Haar Wavelet

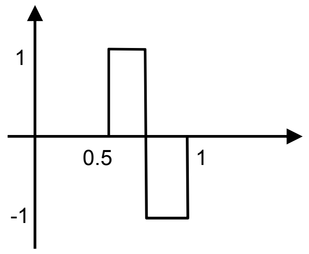
$$\psi_{0,0}(t) = \begin{cases} 1 & 0 \le t < \frac{1}{2} \\ -1 & \frac{1}{2} \le t < 1 \end{cases}$$



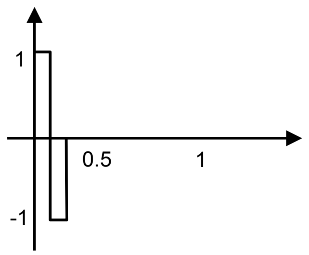
$$\psi_{1,0}(t) = \psi(2t)$$



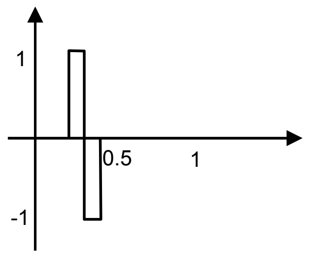
$$\psi_{1,1}(t) = \psi(2t-1)$$



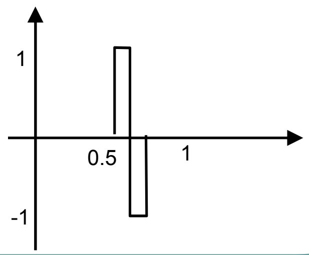
$$\psi_{2,0}(t) = \psi(4t)$$



$$\psi_{2,1}(t) = \psi(4t-1)$$



$$\psi_{2,2}(t) = \psi(4t-2)$$



Averaging and Detailing Functions

- We can now re-write our representation for [8,4,1,-1] as (in terms of basis functions in V^1 and $W^{1)}$:

$$f(x) = c_0^1 \phi_0^1(x) + c_1^1 \phi_1^1(x) + d_0^1 \psi_0^1(x) + d_1^1 \psi_1^1(x)$$

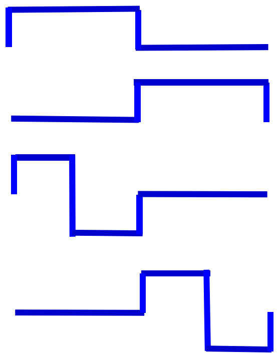
$$\text{where} \quad c_0^1 = 8, \quad c_1^1 = 4, \quad d_0^1 = 1, \quad d_1^1 = -1$$

f(x) = 8 *

+ 4 *

+ 1 *

-1*



Averaging and Detailing Functions

- If we proceed one more stage of averaging and detail coefficients, we can write the sequence as [6, 2, 1,-1]:

$$f(x) = c_0^{\,0}\phi_0^{\,0}(x) + d_0^{\,0}\psi_0^{\,0}(x) + d_0^{\,1}\psi_0^{\,1}(x) + d_1^{\,1}\psi_1^{\,1}(x)$$

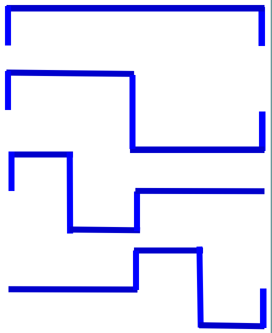
$$\text{where} \quad c_0^{\,0} = 6, \quad d_0^{\,0} = 2, \quad d_0^{\,1} = 1, \quad d_1^{\,1} = -1$$

f(x) = 6*

+ 2 *

+ 1 *

+ -1*



- Instead of writing four scaling (box) functions in V², we can write one scaling function and three wavelet functions. .

- The vector $[\phi_0^{\,0}(x), \psi_0^{\,0}(x), \psi_0^{\,1}(x), \psi_1^{\,1}(x)]$ forms the Haar basis for V².

Orthogonality and Normalization

- Note the Haar basis functions are orthogonal to each other, that is , inner product of any pair is always zero.
- Form energy preservation point of view, another desirable property of any basis function is that they be normalized.
- A basis function is normalized if the inner product of itself is 1.

Normalized Haar basis function

- We can normalize the Haar basis by replacing our earlier definition with

$$\phi_i^{\,j}(x) = \sqrt{2^j}\,\phi(2^j x - i)$$
$$\psi_i^{\,j}(x) = \sqrt{2^j}\,\psi(2^j x - i)$$

- With these modified definition, the new normalized coefficients are obtained by dividing each old coefficient with superscript j by $\sqrt{2^j}$.
- For example, [6, 2,1, -1] becomes $\begin{bmatrix} 6 & 2 & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{bmatrix}$

How to normalize the coefficients ?

- Multiplying each old coefficient with subscript *j* by $2^{-j/2}$.
  - first computing the un-normalized coefficients.
  - then normalizing them.
- Or, include normalization in the Procedure Decomposition.

Normalized 1D Haar Wavelet Transform

Decomposition

```
Procedure Decomposition(c:array [1…$2^j$])
    c = c/sqrt($2^j$)
    g = $2^j$
    while g >= 2 do
        DecompostionStep(c[1..g])
        g = g /2
    end while
End procedure

Procedure DecompositionStep(c:array[1..$2^j$])
    for i = 1 to $2^{j-1}$ do
        c'[i] = (c[2i-1]+c[2i])/sqrt(2)
        c'[$2^{j-1}$+i] = (c[2i-1]-c[2i])/sqrt(2)
    end for
    c = c'
End procedure
```

Reconstruction of original data

```
Procedure Reconstruction(c:array [1…$2^j$])
    g = 2
    while g <= $2^j$ do
        ReconstructionStep(c[1..g])
        g = 2g
    end while
    c = c*sqrt($2^j$) (undo normalization)
End procedure

Procedure ReconstructionStep (c:array[1..$2^j$])
    for i = 1 to $2^{j-1}$/2do
        c'[2i-1] = (c[i]+c[$2^{j-1}$+i])/sqrt(2)
        c'[2i] = (c[i]-c[$2^{j-1+i}$])/sqrt(2)
    end for
    c = c'
End procedure
```

Normalized 1D Haar Wavelet Transform --Example

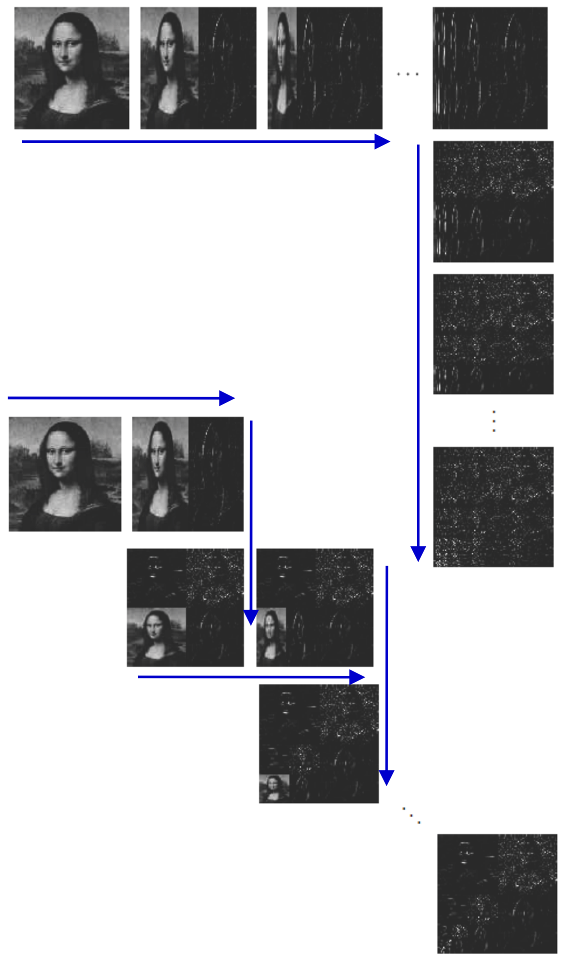
```
Input c=[9,7,3,5], j = 2
c = c / sqrt($2^2$);     c = [4.5, 3.5, 1.5, 2.5]
g = 4
DecompositionStep(c[1..4])
    c'(1) = (c[1]+c[2])/sqrt(2) = 8/sqrt(2)
    c'(3) = (c[1]-c[2])/sqrt(2) = 1/sqrt(2)
    c'(2) = (c[3]+c[4])/sqrt(2) = 4/sqrt(2)
    c'(4) = (c[3]-c[4])/sqrt(2) = -1/sqrt(2)
    c = c';      c=[8/sqrt(2), 4/sqrt(2), 1/sqrt(2), -1/sqrt(2)]
g = 2
DecompositionStep(c[1..2])
    c'(1) = (c[1]+c[2])/sqrt(2) = 6
    c'(3) = (c[1]-c[2])/sqrt(2) = 2
    c = c';      c=[6, 2, 1/sqrt(2), -1/sqrt(2)]
```

Try it yourself:

Work out the reconstruction procedure.

From 1D Haar to 2D Haar

- Standard Decomposition
  - First horizontal transform to each row of pixel values.
  - Then apply vertical transform on the columns.
- Nonstandard Decomposition
  - Apply horizontal transform and vertical transform alternatively.



Nonstandard 2D Haar Wavelet Transform

Decomposition

```
Procedure Decomposition(c:array [1…$2^j$,1…$2^j$])
    c = c/$2^j$
    g = $2^j$
    while g >= 2 do
        for row = 1 to g do
            DecompostionStep(c[row, 1..g])
/*  pair wise averaging and differencing is done on
    the row elements */
        end for
        for col = 1 to g do
            DecompostionStep(c[1..g, col])
/*  pair wise averaging and differencing is done on
    the column elements */
        end for
        g = g /2 /* repeat only for the quadrant
    containing the averages in both directions. */
    end while
End procedure
```

Reconstruction

```
Procedure Reconstruction(c:array [1…$2^j$, 1…$2^j$])
    g = 2
    while g<= $2^j$ do
      for col = 1 to g do
            ReconstructionStep(c[1..g, col])
/* the inverese pair wise operation on columns. */
      end for
      for row = 1 to g do
            ReconstructionStep (c[row, 1..g])
/* the inverese pair wise operation on columns. */
      end for
      g = 2g /* enlarging the computation to 4
    quadrants */
    end while
    c = c*$2^j$   /* Note the size of the array c is
    variable to accommodate for finer resolution */
End procedure
```

Two Dimensional Haar Basis

- The nonstandard construction of a two-dimensional basis proceeds by first defining a two-dimensional scaling function, $\phi\phi(x,y) := \phi(x)\phi(y)$
  and three wavelet functions,

$$\phi\psi(x,y) := \phi(x)\psi(y)$$

$$\psi\phi(x,y) := \psi(x)\phi(y)$$

$$\psi\psi(x,y) := \psi(x)\psi(y)$$

Two Dimensional Haar Basis

- We now denote levels of scaling with a superscript *j* and horizontal and vertical translations with a pair subscripts *k* and *l.*
- The nonstandard basis consists of a single coarse scaling function

$$\phi\phi^{0}{}_{0,0}(x,y) := \phi\phi(x,y)$$

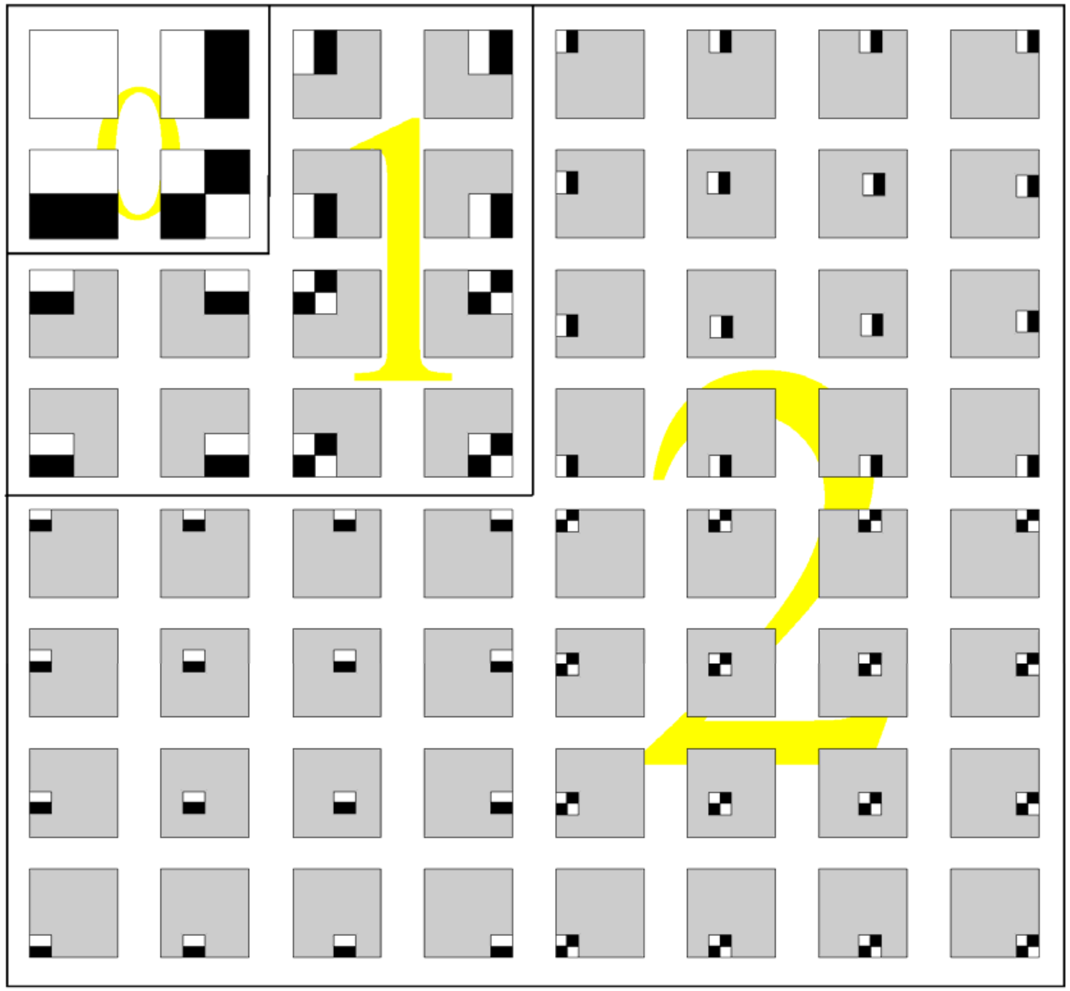
along with dilations and translations of three wavelet functions:

$$\phi\psi^{j}{}_{kl}(x,y) := 2^{j}\phi(2^{j}x-k)\psi(2^{j}y-l)$$

$$\psi\phi^{j}{}_{kl}(x,y) := 2^{j}\psi(2^{j}x-k)\phi(2^{j}y-l)$$

$$\psi\psi^{j}{}_{kl}(x,y) := 2^{j}\psi(2^{j}x-k)\psi(2^{j}y-l)$$

Example of 2D (8x8) Haar Basis



Linear Time Complexity of 2D Wavelet Transform

- Let *n* = number of pixels and let *b* be the number of coefficients in the filters.
- One level of transform takes time
  - $O(bn)$
- levels of transform takes time proportional to
  - $bn + bn/4 + \ldots + bn/4^{k-1} < (4/3)bn$.
- The wavelet transform is linear time when the filters have constant size.

$$1 + a + a^2 + \ldots + a^k = (1-a^{k+1})/(1-a)$$

When Multiresolution Coding was a New Idea . . .

“This manuscript is okay if compared to some of the weaker papers. [. . .] however, I doubt that anyone will ever use this algorithm again.”

---- Anonymous reviewer of Burt and Adelson‘s original paper, 1982

Multi-resolution Analysis

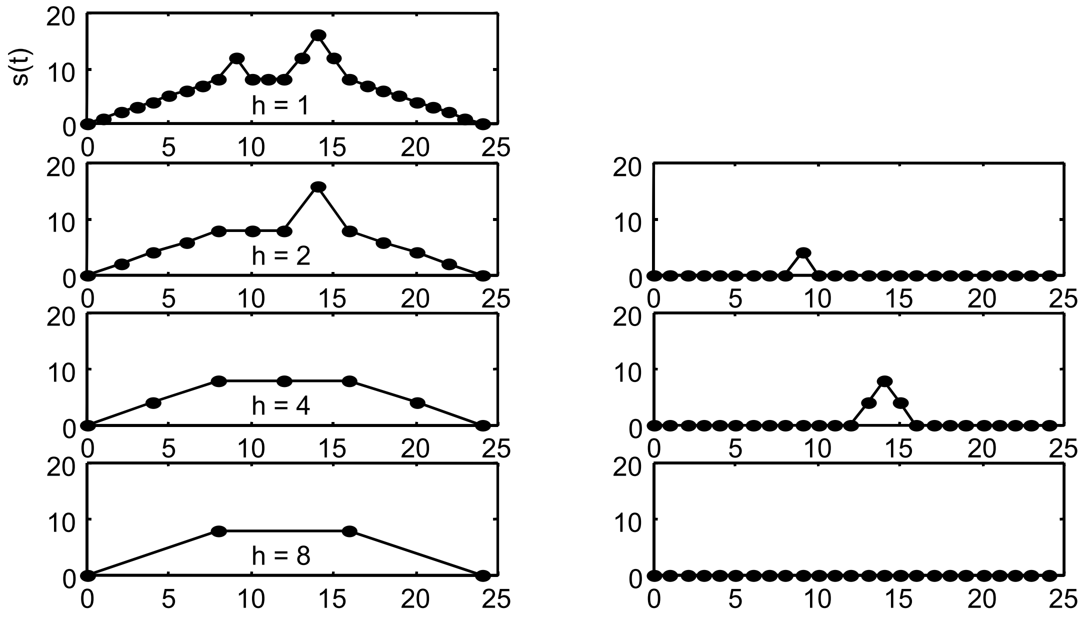
- Let’s say , we want to represent a real number N=87/7=12. 4285714… by a series of successive approximation. Depending on the desired accuracy, we can approximate N successively as sequence of “round-off” values 10, 12, 12.4,12.42, …etc. The successive difference between two consecutive round-off values (2, 0.4, 0.02, 0.008, …) is called the “detail” part. The round off values are sometimes also called the “averages”.
- In multi-resolution analysis, we use a *scaling function* to represent the round off values and a *wavelet function* to represent the detail values.

Multi-resolution Analysis

- The further we descend the level of details, the more accurate is the approximation. In the other direction, if we “stretch” the scaling function more and more, we end up seeing nothing, as if we trying to approximate 87/7 by 100’s and at that point all information is in the details: 10+2+0.4+0.02+0.008.
- This numerical example gives only a conceptual idea of multi-resolution analysis.

Multiresolution Analysis (MRA)

- Objective: To analyze a complicated function by dividing it into several simpler ones and studying them separately.
- The further we descend the level of details, the less accurate is the approximation.



Multiresolution Analysis using Haar Wavelets

- Suppose we have a function *f*(*x*) that can be accurately represented at a resolution 2^j. We have seen that *f*(*x*) can be expanded as

$$f^j(x) = \sum_{i=0}^{2^j-1} c_i^{\,j} \phi_i^{\,j}(x)$$

where $\phi_i^{\,j}$ is the set of 'box' basis function for V^j and $f(x) \in V^j$

Multiresolution Analysis using Haar Wavelets

- We can represent the function at a lower resolution level (j-1) with some detail coefficients:

$$f^j(x) = \sum_{i=0}^{2^{j-1}-1} c_i^{j-1}\phi_i^{j-1}(x) + \sum_{i=0}^{2^{j-1}-1} d_i^{j-1}\psi_i^{j-1}(x)$$

$$\text{where } c_i^{j-1} = \frac{c_{2i}^{\ j} + c_{2i+1}^{\ j}}{2} \text{ and } d_i^{j-1} = \frac{c_{2i}^{\ j} - c_{2i+1}^{\ j}}{2}$$

where $\phi_i^{j-1}(x)$ is the ‘box’ scaling function at level (j-1) spanning the vector space V^{j-1}, $\psi_i^{j-1}(x)$ are the wavelet functions for the vector space W^{j-1}.

- Thus, we have $V^j = V^{j-1} \oplus W^{j-1}$

Multiresolution Analysis using Haar Wavelets

- We can now continue the decomposition process as:

$$f^{j-1}(x) = \sum_{i=0}^{2^{j-1}-1} c_i^{j-1}\phi_i^{j-1}(x)$$

$$= \sum_{i=0}^{2^{j-2}-1} c_i^{j-2}\phi_i^{j-2}(x) + \sum_{i=0}^{2^{j-2}-1} d_i^{j-2}\psi_i^{j-2}(x)$$

$$\text{where} \quad c_i^{j-2} = \frac{c_{2i}^{j-1} + c_{2i+1}^{j-1}}{2} \quad \text{and} \quad d_i^{j-2} = \frac{c_{2i}^{j-1} - c_{2i+1}^{j-1}}{2}$$

- So, we have:

$$V^j = V^{j-2} \oplus W^{j-1} \oplus W^{j-1}$$

Multiresolution Analysis using Haar Wavelets

- If we carry on the process all the way to V^0, we have:

$$V^j = V^0 \oplus W^0 \oplus W^1 \oplus \cdots \oplus W^{j-1}$$

  Where $(V^0, W^0, W^1, \cdots, W^{j-1})$ form a Haar basis for V^j.

- For example, the Haar basis for V^2 is:

$$\left[\phi_0^{\ 0} \quad \psi_0^{\ 0} \quad \psi_0^{\ 1} \quad \psi_1^{\ 1}\right] \equiv V^0 \oplus W^0 \oplus W^1$$

Multiresolution Analysis using Haar Wavelets

- Alternatively, we can write an expansion as :

$$f^j(x) = c_0{}^0 \phi_0{}^0(x) + \sum_{k=0}^{j-1} \sum_{i=0}^{2^{k-1}-1} d_i{}^k \psi_i{}^k(x)$$

- This equation is the Haar wavelet decomposition of an arbitrary function
- *f*(*x*).

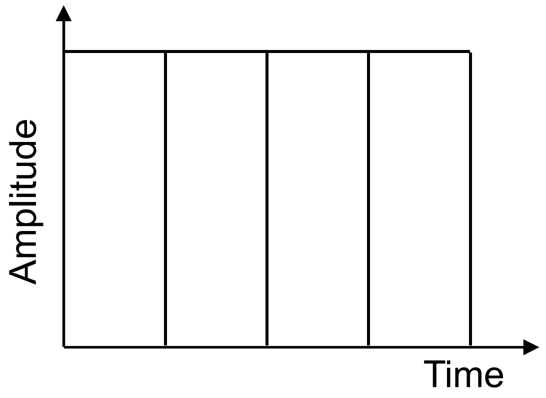
Multiresolution Analysis using Haar Wavelets

- For *s<=j-1*, if the reconstruction of the image using only up to s^{th} level of detail is satisfactory, we can stop the decomposition at the s^{th} level:

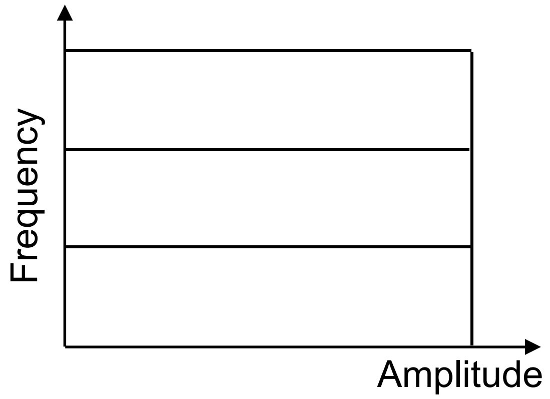
$$V^{j} \approx V^{s} \oplus W^{s} \oplus W^{s+1} \oplus \cdots \oplus W^{j-1}$$

- That is, if a function *f*(*x*) can be exactly represented at 2^j resolution, we can decompose it into a sum of functions, starting with a low-resolution approximation followed by a sequence of functions generated by dilations of the wavelet that represent the detailed coefficients.

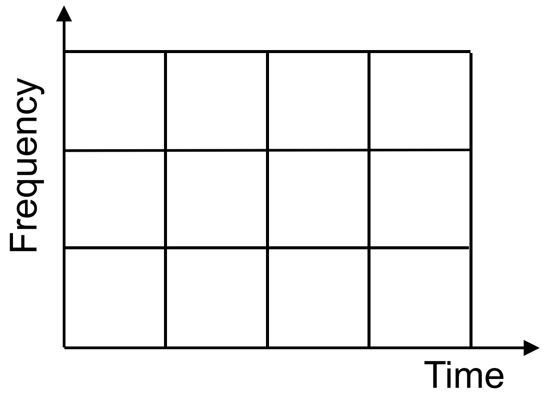
Overview: FFT - STFT - Wavelet



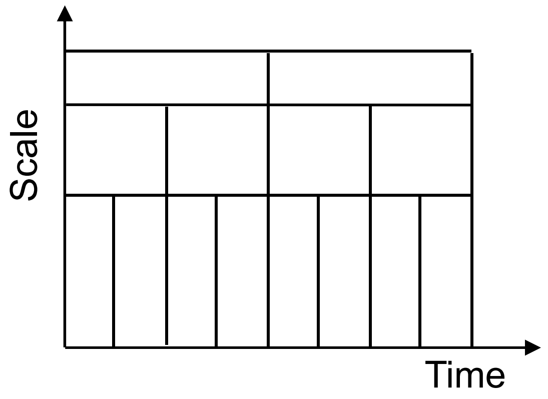
Time Domain



Frequency Domain



STFT (Gabor)



Wavelet Analysis

The Wavelet Transform

- Linear transform.
- All analysis functions $\psi_{s,\tau} = \psi(\frac{t-\tau}{s})$ are shifts and dilations of the mother wavelet $\psi(t)$
- Time resolution and frequency resolution vary as a function of scale.
- Continuous wavelet transform (CWT)
  - s and тare continuous.
- Discrete wavelet transform (DWT)
  - s and тare discrete.

General Wavelet Transformation

Analysis $\psi(t)$ is a “mother wavelet function

$$\psi_{s,\tau}(t) = \frac{1}{\sqrt{s}}\psi(\frac{t-\tau}{s})$$ Is a scaled and translated mother wavelet

Where s, τ are scaling and translation parameters.

Let $f(t)$ be an arbitrary function of time (or space)

The wavelet transform of the function is given by the set of coefficients:

$$WT(s,t) = \langle \psi_{s,t}, f(t) \rangle = \int_{-\infty}^{\infty} \frac{1}{\sqrt{s}} \psi_{s,\tau}(t) f(t) dt$$

$$= \frac{1}{\sqrt{s}} \int f(t) \psi(\frac{x-\tau}{s}) dt$$

Synthesis

The inverse wavelet transform is given by

$$f(t) = \frac{1}{C_\psi \sqrt{s}} \int\int WT(s,\tau)\psi_{s,\tau} ds d\tau$$

where $$C_\psi = \int_{-\infty}^{\infty} \frac{|\Psi(w)|^2}{|w|} dw$$

and $\Psi(w)$ is the Fourier transform of $\psi_{s,\tau}(t)$

Two conditions must be satisfied by the wavelet coefficients:

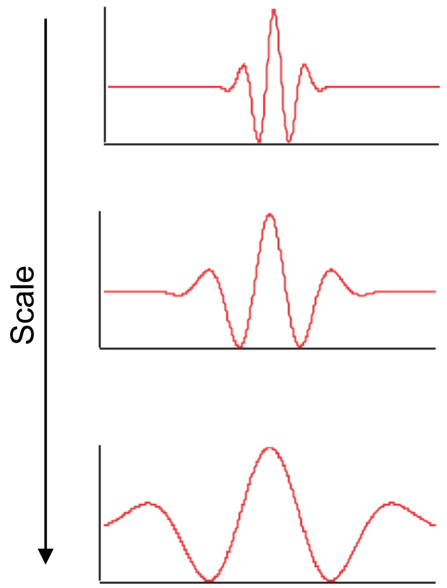
1)Wave condition: $\int_{-\infty}^{\infty} \psi_{s,\tau}(t) dt = 0$

2) Admissibility condition: $\int_{-\infty}^{\infty} |\psi_{s,\tau}(t)|^2 dt < \infty$

For an arbitrary pair of real values (s,τ) there is a wavelet transform. So, the entire real plane is its support. (Read example p.471 Salomon)

Scaling

- Scaling = frequency band
- Small scale
  - Rapidly changing details,
  - Like high frequency
- Large scale
  - Slowly changing details
  - Like low frequency



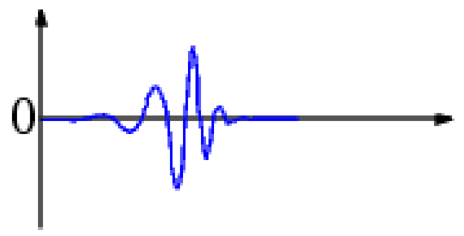
Wavelet Basis functions at 3 different scales

More on Scale

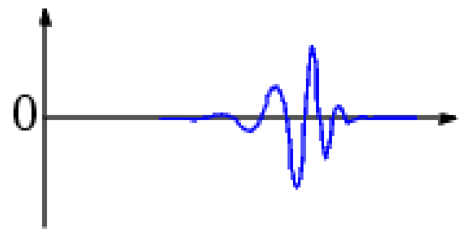
- It lets you either narrow down the frequency band of interest, or determine the frequency content in a narrower time interval
- Good for <u>non-stationary</u> data
- Low scale→ *a* Compressed wavelet→ Rapidly changing details→ High frequency.
- High scale →*a* Stretched wavelet → Slowly changing, coarse features → Low frequency.

Shifting

- Shifting a wavelet simply means delaying (or hastening) its onset.
- Mathematically, shifting a function f(t) by *k* is represented by f(t-k).

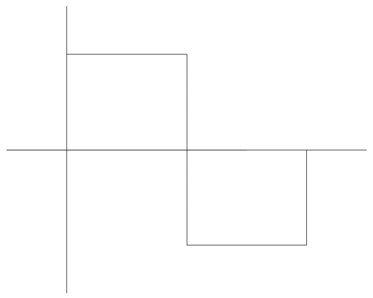


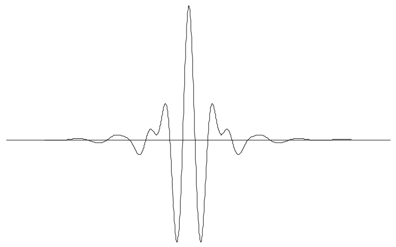
Wavelet function
$\psi(t)$

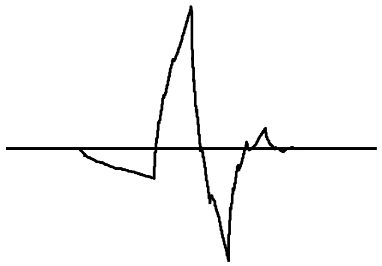


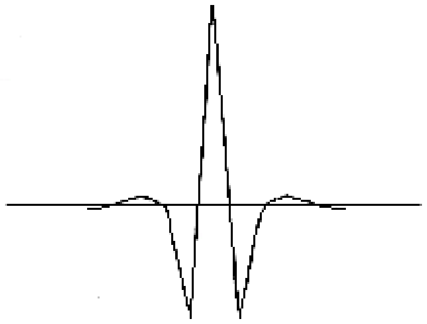
Shifted wavelet function
$\psi(t-k)$

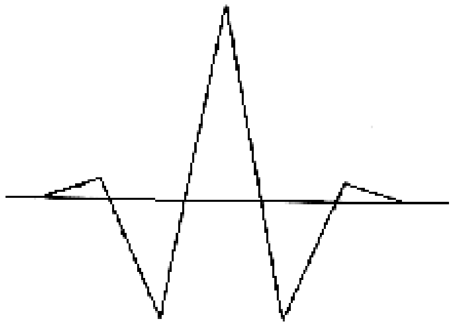
Different Types of Mother Wavelets


Haar


Meyer


Daubechies


Battle-Lemarie


Chui-Wang

Calculate the CWT Coefficients

- The result of the CWT are many *wavelet coefficients* *WT*

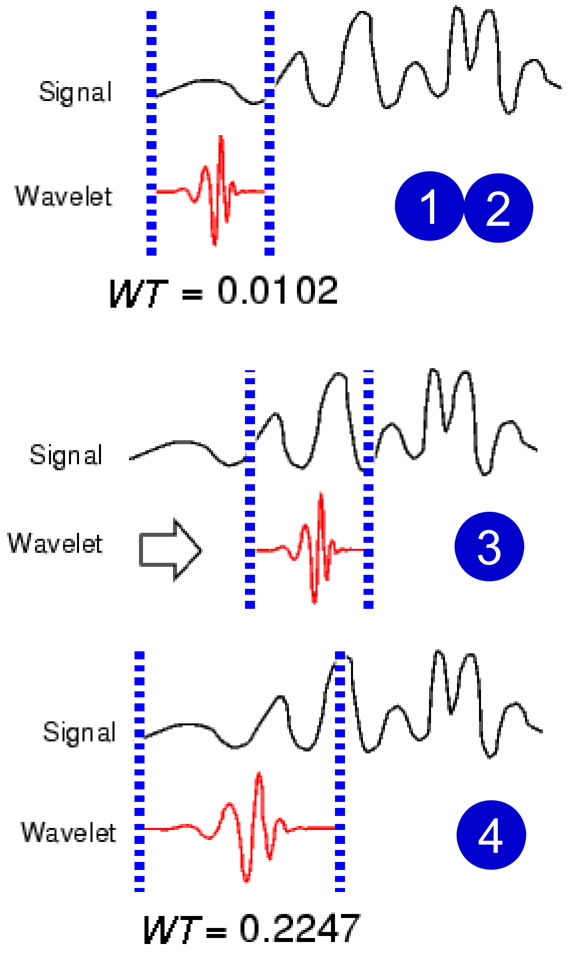
$$WT(s,\tau) = \langle \psi_{\tau,s}, f(t) \rangle = \frac{1}{\sqrt{s}} \int f(t)\psi(\frac{t-\tau}{s})dt$$

  - Function of scale and position.
- How to calculate the coefficient?

```
for each SCALE s
    for each POSITION t
        WT (s, t) =   Signal  x Wavelet (s, t)
    end
end
```

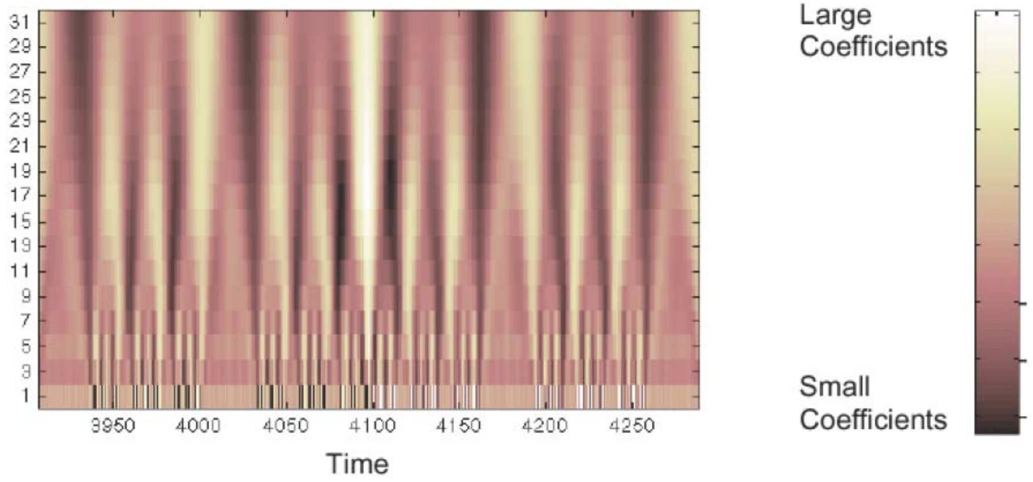
Calculate the CWT Coefficients

1. Take a wavelet and compare it to a section at the start of the original signal.
2. Calculate a correlation coefficient *WT*
3. Shift the wavelet to the right and repeat steps 1 and 2 until you've covered the whole signal.
4. Scale (stretch) the wavelet and repeat steps 1 through 3.
5. Repeat steps 1 through 4 for all scales.

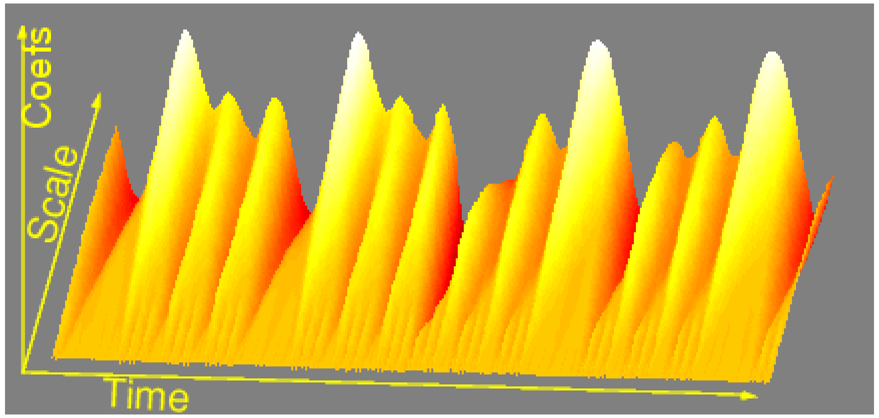


Visualizing the CWT Coefficients

- 2D



- 3D



Discrete Wavelet Transform

- Calculating wavelet coefficients at every possible scale is a fair amount of work, and it generates an awful lot of data.
- What if we choose only a subset of scales and positions at which to make our calculations?
- It turns out, rather remarkably, that if we choose scales and positions based on powers of two --- so-called *dyadic* scales and positions --- then our analysis will be much more efficient and just as accurate.

Discrete Wavelet Transform

- If *(s, τ)* take discrete value in R^2, we get DWT.
- A popular approach to select *(s, τ)* is

$$s = \frac{1}{s_0^m} \qquad s_0 = 2 \quad \rightarrow \quad s = \frac{1}{2^m} = < 1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \ldots >, \qquad \text{m: integer}$$

$$\tau = \frac{n\tau_0}{s_0^m} \qquad s_0 = 2\,, \quad \tau_0 = 1, \qquad \tau = \frac{n}{2^m} \qquad \text{n, m: integer}$$

- So,

$$\psi_{s,\tau}(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t-\tau}{s}\right) = 2^{m/2}\psi\left(\frac{t-\frac{n}{2^m}}{\frac{1}{2^m}}\right) = \psi_{m,n}(t) = 2^{m/2}\psi\left(2^m t - n\right)$$

Discrete Wavelet Transform

- Wavelet Transform:

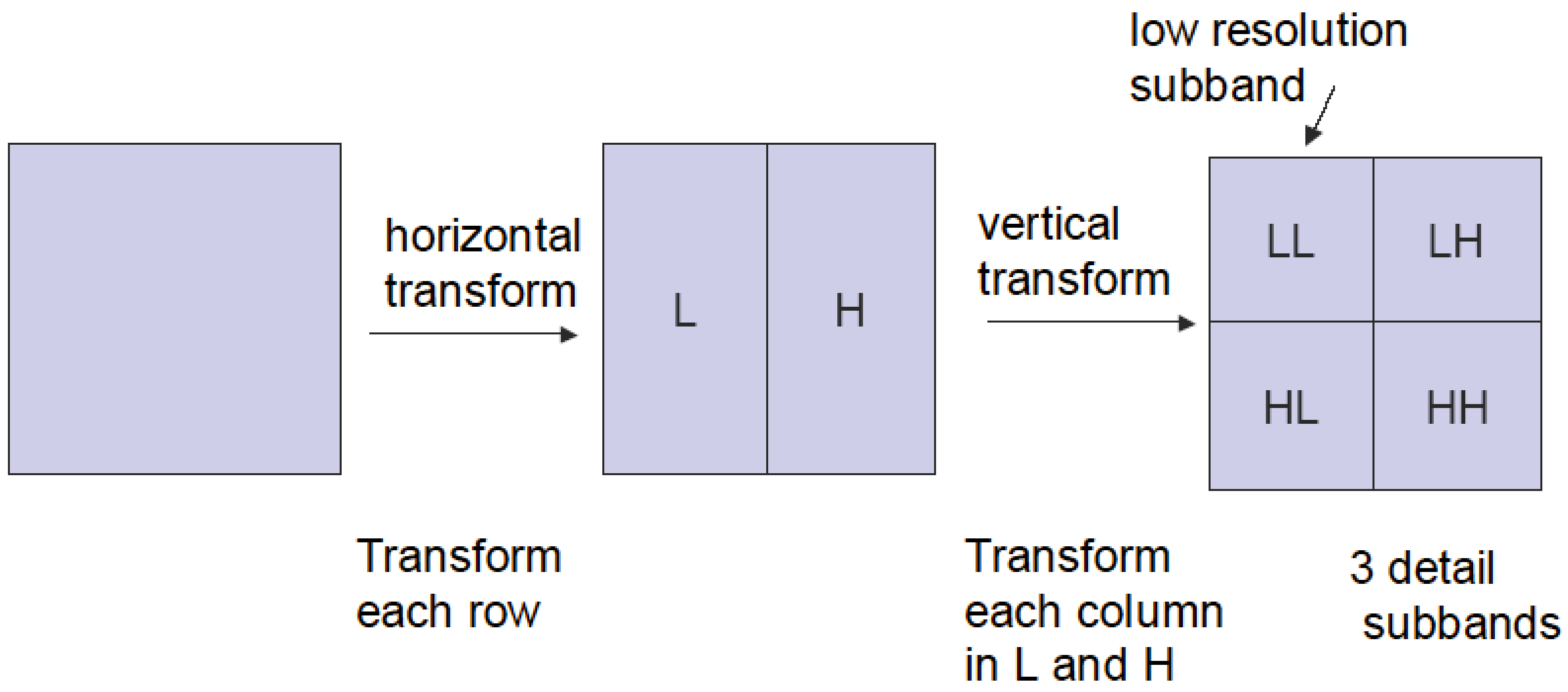
$$DWT_{m,n} = \langle f(t), \psi_{m,n}(t) \rangle = 2^{m/2} \int f(t)\psi(2^m t - n)dt$$

- Inverse Wavelet Transform

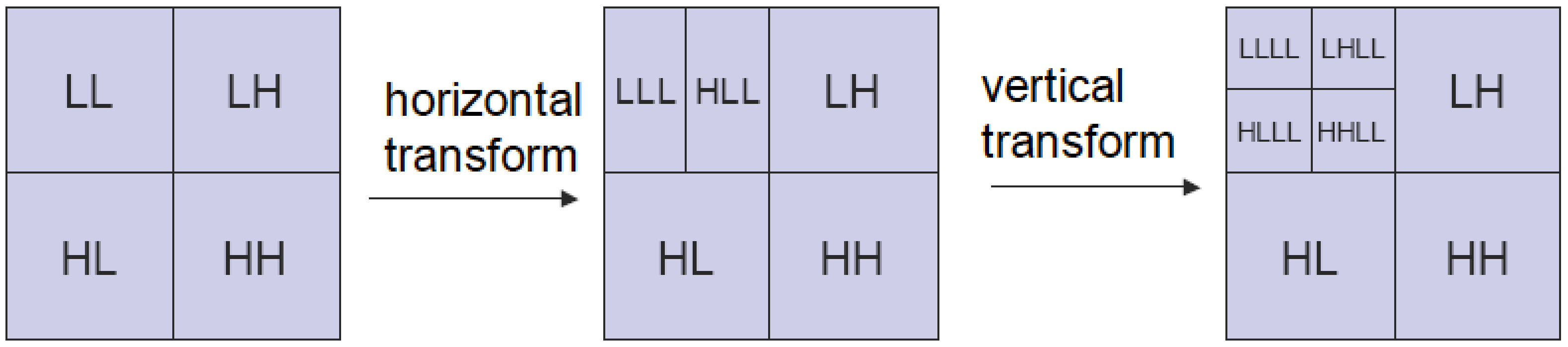
$$f(t) = \sum_m \sum_n DWT_{m,n}\psi_{m,n}(t) + c\Phi(t)$$

- If *f(t)* is continuous while *(s, τ)* consists of discrete values, the series is called the Discrete Time Wavelet Transform (DTWT).
- If *f(t)* is sampled (that is, discrete in time, as well as *(s, τ)* are discrete, we get Discrete Wavelet Transform (DWT).

Two Dimensional Transform



Two Dimensional Transform (Continued)

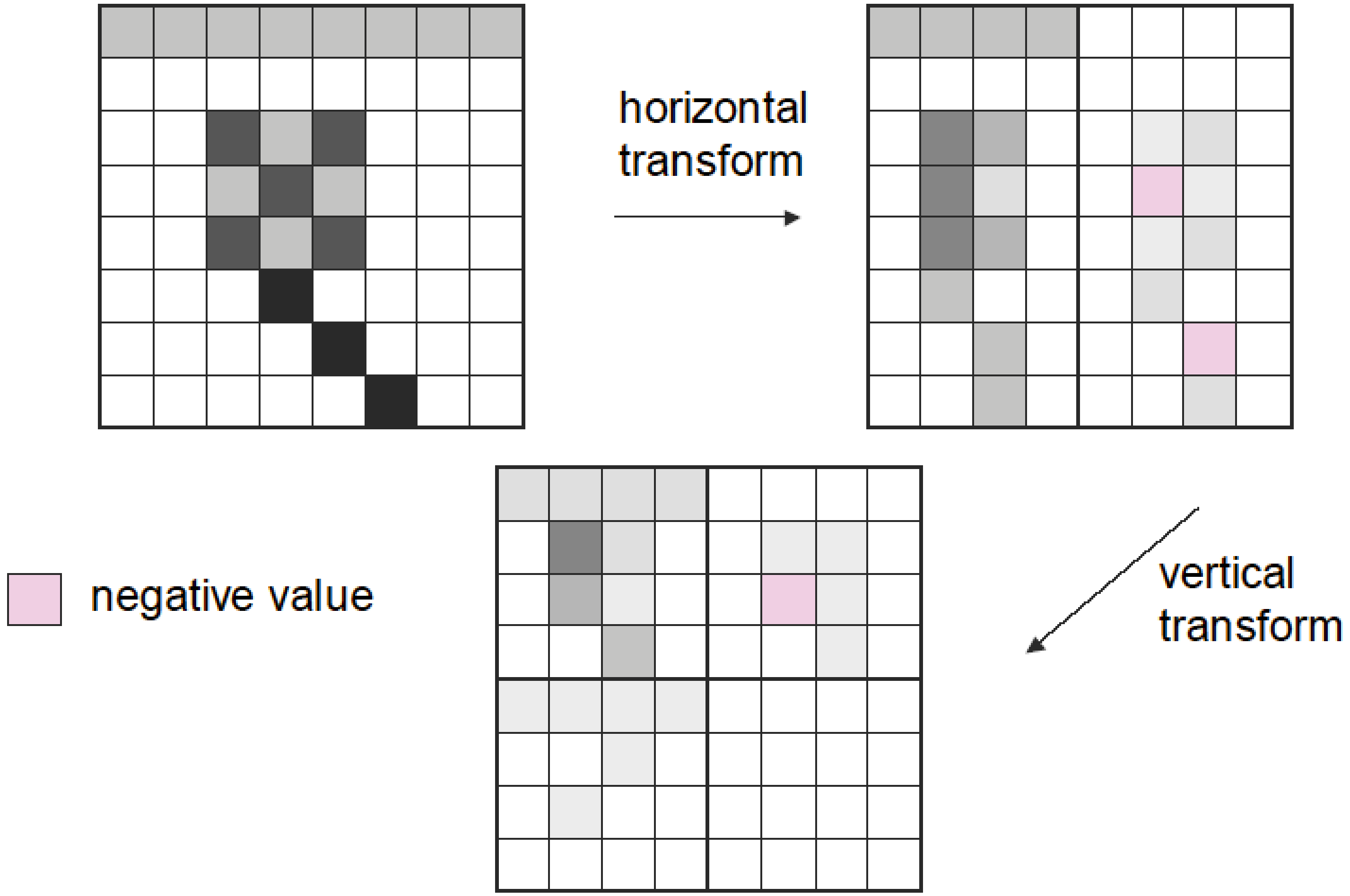


Transform each row in LL

Transform each column in LLL and HLL

2 levels of transform gives 7 subbands.
k levels of transform gives 3k + 1 subbands.

Two Dimensional Average Transform



Wavelet Transform Details

- Conversion
  - Convert gray scale to floating point.
  - Convert color to Y U V and then convert each band to floating point. Compress separately.
- After several levels (3-8) of transform we have a matrix of floating point numbers called the wavelet transformed image (coefficients).

Coding Wavelet Coefficients

- No compression in the Wavelet Transform.
  - Wavelet coefficients is just another lossless representation of the original signal.
- However, most of the energy is concentrated in the low-frequency part of the wavelet coefficients.
- So, we can compress the coefficients.
  - Entropy coder (Huffman, Arithmetic)
  - Vector Quantization
  - Organization of the coefficients is key to efficient coding
    - The low pass residual subband has the same statistics as an image.
    - The other subbands are zero mean.

EZW: How Do Zero Trees Work

- Invented by Shapiro, 1993 (EZW)
- Refined by Said and Pearlman, 1996(SPIHT).
- Transform Coefficients are losslessly encoded (after quantization).
- How we present the coefficients for coding will determine how efficient the encoding will be.
- Coefficients can be ordered so that progressive transmission or region of interest is presented to the coder.

Example

- Image Sample

139	144	149	153	155	155	155	155
144	151	153	156	159	156	156	156
150	155	160	163	158	156	156	156
159	161	162	160	160	159	159	159
159	160	161	162	162	155	155	155
161	161	161	161	160	157	157	157
162	162	161	163	162	157	157	157
162	162	161	161	163	158	158	158

Example

- Haar Filter Results

1259.6	0.1	-13.2	1.5	-6.0	-3.5	1.5	0.0
-17.4	-12.9	-0.5	5.0	-3.5	-0.5	1.5	0.0
-20.2	4.0	-3.2	0.0	-0.5	-0.5	5.0	0.0
-2.0	-3.0	1.5	0.0	0.0	-1.0	5.0	0.0
-6.0	-3.5	-2.5	-1.0	1.0	-0.5	-1.5	0.0
-7.5	0.5	-2.5	-3.0	-1.5	-2.5	0.5	0.0
-1.5	0.5	0.0	-2.0	-0.5	-0.5	2.0	0.0
0.0	1.0	1.0	-1.0	0.0	-1.0	0.0	0.0

Quantization (Threshold 9)

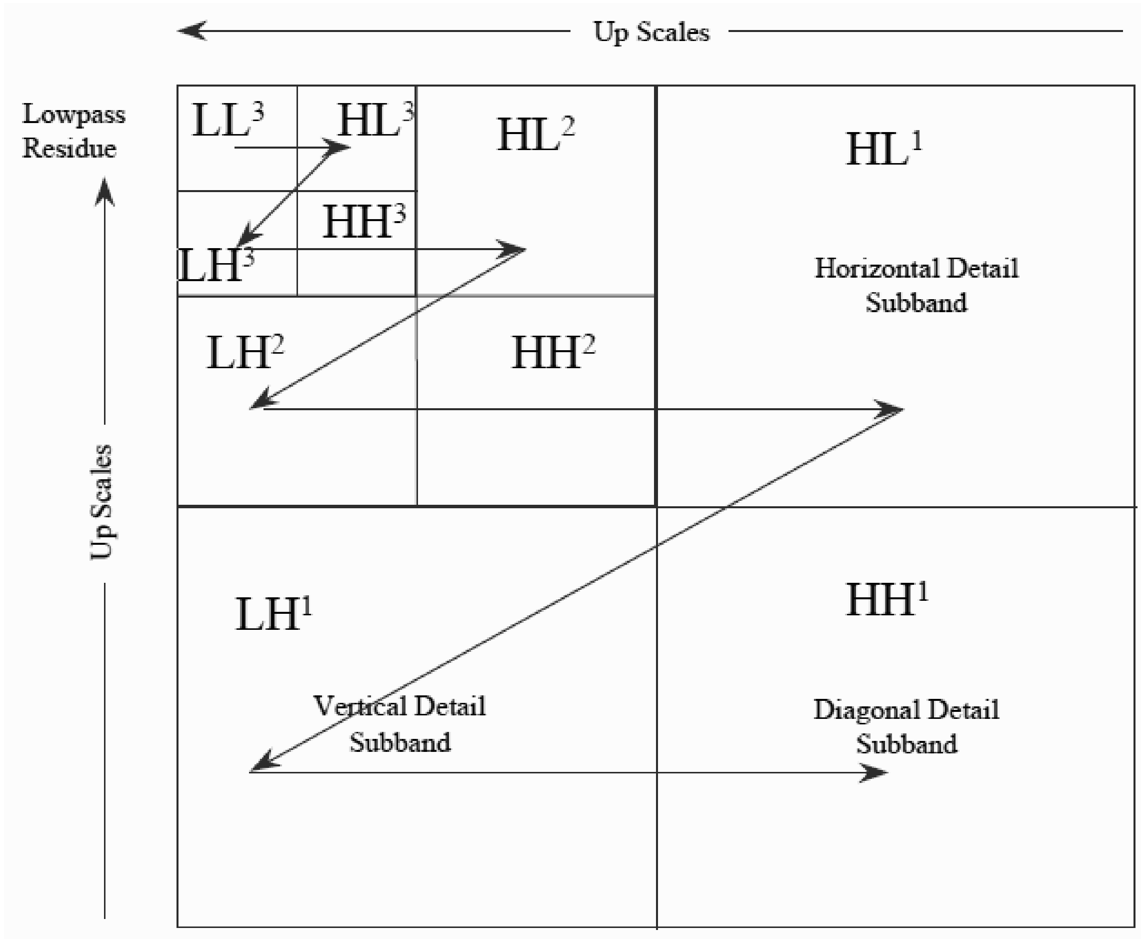
9 *

140	0	-1	0	-1	0	0	0
-2	-1	0	1	0	0	0	0
0	0	0	0	0	0	1	0
0	0	0	0	0	0	1	0
-1	0	0	0	0	0	0	0
-1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

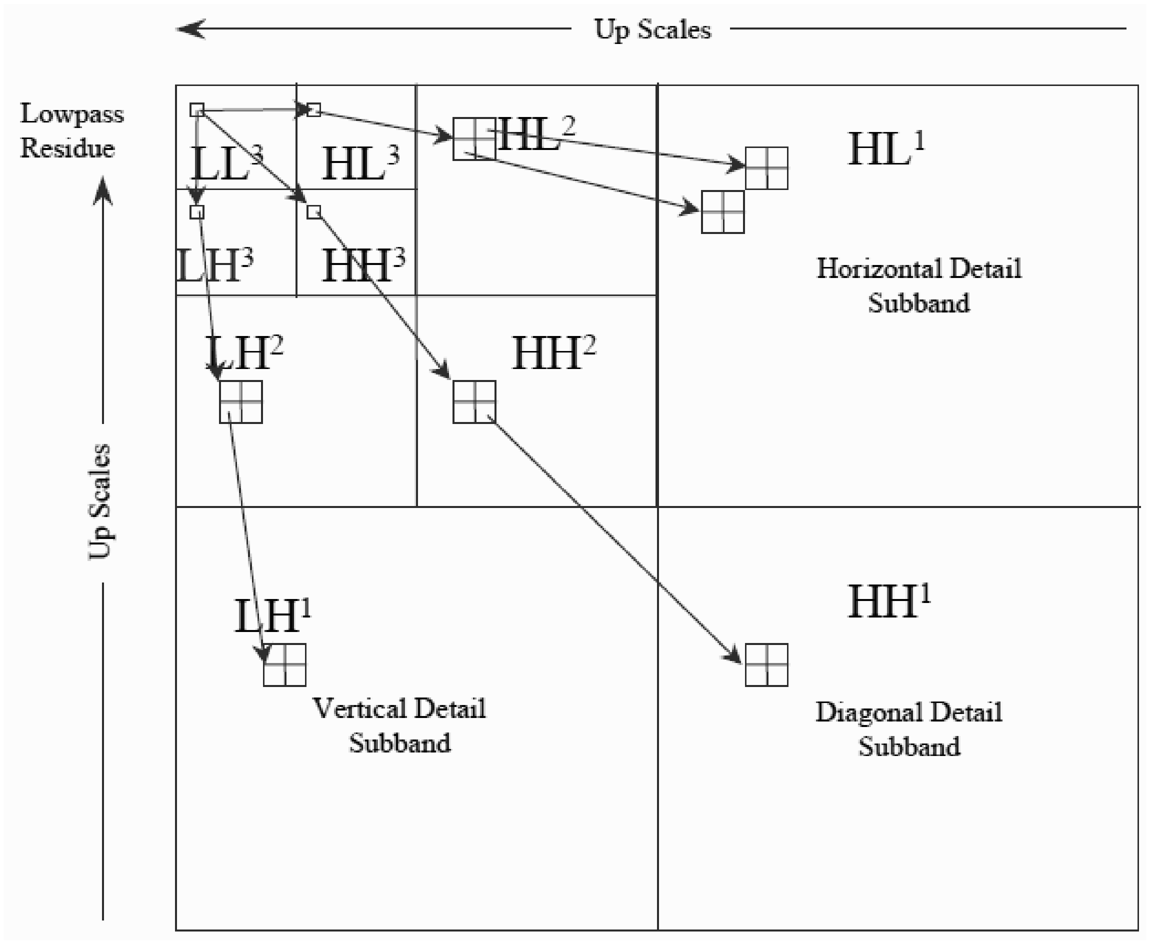
Recovered Image

138	147	152	152	156	156	156	156
147	156	152	152	156	156	156	156
152	152	161	161	156	156	156	156
161	161	161	161	156	156	156	156
161	161	161	161	165	156	156	156
161	161	161	161	165	156	156	156
161	161	161	161	165	156	156	156
161	161	161	161	165	156	156	156

Wavelet Zero Tree

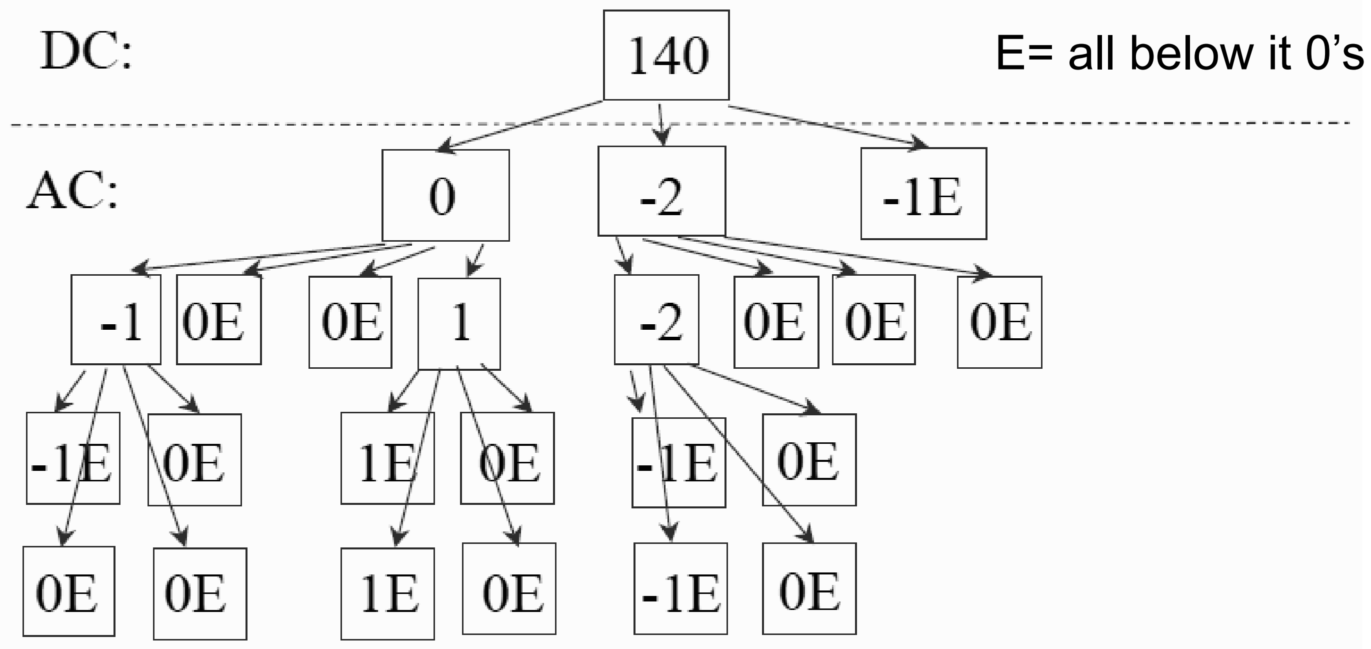


Wavelet Zero Tree



Coded Sample Images

AC Coefficient Symbol String:
0,-2,-1E; -1,0E,0E,1 -2,0E,0E,0E;
-1E,0E,0E,0E, 1E, 0E, 1E, 0E, -1E, -1E, 0E, 0E.



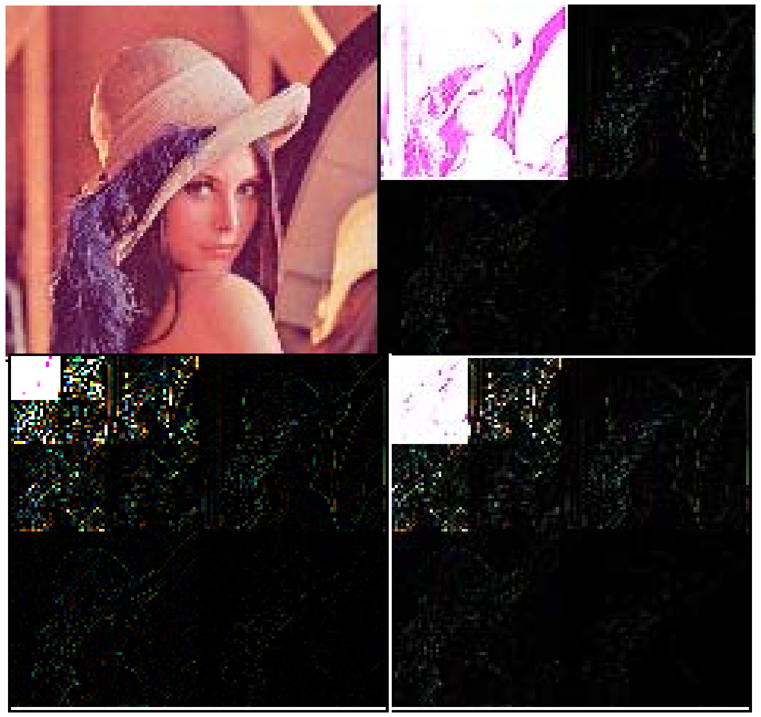
E= all below it 0's

EZW – basic concepts(1)

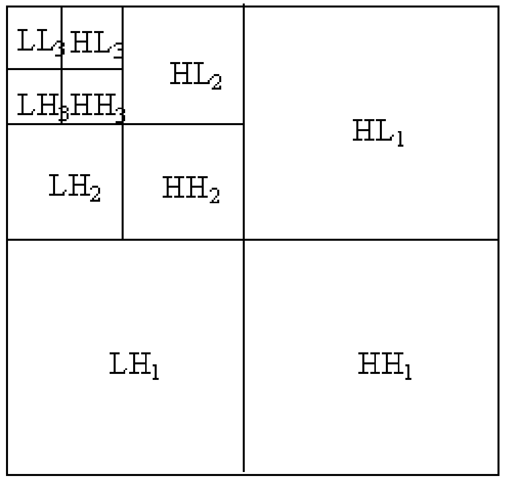
- E – The EZW encoder is based on progressive encoding. Progressive encoding is also known as <u>embedded encoding</u>
- Z – A data structure called <u>zero-tree</u> is used in EZW algorithm to encode the data
- W – The EZW encoder is specially designed to use with <u>wavelet transform</u>. It was originally designed to operate on images (2-D signals)

- http://perso.wanadoo.fr/polyvalens/clemens/ezw/ezw.html

EZW – basic concepts(2)



A Multi-resolution Analysis Example



Lower octave has higher resolution and contains higher frequency information

EZW – basic concepts(3)

The EZW algorithm is based on two observations:

- Natural images in general have a low pass spectrum. When an image is wavelet transformed, the energy in the sub-bands decreases as the scale goes lower (low scale means high resolution), so the wavelet coefficient will, on average, be smaller in the lower levels than in the higher levels.
- Large wavelet coefficients are more important than small wavelet coefficients.

631	544	86	10	-7	29	55	-54
730	655	-13	30	-12	44	41	32
19	23	37	17	-4	–13	-13	39
25	-49	32	-4	9	-23	-17	-35
32	-10	56	-22	-7	-25	40	-10
6	34	-44	4	13	-12	21	24
-12	-2	-8	-24	-42	9	-21	45
13	-3	-16	-15	31	-11	-10	-17

typical wavelet coefficients for a 8*8 block in a real image

EZW – basic concepts(4)

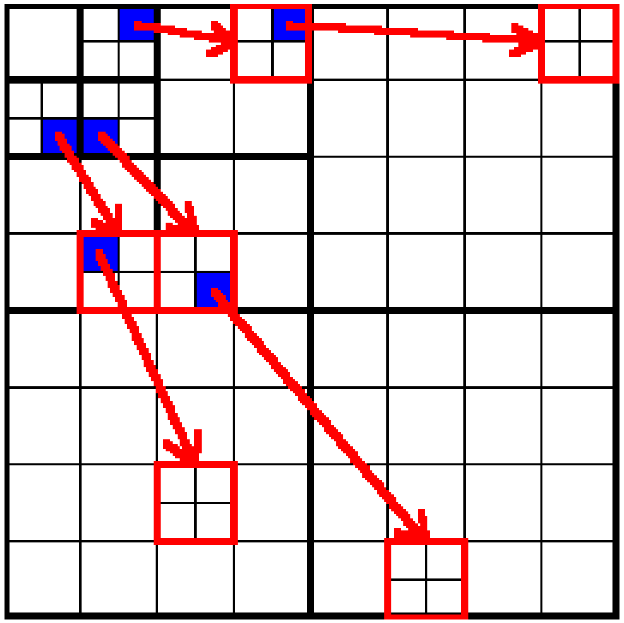
The observations give rise to the basic progressive coding idea:

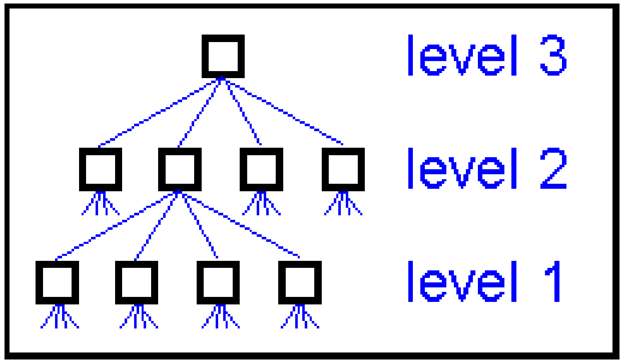
1. We can set a threshold T, if the wavelet coefficient is larger than T, then encode it as 1, otherwise we code it as 0.
2. ‘1’ will be reconstructed as T (or a number larger than T) and ‘0’ will be reconstructed as 0.
3. We then decrease T to a lower value, repeat 1 and 2. So we get finer and finer reconstructed data.

The actual implementation of EZA algorithm should consider :

1. What should we do to the sign of the coefficients. (positive or negative) ? – answer: use POS and NEG
2. Can we code the ‘0’s more efficiently? -- answer: zero-tree
3. How to decide the threshold T and how to reconstruct? – answer: see the algorithm

EZW – basic concepts(5)





coefficients that are in the same spatial location consist of a ***quad-tree***.

EZW – basic concepts(6)

- The definition of the zero-tree:

  There are coefficients in different subbands that represent the same spatial location in the image and this spatial relation can be depicted by a quad tree except for the root node at top left corner representing the DC coeficient which only has three children nodes.

- Zero-tree Hypothesis

  If a wavelet coefficient c at a coarse scale is insignificant with respect to a given threshold T, i.e. |c|<T then all wavelet coefficients of the same orientation at finer scales are also likely to be insignificant with respect to T.

EZW – the algorithm(1)

- First step: The DWT of the entire 2-D image will be computed by FWT
- Second step: Progressively EZW encodes the coefficients by decreasing the threshold
- Third step: Arithmetic coding is used to entropy code the symbols

EZW – the algorithm(2)

What is inside the second step?

```
threshold = initial_threshold;
do {
  dominant_pass(image);
  subordinate_pass(image);
  threshold = threshold / 2;
} while (threshold >
minimum_threshold);
```

The main loop ends when the threshold reaches a minimum value, which could be specified to control the encoding performance, a "0" minimum value gives the lossless reconstruction of the image

The initial threshold t0 is decided as:

$$t_0 = 2^{\lfloor \log_2(MAX(|\gamma(x,y)|)) \rfloor}$$

Here MAX() means the maximum coefficient value in the image and *y*(x,y) denotes the coefficient. With this threshold we enter the main coding loop

EZW – the algorithm(3)

In the *dominant_pass*

- All the coefficients are scanned in a special order
- If the coefficient is a zero tree root, it will be encoded as ZTR. All its descendants don't need to be encoded – they will be reconstructed as zero at this threshold level
- If the coefficient itself is insignificant but one of its descendants is significant, it is encoded as IZ (isolated zero).
- If the coefficient is significant then it is encoded as POS (positive) or NEG (negative) depends on its sign.

This encoding of the zero tree produces significant compression because gray level images resulting from natural sources typically result in DWTs with many ZTR symbols. Each ZTR indicates that no more bits are needed for encoding the descendants of the corresponding coefficient

EZW – the algorithm(5)

At the end of *dominant_pass*

- all the coefficients that are in absolute value larger than the current threshold are extracted and placed without their sign on the subordinate list and their positions in the image are filled with zeroes. This will prevent them from being coded again.

In the *subordinate_pass*

- All the values in the subordinate list are refined. this gives rise to some juggling with uncertainty intervals and it outputs next most significant bit of all the coefficients in the subordinate list.

EZW – An example(1)

63	-34	49	10	7	13	-12	7
-31	23	14	-13	3	4	6	-1
15	14	3	-12	5	-7	3	9
-9	-7	-14	8	4	-2	3	2
-5	9	-1	47	4	6	-2	2
3	0	-3	2	3	-2	0	4
2	-3	6	-4	3	6	3	6
5	11	5	6	0	3	-4	4

Wavelet coefficients for a 8*8 block

EZW – An example(2)

The initial threshold is 32 and the result from the dominant_pass is shown in the figure

63 POS	-34 NEG	49 POS	10 ZTR	7 ZTR	13 ZTR	-12	7
-31 IZ	23 ZTR	14 ZTR	-13 ZTR	3 ZTR	4 ZTR	6	-1
15 ZTR	14 IZ	3	-12	5	-7	3	9
-9 ZTR	-7 ZTR	-14	8	4	-2	3	2
--5	9	-1 ZTR	47 POS	4	6	-2	2
3	0	-3 ZTR	2 ZTR	3	-2	0	4
2	-3	6	-4	3	6	3	6
5	11	5	6	0	3	-4	4

Data without any symbol is a node in the zero-tree.

```
/* * Subordinate pass */
subordinate_threshold = current_threshold/2;
for all elements on subordinate list do {
      if (coefficient > subordinate_threshold) {
              output a one;
             coefficient = coefficient-subordinate_threshold;
     }
     else output a zero;
}
```

EZW – An example(3)

The result from the dominant_pass is output as the following:

POS, NEG, IZ, ZTR, POS, ZTR, ZTR, ZTR, ZTR, IZ, ZTR, ZTR, ZTR,ZTR,ZTR,ZTR, ZTR,POS, ZTR,ZTR

The significant coefficients are put in a subordinate list and are refined. A one-bit symbol S is output to the decoder.

ABS(Original data)	63	34	49	47
Output symbol	1	0	1	0
ABS(Reconstructed data)	56	40	56	40

- S Bit: If Data- threshold (T) is greater than or equal to 0.5T, then S bit is set 1; otherwise, if it is less than 0.5T, it is set to be 0.
- Reconstruction Value: Create a binary number starting from the most significant bits whose values can be predicted certainly using T and 0.5T only. For example 63 is greater than T=32 and 63-32=31 is greater than 16. So, its S bit is 1. Also the most significant value bit is ‘1’. Then 63-32=31 is greater than 16 so the second value bit is also 1. Now, 32+16=48. So we can say with certainty that the number is greater than 48. Then pick the midpoint between 48 and next number which is a power of 2. This gives 56 as the reconstructed value.
- sign 32 16 8 4 2 1
- 0 1 1 ? ? ? ?
- The number 34 is greater than 32 but 34-32 is less than 16; so its S bit is ‘0’ and then pick the number which is the midpoint between 32 and 48. Thus 32 is reconstructed as 40. Similarly, 49 and 47 are reconstructed as 56 and 40, respectively with S bits 1 and 0, respectively. The actual sign of the numbers are captured in POS and NEG symbols. So, S is not a sign bit.

EZW – An example(4)

* IZ	* ZTR	*	10	7	13	-12	7
-31 NEG	23 POS	14	-13	3	4	6	-1
15 ZTR	14 ZTR	3 ZTR	-12 ZTR	5	-7	3	9
-9 ZTR	-7 ZTR	-14 ZTR	8 ZTR	4	-2	3	2
--5	9	-1	*	4	6	-2	2
3	0	-3	2	3	-2	0	4
2	-3	6	-4	3	6	3	6
5	11	5	6	0	3	-4	4

After dominant_pass, the significant coefficients will be replaced by * or 0
Then the threshold is divided by 2, so we have **16** as current threshold

EZW – An example(5)

The result from the second dominant_pass is output as the following:

IZ, ZTR, NEG, POS, ZTR, ZTR, ZTR, ZTR, ZTR, ZTR, ZTR, ZTR

The significant coefficients are put in the subordinate list and all data in this list will be refined as:

ABS(Original data)	63-32	34-32	49-32	47-32	-31	23
Output symbol	1	0	0	1	1	0
ABS(Reconstructed data)	60	36	52	44	28	20

For example, the output for 63 is:

sign	32	16	8	4	2	1
0	1	1	1	?	?	?

The computation is now extended with respect to the next significant bit. So 63 will be reconstructed as the average of 56 and 64 –- **60**!

EZW – An example(6)

The process is going on until threshold =1, the final output as:

D1: pnzt pttt tztt tttt tptt
S1: 1010
D2: ztnptttttttt
S2: 100110
D3: zzzzzppnppnttnnptpttnttttttttptttptttttttttptttttttttttt
S3: 10011101111011011000
D4: zzzzzzztztznzzzzpttptpptpnptntttttptpnpppptttttptptttpnp
S4: 11011111011001000001110110100010010101100
D5: zzzzztzzzzztpzzzttpttttnptppttptttnppnttttpnnpttpttppttt
S5: 10111100110100010111110101101100100000000110110110011000111
D6: zzzttztttztttttnnttt
Here p=pos, n=neg, z=iz, t=ztr The encoding used is : *POS—01, NEG—11, ZTR—00, IZ--10*

For example, the output for 63 is:

sign	32	16	8	4	2	1
0	1	1	1	1	1	1

So 63 will be reconstructed as 32+16+8+4+2+1=63!
Note, how progressive transmission can be done.

Wavelet Theory: Conclusions

- Wavelets are a useful tool, better than the Fourier transform in many applications.
- Most powerful aspect of wavelets is their inherent multi-resolution analysis of signals and images.
- Wavelets have popularized multi-resolution approaches.
- Very active field, many conferences, special journal issues, everyone is trying wavelets.
- Applications are wide-ranging.
  - Signal processing, data compression, computer vision, computer graphics, quantum physics, fast numerical algorithms.