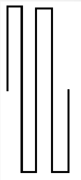
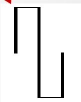
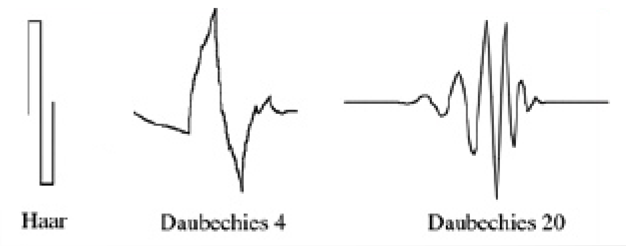
Discrete Wavelet Transform

- What is wavelet?
- A wavelet is a waveform of effectively limited duration.
- Sine waves do not have limited duration which extend from minus to plus infinity. Sine waves are smooth and predictable, wavelets tend to be irregular and asymmetric.
- Wavelet analysis break up a signal into shifted and scaled versions of the wavelet. Fourier analysis break up a signal into sine waves of various frequencies.
- Intuitively, signals with sharp changes may be better analyzed by irregular wavelet than smooth sinusoid.







Fourier transform

- Fourier transform represents signals as the sum of sine and cosines that have infinite duration.

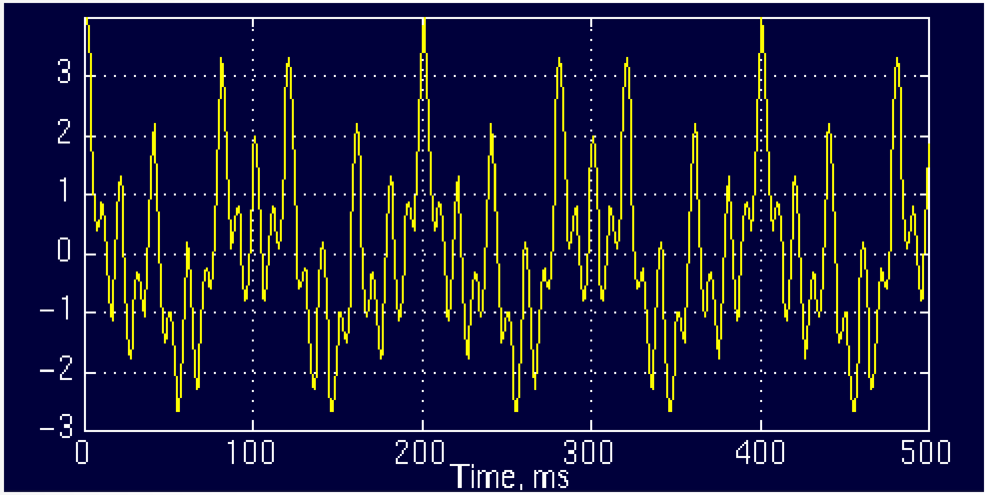
$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-j\omega t} dt$$

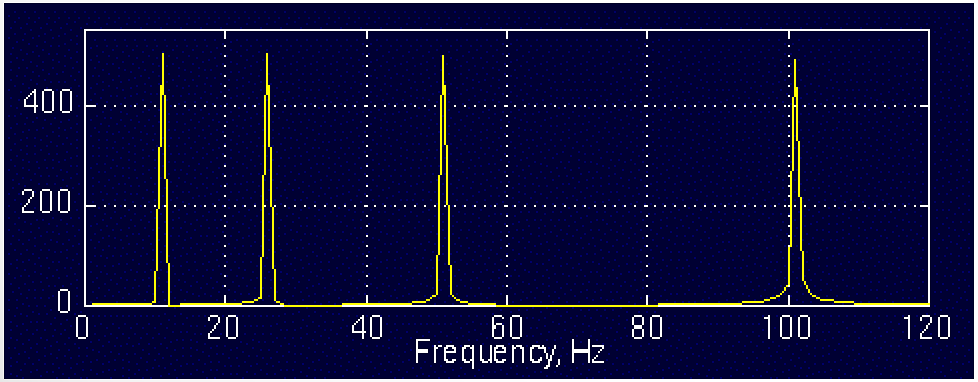
- It requires complete past and future signal to determine the frequency ω.
- For example the following signal is a stationary signal, because it has frequencies of 10, 25, 50, and 100 Hz at any given time instant.

f(t)=cos(2*π*10*t)+cos(2*π*25*t)+cos(2*π*50*t)+cos(2*π*100*t)

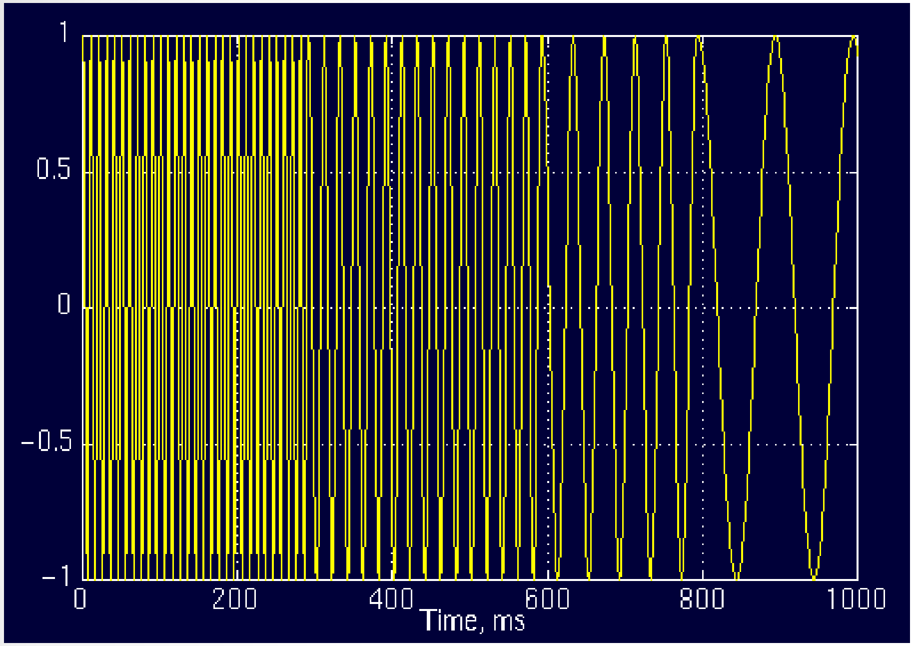
Fourier transform

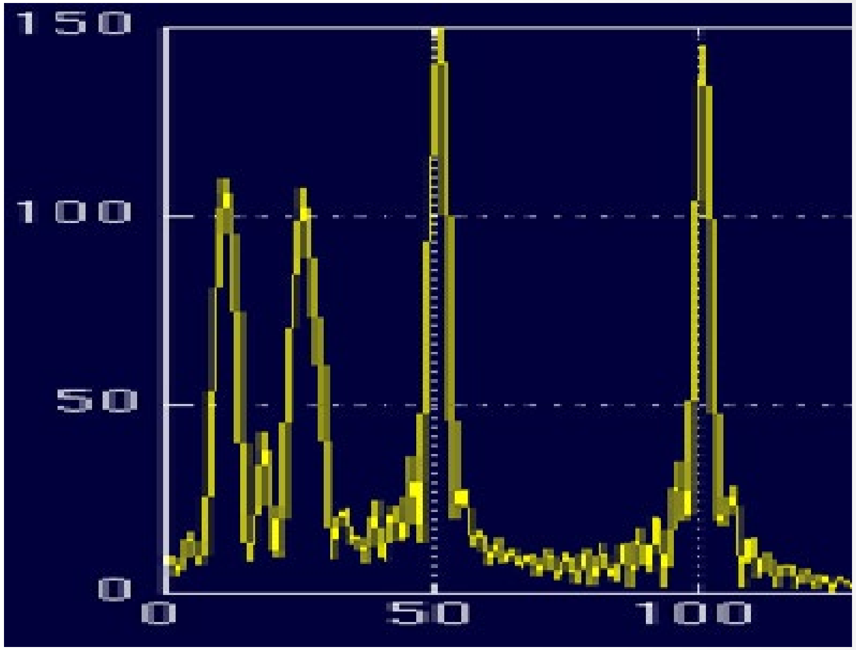
- This signal is plotted below: f(t)=cos(2*π*10*t)+cos(2*π*25*t)+cos(2*π*50*t)+cos(2*π*100*t)





Fourier transform

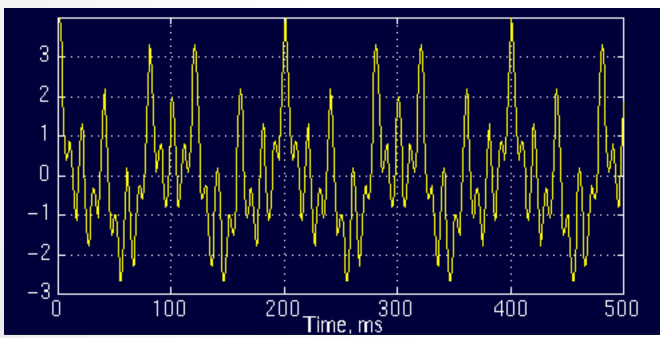


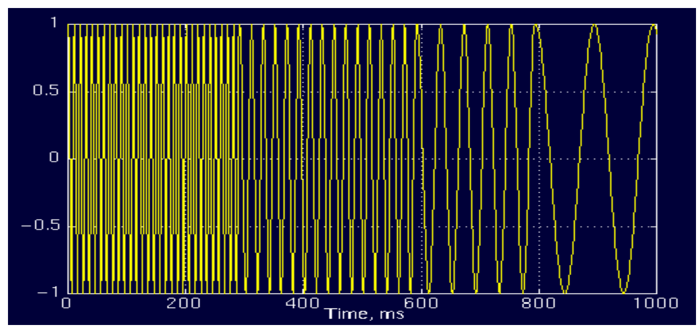


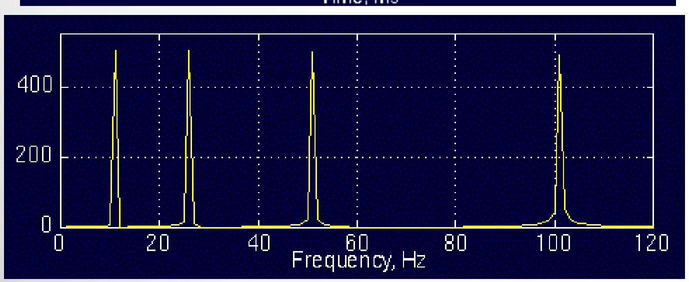
- For non-stationary signal, whose properties (frequency constantly) change with time. In the first interval we have the highest frequency component, and in the last interval we have the lowest frequency component.
- FT and DFT is a function independent of time, it does not reflect frequency that vary with time.

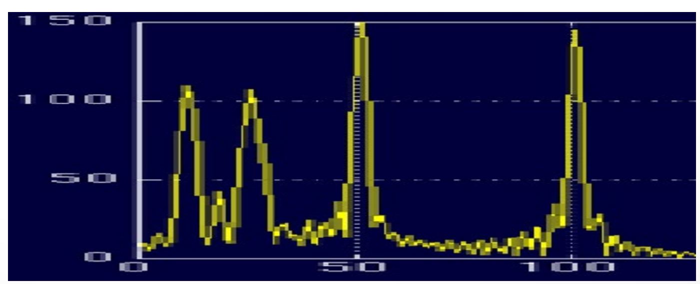
Fourier transform

- Both of the signals involves the same frequency components, but the first one has these frequencies at all times, the second one has these frequencies at different intervals.
- So, how come the spectrums of two entirely different signals look very much alike? FT gives the spectral content of the signal, but it doesn't provide information regarding when those spectral components appear.









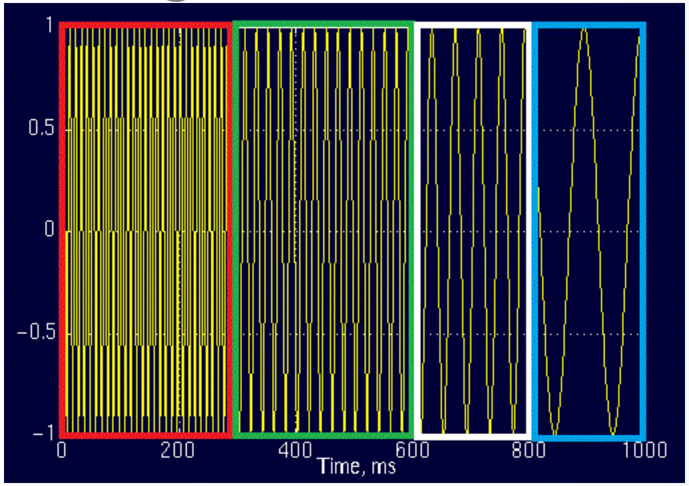
**** FT gives what frequency components exist in the signal. Nothing more, nothing less.**

Short-time Fourier transform

- Short-time Fourier transform (STFT) used to determine the sinusoidal frequency and phase content of locality (short time) of a signal as it changes over time.

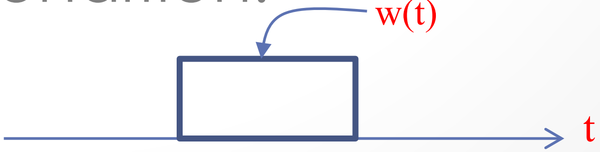
$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-j\omega t} dt$$

$$F(\omega, \underline{\tau}) = \int_{-\infty}^{\infty} f(t) e^{-j\omega t} w(t - \underline{\tau}) dt$$



- τ is a shift parameter, and w is a windows function subject to the following condition.

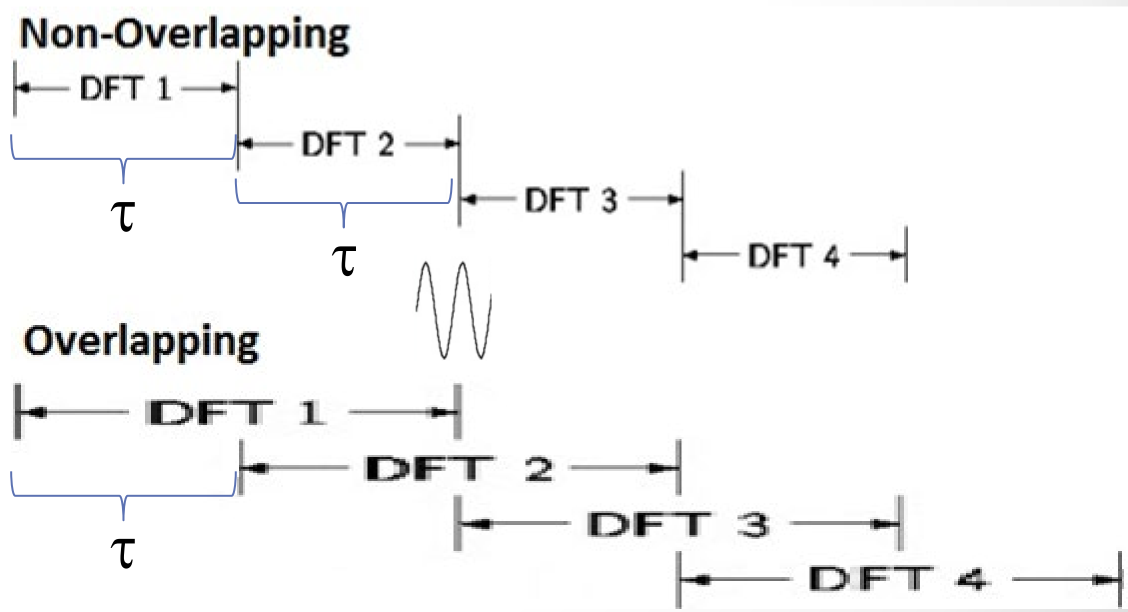
$$\int_{-\infty}^{\infty} w(t) dt = 1$$



Short-time Fourier transform

- The window is shifted along the time axis at various location of τ by multiplying the signal f(t) to calculate the STFT.

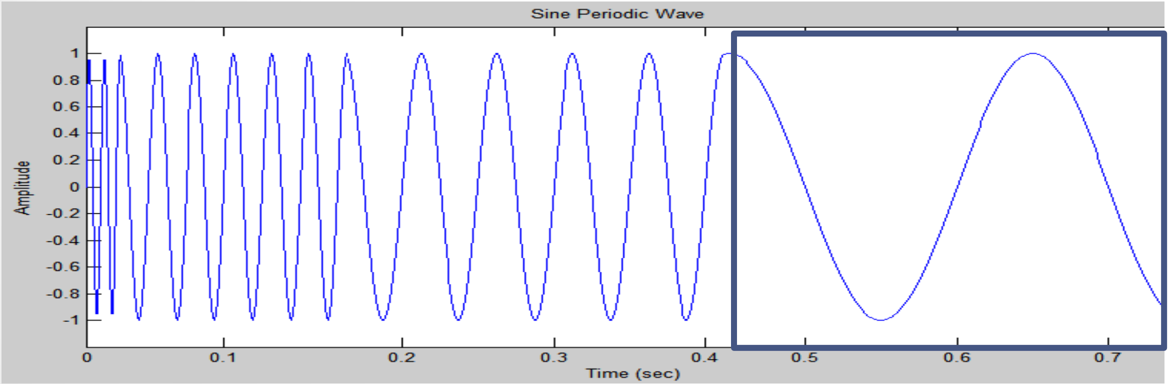
$$F(\omega,\tau)=\int_{-\infty}^{\infty} f(t)e^{-j\omega t}w(t-\tau)dt$$

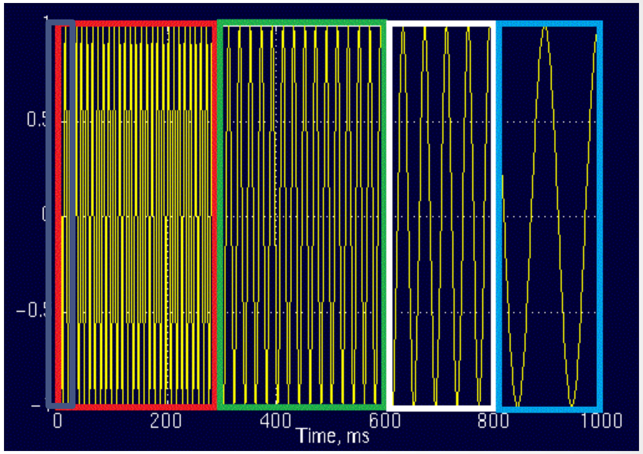


- If the window size is larger than τ, --> overlapping

Short-time Fourier transform

- As the window size is fixed, STFT cannot adapted to the changing characteristic of the signal. => STFT is not an ideal method for non-stationary signal.





- For lower frequency signal, the signal period is longer than the window size. Windows size is not long enough for low frequency signal.
- The limitation of STFT is that time frequency can't be measured at high precision at any time.
- High frequency cannot be localized to very large time window.

Short-time Fourier transform

- The time varying signal is better represented by a sum of basis function that localized in time.

  => wavelet analysis
- Wavelet analysis use short windows at high frequencies and long windows for low frequencies.
- In wavelet analysis, a basic function (analyzing function) called basic wavelet or mother wavelet is used as a window.
- Wavelet are generated by translating (shifting) and dilating (scaling) the basic wavelet.

Wavelet Transform

- Continuous wavelet transform is defined as follows,

$$CWT(a,\tau) = \int_{-\infty}^{\infty} f(t) \frac{1}{\sqrt{a}} \psi\left(\frac{t-\tau}{a}\right) dt \qquad F(\omega,\tau) = \int_{-\infty}^{\infty} f(t) e^{-j\omega t} w(t-\tau) dt$$

- Ψ is a basic wavelet, τ is a shift parameter and a is a scale parameter.
- The basic wavelet generates a set of wavelet basis functions

$$\psi_{a,\tau}(t) = \frac{1}{\sqrt{a}} \psi\left(\frac{t-\tau}{a}\right)$$

Discrete Wavelet Transform

- Discrete wavelet transform (DWT) is defined as

$$DWT(j,k)=\int_{-\infty}^{\infty} f(t)\psi_{j,k}(t)dt \qquad CWT(a,\tau)=\int_{-\infty}^{\infty} f(t)\frac{1}{\sqrt{a}}\psi\left(\frac{t-\tau}{a}\right)dt$$

- where

$$\psi_{j,k}(t)=\frac{1}{\sqrt{a^j}}\psi\left(\frac{t-ka^j}{a^j}\right) \qquad \psi_{a,\tau}(t)=\frac{1}{\sqrt{a}}\psi\left(\frac{t-\tau}{a}\right)$$

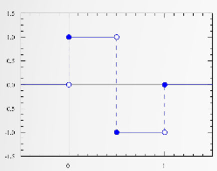
$$\Rightarrow \psi_{j,k}(t)=\frac{1}{\sqrt{a^j}}\psi\left(\frac{t}{a^j}-k\right)$$

j is scaling factor, k is shift factor that is shifted by $\tau = ka^j$

Discrete Wavelet Transform

$$\psi_{j,k}(t) = \frac{1}{\sqrt{a^j}}\,\psi\left(\frac{t - ka^j}{a^j}\right)$$

- Haar wavelet constitute the simplest wavelet. The mother wavelet is defined as



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

$$\psi_{j,k}(t) = \frac{1}{\sqrt{a^j}}\,\psi\left(\frac{t}{a^j} - k\right)$$

where a = 2

- The wavelet are defined as

$$k2^j + 2^j$$

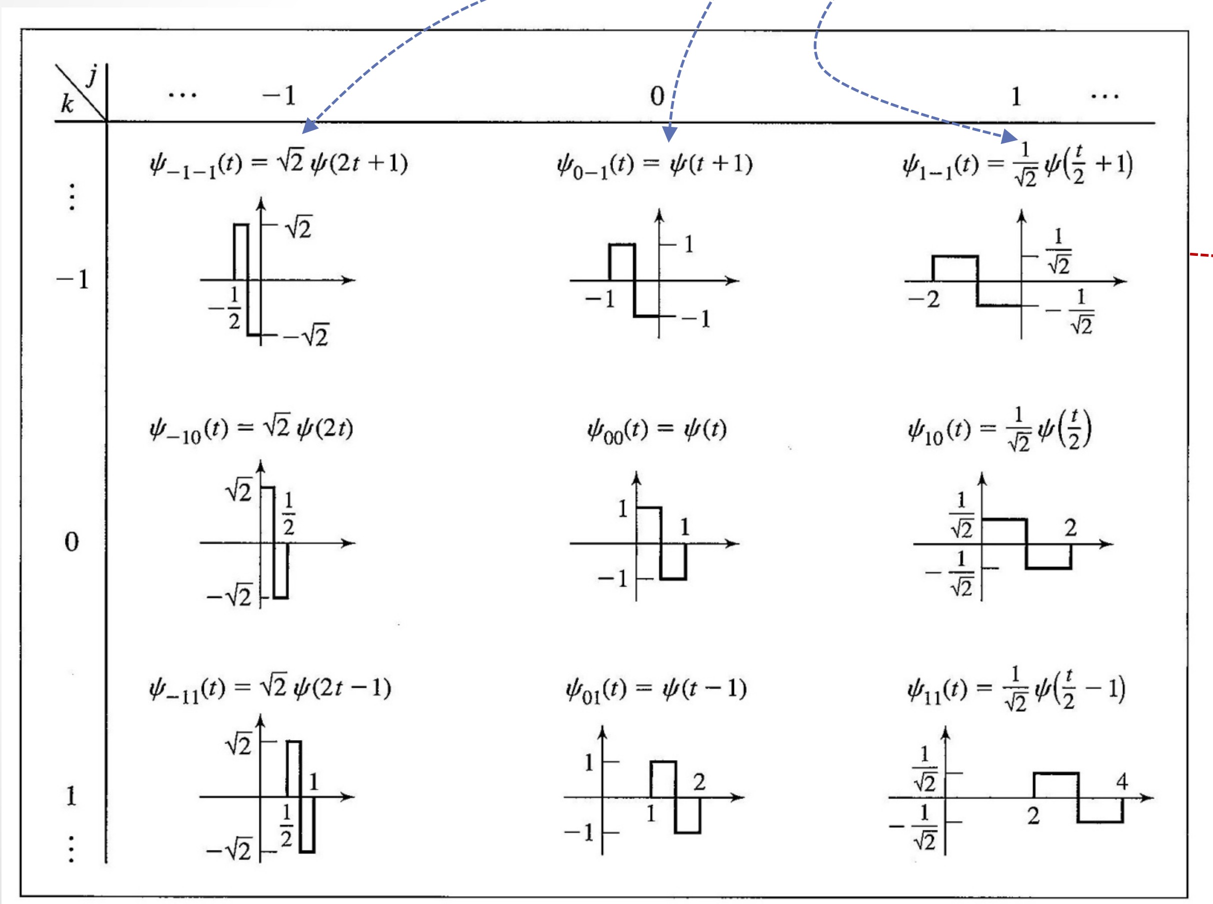
$$\psi_{j,k}(t) = 2^{-j/2}\,\psi_H\left(2^{-j}t - k\right) = \begin{cases} 1 & k2^j \le t < k2^j + 2^{j-1} \\ -1 & k2^j + 2^{j-1} \le t < (k+1)2^j \\ 0 & otherwise \end{cases}$$

j is scaling factor, k is shift factor that is shifted by $\tau = k2^j$

Discrete Wavelet Transform

- Haar wavelets

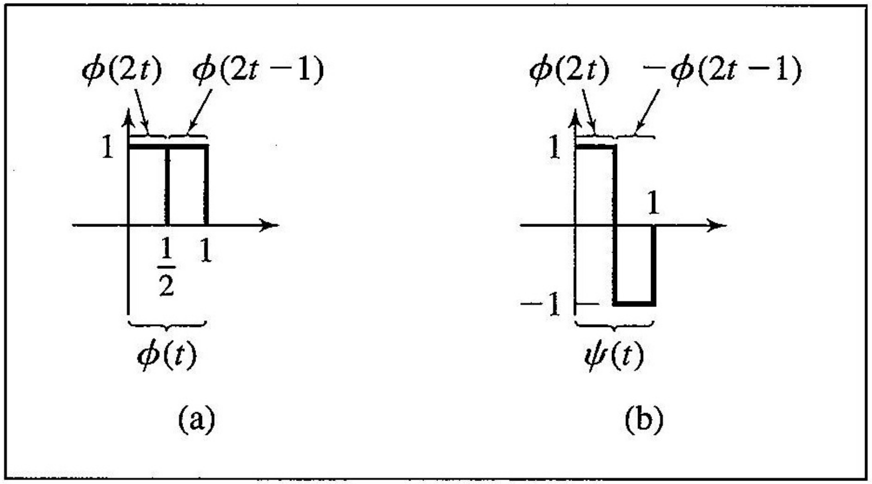
$$\psi_{j,k}(t) = \boxed{2^{-j/2}}\,\psi_H\left(2^{-j}t - k\right)$$



$$DWT(j,k) = \int_{-\infty}^{\infty} f(t)\,\boxed{\psi_{j,k}(t)}\,dt$$

Discrete Wavelet Transform

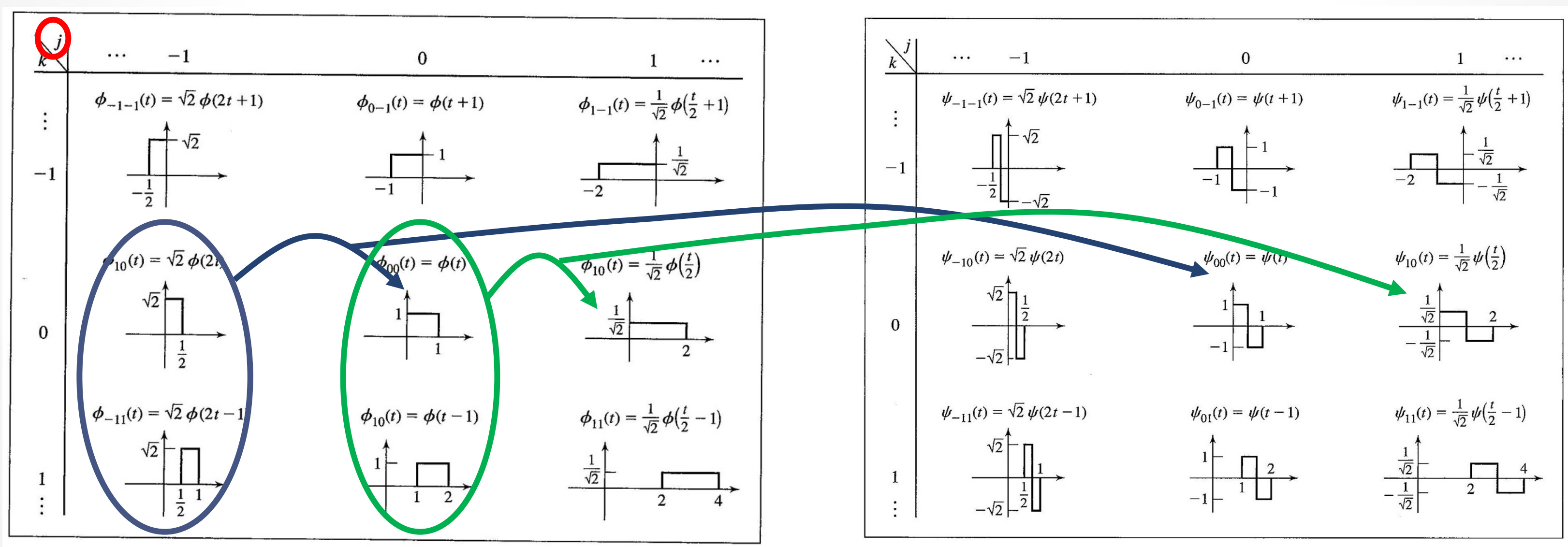
- The wavelet function can be viewed as a high-pass filter and scaling it for each level halves its bandwidth.
- The scaling function filters can be viewed as a low-pass filter which calculates a smoothed version of the data, and ensures all the spectrum is covered.



(a) Haar scaling function $\phi(t)$, (b) Haar wavelet $\psi(t)$.

Multiresolution Analysis

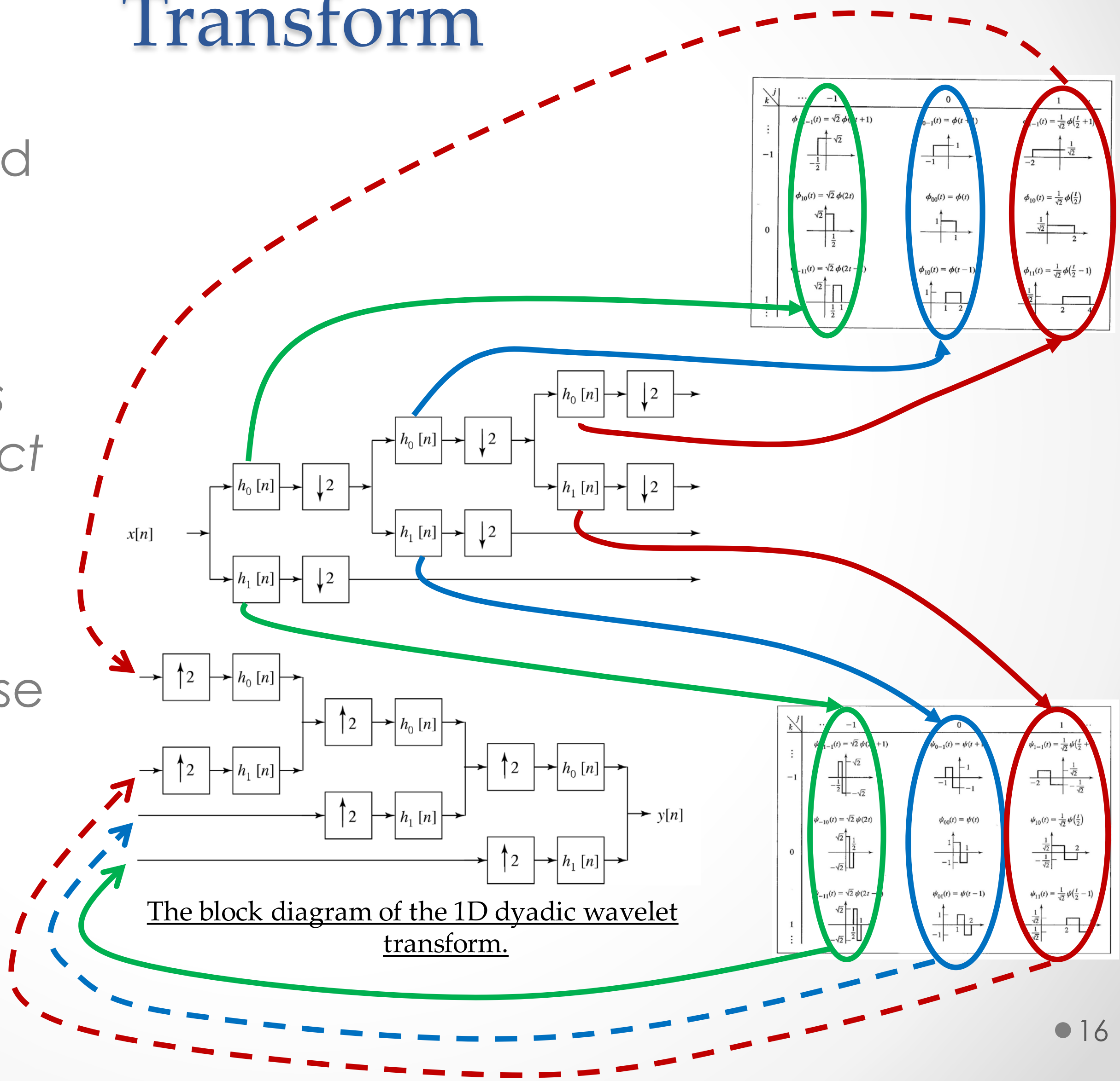
- Haar scaling functions



- Wavelets are set up such that the approximation at resolution 2^{j} contains all the necessary information to compute an approximation at coarse resolution $2^{(j+1)}$.

Block Diagram of 1D Dyadic Wavelet Transform

- The vectors $h_0[n]$ and $h_1[n]$ are called the low-pass (scaling functions) and high-pass (wavelets) *analysis* filters. To *reconstruct* the original input, an inverse operation is needed. The inverse filters are called *synthesis* filters.



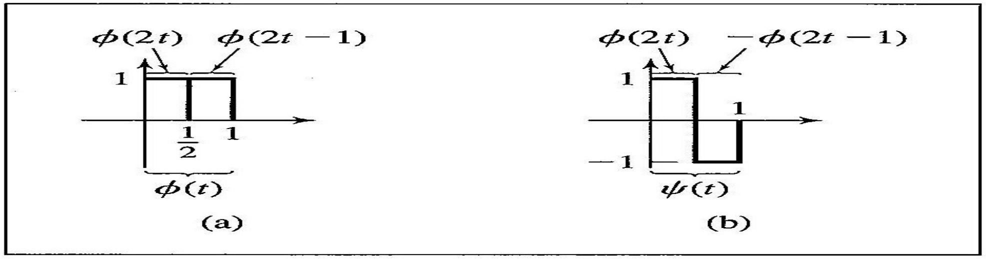
The block diagram of the 1D dyadic wavelet transform.

Haar Wavelet Transform Example

- Suppose we are given the following input sequence.

  $\{x_{n,i}\} = \{10, 13, 25, 26, 29, 21, 7, 15\}$

- Consider the transform that replaces the original sequence with its pairwise *average* $x_{n-1,i}$ and *difference* $d_{n-1,i}$ defined as follows:



$$x_{n-1,i} = \frac{x_{n,2i} + x_{n,2i+1}}{2} \qquad d_{n-1,i} = \frac{x_{n,2i} - x_{n,2i+1}}{2}$$

- The averages and differences are applied only on consecutive *pairs* of input sequences whose first element has an even index. Therefore, the number of elements in each set $\{x_{n-1,i}\}$ and $\{d_{n-1,i}\}$ is exactly half of the number of elements in the original sequence.

Haar Wavelet Transform Example

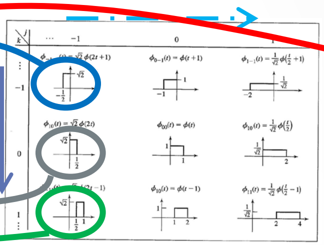
- Form a new sequence having length equal to that of the original sequence by concatenating the two sequences $\{x_{n-1,i}\}$ and $\{d_{n-1,i}\}$. The resulting sequence is

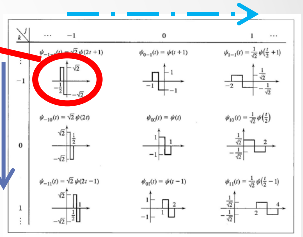
$\{x_{n,i}\} = \{10, 13, 25, 26, 29, 21, 7, 15\}$

$\{x_{n-1,i}, d_{n-1,i}\} = \{11.5, 25.5, 25, 11, -1.5, -0.5, 4, -4\}$

$\{x_{n-2,i}, d_{n-2,i}, d_{n-1,i}\} = \{18.5, 18, -7, 7, -1.5, -0.5, 4, -4\}$

$\{x_{n-3,i}, d_{n-3,i}, d_{n-2,i}, d_{n-1,i}\} = \{18.25, 0.25, -7, 7, -1.5, -0.5, 4, -4\}$





- This sequence has exactly the same number of elements as the input sequence — the transform did not increase the amount of data.
- Since the first half of the above sequence contain averages from the original sequence, we can view it as a coarser approximation to the original signal. The second half of this sequence can be viewed as the details or approximation errors of the first half.
- It is easily verified that the original sequence can be reconstructed from the transformed sequence using the relations

$$x_{n,2i} = x_{n-1,i} + d_{n-1,i}$$

$$x_{n,2i+1} = x_{n-1,i} - d_{n-1,i}$$

2D Haar Wavelet Transform

40	100	100	40
100	180	180	100
100	180	180	100
40	100	100	40

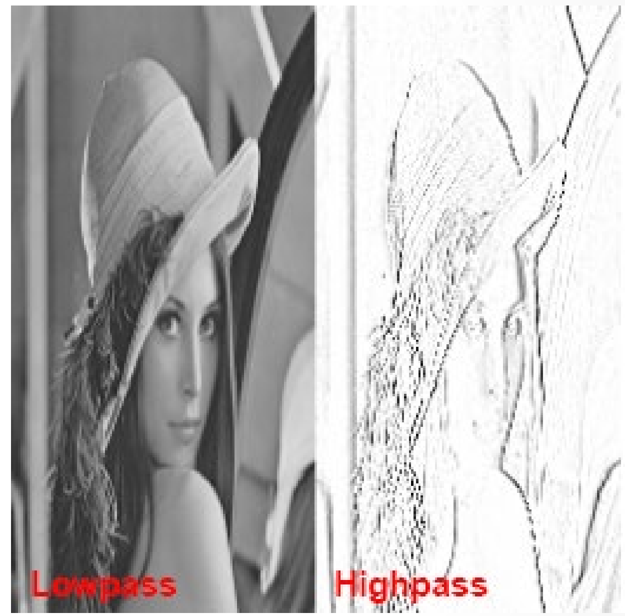
$\Rightarrow$

70	70	-30	30
140	140	-40	40
140	140	-40	40
70	70	-30	30

- Apply 1D row transforms for 4 rows (Decomposition on rows).

Example: Decomposition on rows

Original Image



Decomposition on rows
(decimation in column number)

2D Haar Wavelet Transform

70	70	-30	30
140	140	-40	40
140	140	-40	40
70	70	-30	30

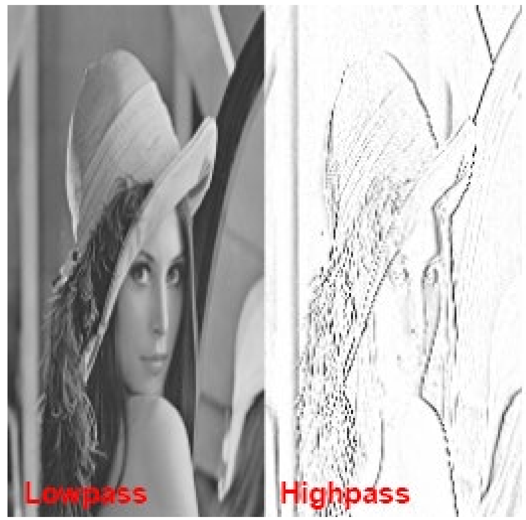
105	105	-35	35
105	105	-35	35
-35	-35	5	-5
35	35	-5	5

One-level decomposition

- Apply 1D column transforms for 4 columns.

2D Decomposition

Original Image



First decomposition on rows

Further decomposition on columns

2D Haar Wavelet Transform

105	0	-35	35
0	0	-35	35
-35	-35	5	-5
35	35	-5	5

Two-level decomposition

Coarse → Fine

LL

HL

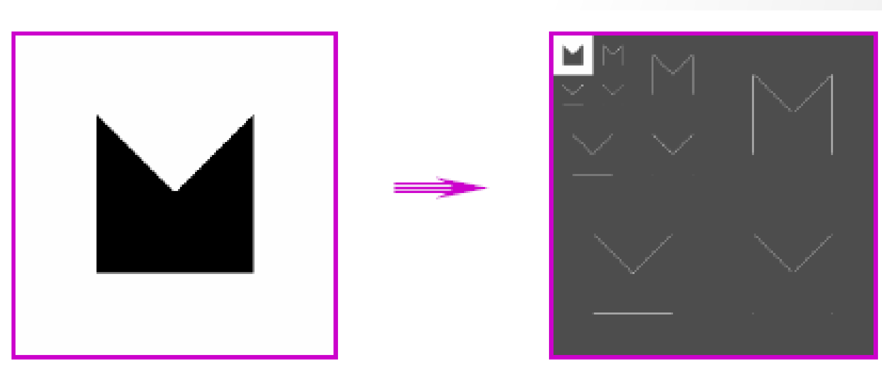
LH

HH

105	105	-35	35
105	105	-35	35
-35	-35	5	-5
35	35	-5	5

One-level decomposition

40	100	100	40
100	180	180	100
100	180	180	100
40	100	100	40



Basic Lifting Scheme

- Predict: $y'_0 = x_1 - x_0$ (difference => high pass)
- Update: $y_0 = (x_0 + x_1)/2 = x_0 + y'_0/2$ (average => low pass)

linear algebra **lifting scheme**

- What is the difference? Why lifting?

Temporary buffer

$a' = (a+b)/2$

$b' = b - a$

$a' \Rightarrow a$

$b' \Rightarrow b$

Lifting scheme (C program concept)

b = b - a

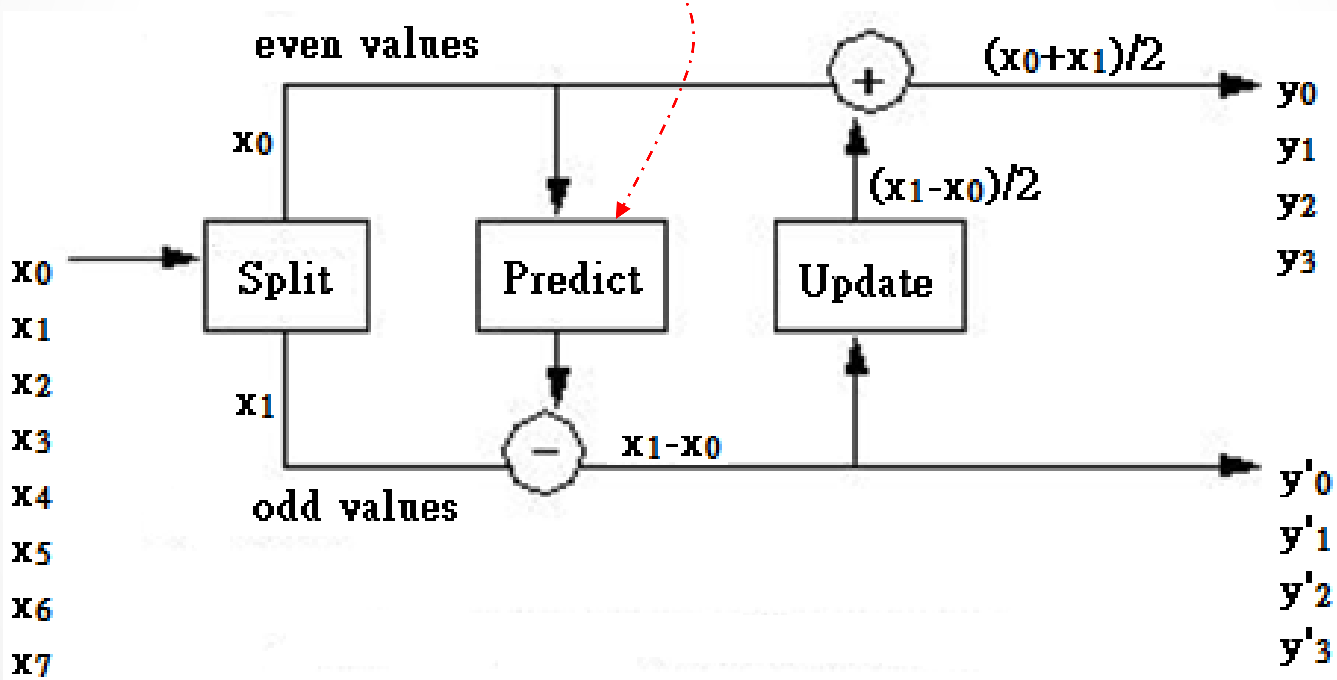
Update high pass, follow by low pass

No buffer

a = a + b / 2

Basic Lifting Scheme

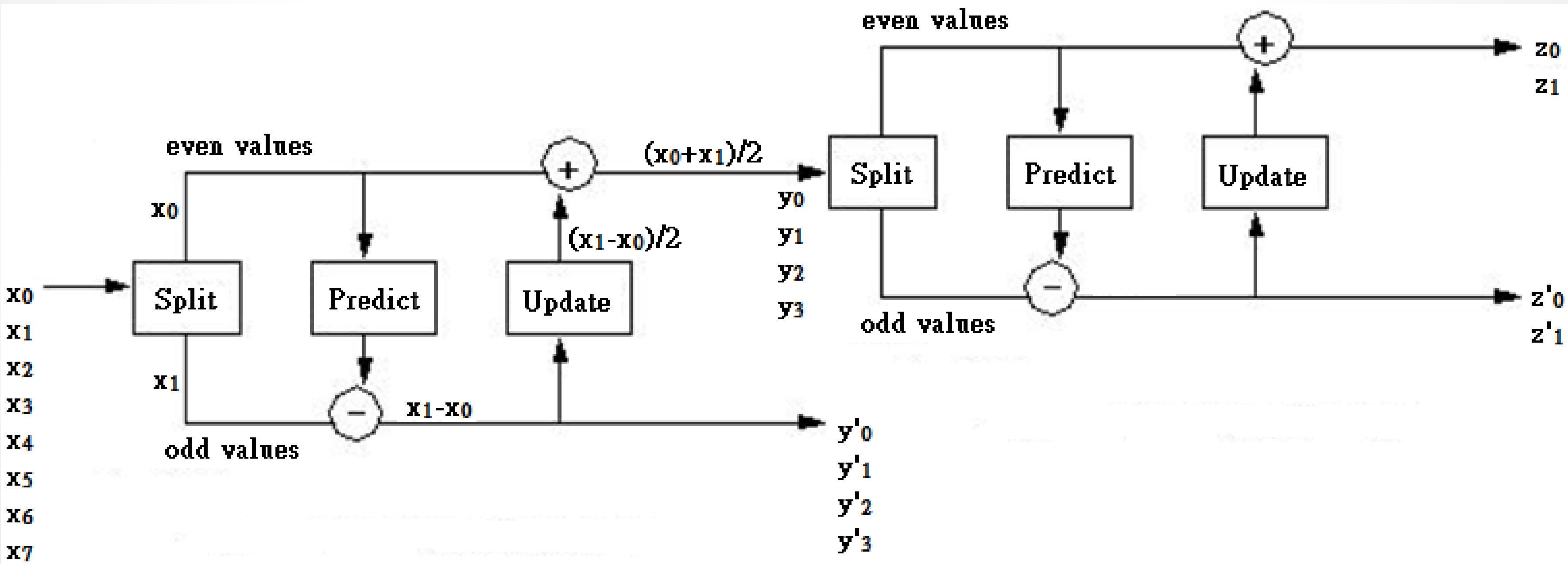
predict the next value equal to the current value, so predictor is x0



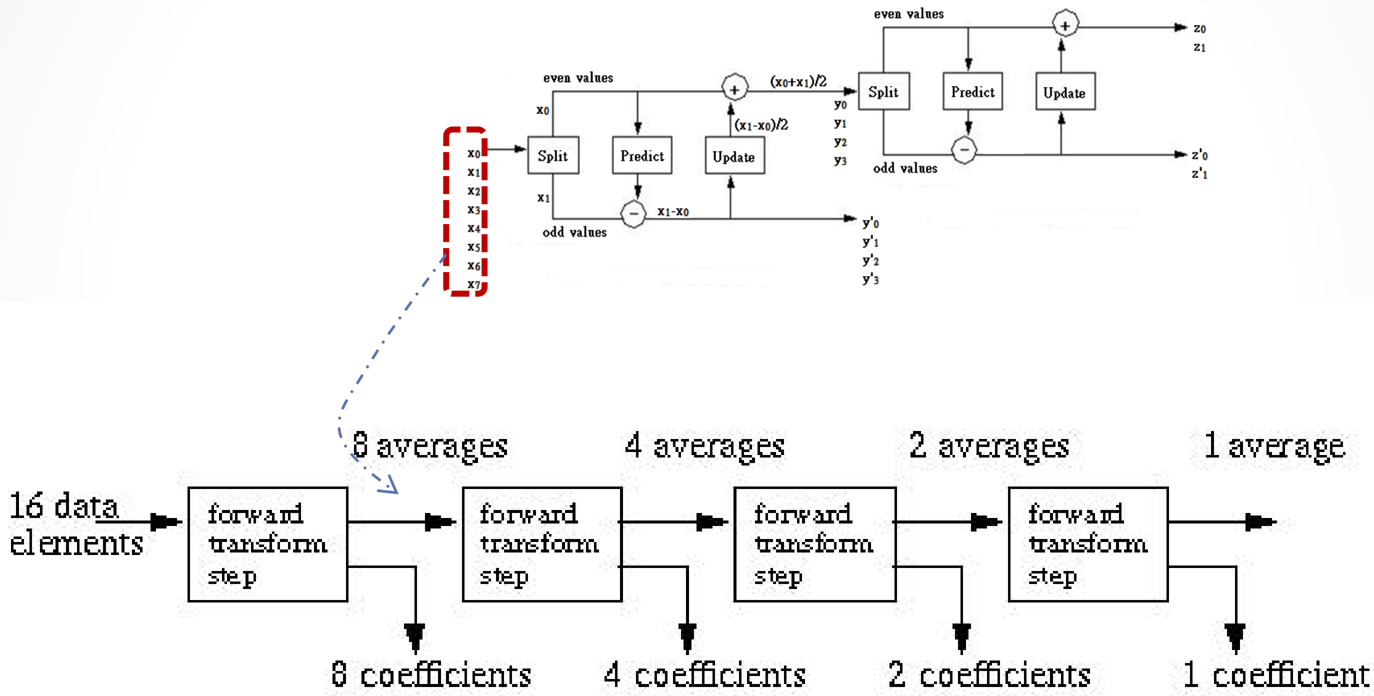
$y_0 = (x_o + x_1)/2$
$= x_0 + y'_0/2$

$y'_0 = x_1 - x_0$

Lifting Scheme Haar Transform

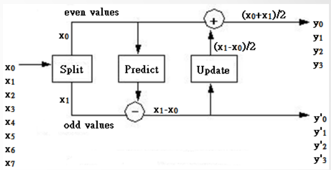


Lifting Scheme Haar Transform



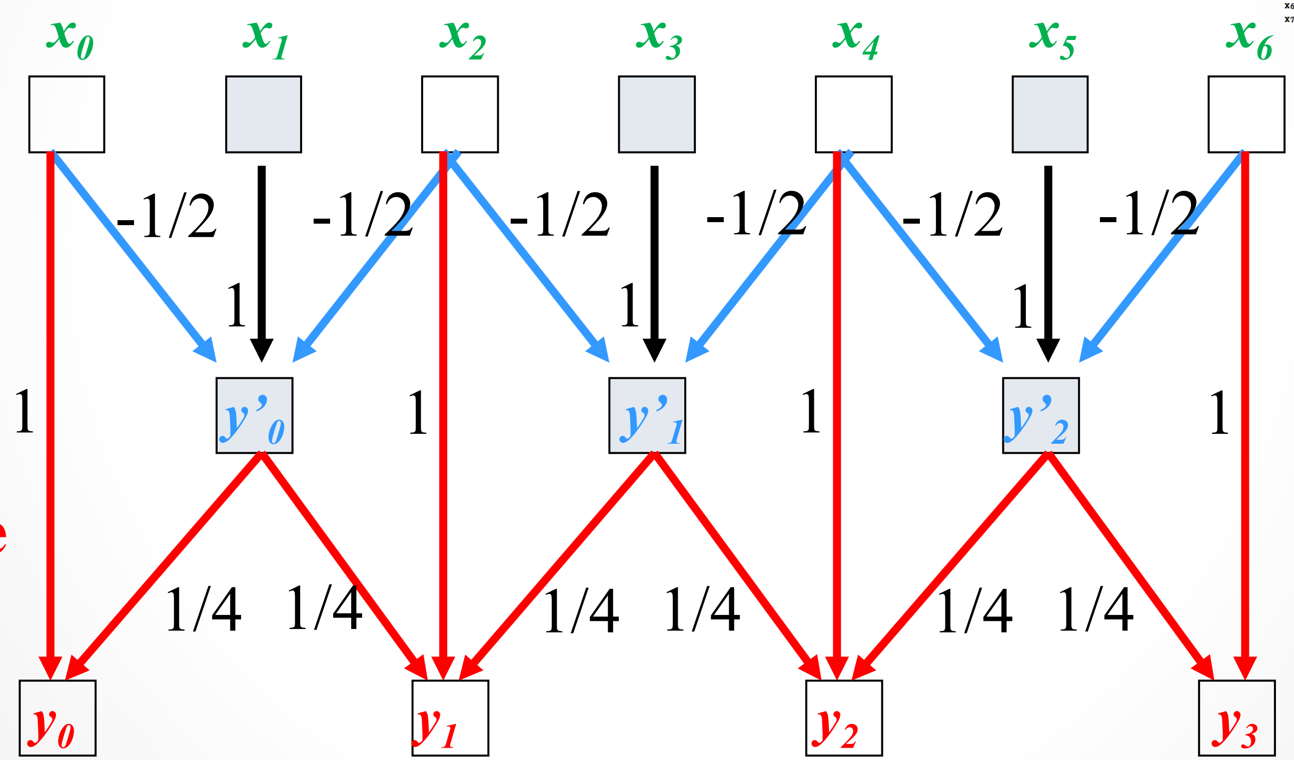
4 steps of a 16 element wavelet transform

5/3 Lifting Discrete Wavelet Transform (DWT)



Predict step

Update step



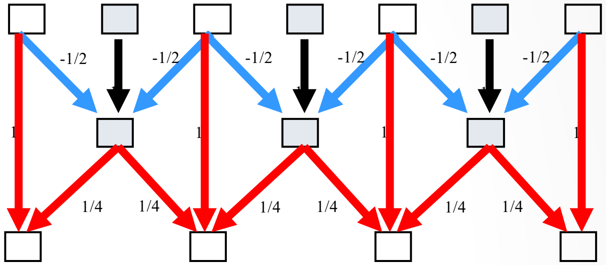
High-pass subband coefficients

Low-pass subband coefficients

5/3 Lifting DWT

- The linear interpolation function "predicts" that an odd element will be located at the mid-point of a line between its two even neighbours.

  predictor $= (x_2+x_0)/2$

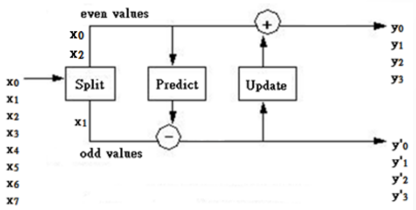


- The difference between the predicted value and the actual value of the odd element replaces the odd element.

  $y'_0 = x_1 - (x_2+x_0)/2$

  $= -x_0/2 + x_1 - x_2/2$

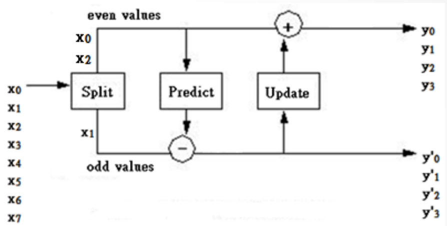
  (high pass)



5/3 Lifting DWT

- Update function

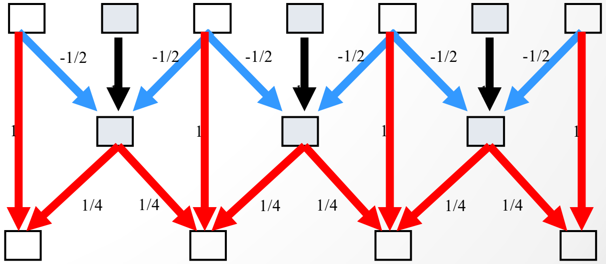
  $(y'_0+y'_1)/4$



- Output

  $$y1 = x2 + (y'_0+y'_1)/4$$
  $$= -x_0/8 + x_1/4 + 3x_2/4 + x_3/4 – x_4/8$$

  (low pass)



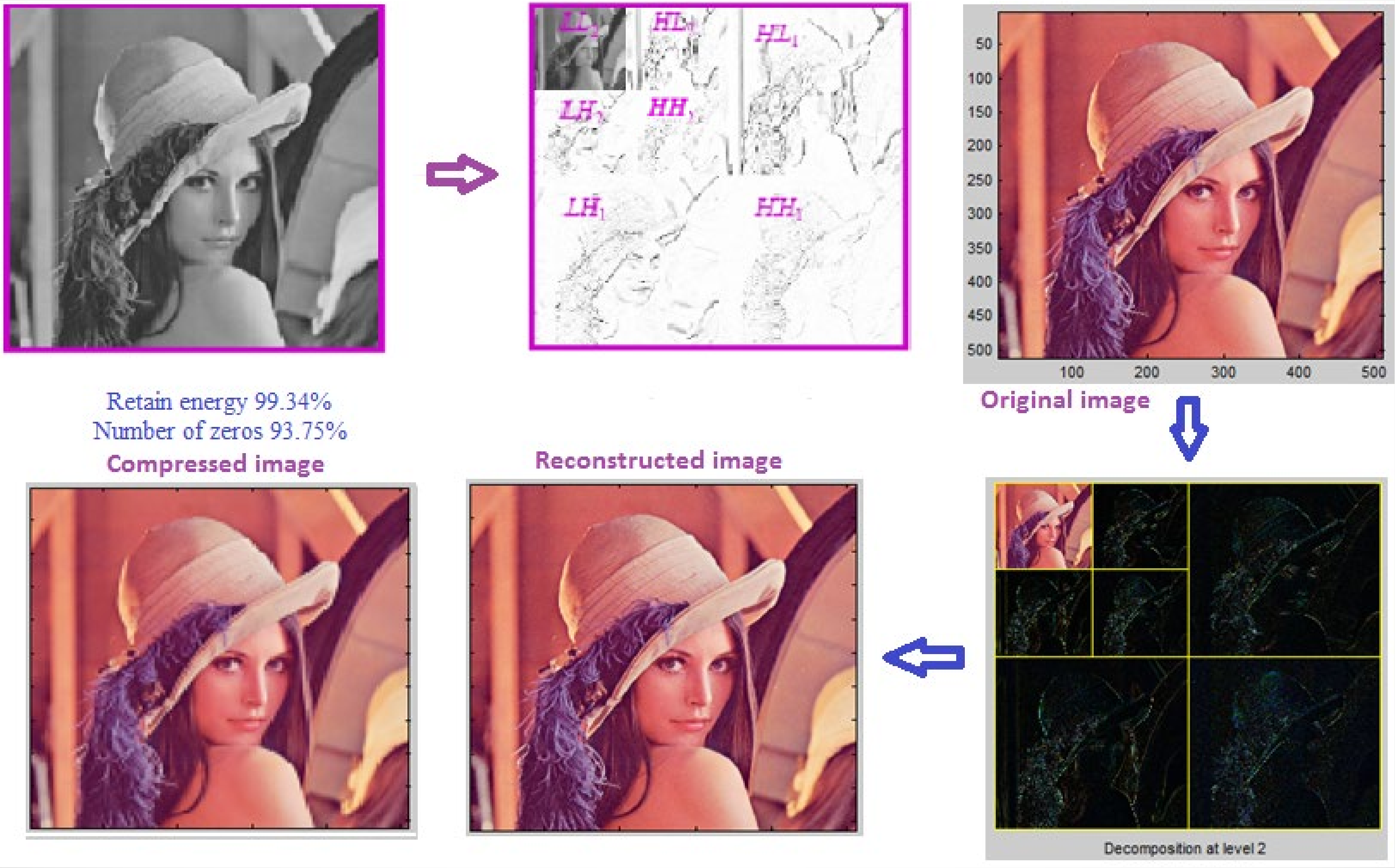
5/3 Lifting DWT

- How about y'_3?
- There is no $\text{even}_{j,i+1}$ element that brackets this element. For this last element the predict step "predicts" that the odd element lines on a line defined by its two even predecessors.

  $x_4 - x_6 = x_6 - x_8 \Rightarrow x_8 = 2x_6 - x_4$ (x_8 is dummy)

  $y'_3 = x_7 - (x_8 + x_6)/2 = x_7 - (3x_6)/2 + x_4/2$

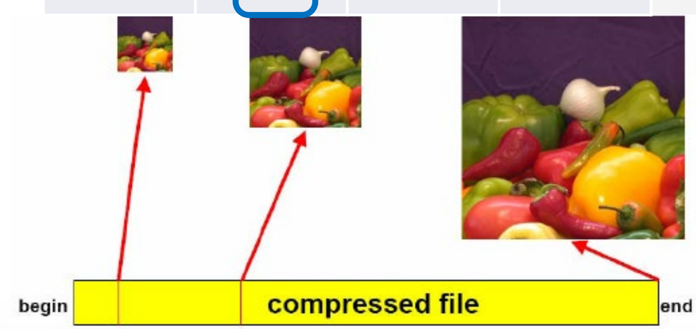
2D Haar Wavelet Transform

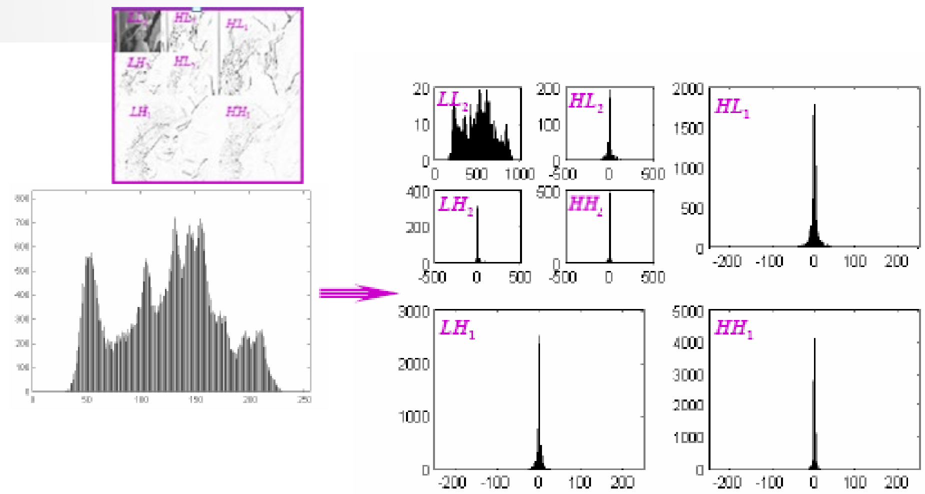


Embedded Zero-tree Wavelet (EZW) Coding

- How to encode it? Huffman? How about progressive encoding?
- Recall JPEG progressive mode
  - o *Spectral selection*: Send DC component and first few AC coefficients first, then gradually some more ACs.
  - o *Successive approximation*: send DCT coefficients MSB (most significant bit) to LSB (least significant bit).
- What if large coefficients in high frequency band?
- Instead of start from DC then ACs, let start from large coefficients, and increase resolution progressively.

	DC	AC1	AC2
MSB			
Bit 6			
LSB			





Histogram of original Lena

Histograms of subbands

Embedded Zero-tree Wavelet (EZW) Coding

- The EZW encoder is based on two important observations:
  - Natural images in general have a low pass spectrum. When an image is wavelet transformed the energy in the subbands decreases as the scale decreases (low scale means high resolution), so the wavelet coefficients will be smaller in the higher subbands than in the lower subbands.
    - Progressive encoding is a very natural choice for compressing wavelet transformed images, since the higher subbands only add detail.
  - Large wavelet coefficients are more important than smaller wavelet coefficients.

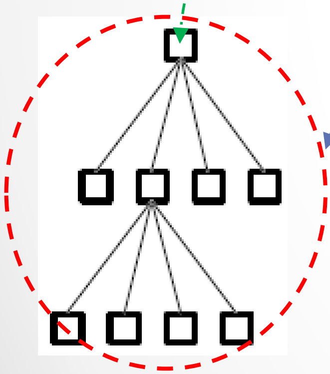
Embedded Zerotree Wavelet (EZW) Coding

- **E**mbedded
  - EZW encoder is a progressive encoder to represent an image in bitstream form with progressively increasing accuracy.
  - This progressive encoding is also known as embedded encoding.
- **Z**erotree
  - A data structure called zerotree is used in EZW algorithm to encode the data.
- **W**avelet
  - Wavelet transform a 2D signal especially for image.

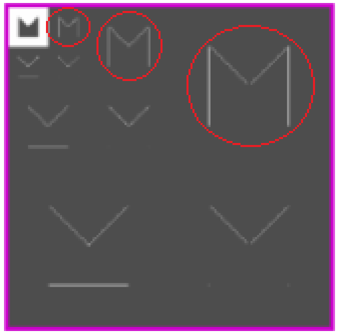
Zerotree

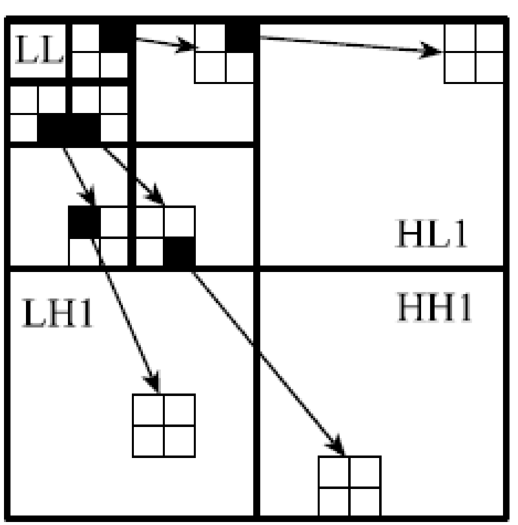
- Zerotree is a quad-tree.
- Hypothesis: if a wavelet coefficient at a coarse scale is insignificant with respect to a threshold, then all wavelet coefficients of the same orientation in the same spatial location at the finer scale are likely to be insignificant.

Root of a zerotree (T).



All coefficients are insignificant or zero, then this is a zerotree.



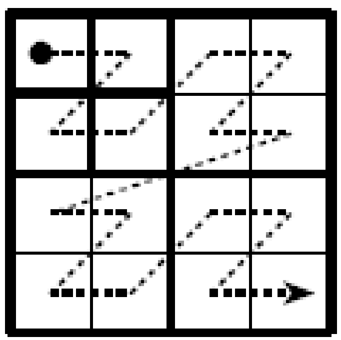


A coefficient in a low subband can be considered as having four descendants in the next higher subband.

- Code not only the coefficient values, but also their position.

Zerotree coding

Scanning order



H	H	H	L
L	H	L	H
L	L	H	L
L	L	H	L

H means coefficients higher than or equal to threshold.
L means coefficients lower than threshold .
T means root of zerotree

- Standard coding

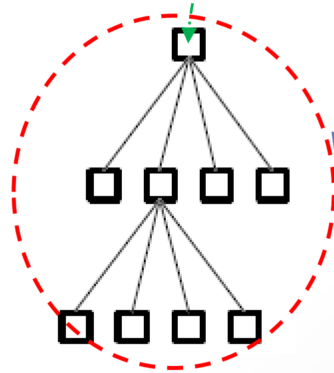
  HHLH HLLH LLLL HLHL

- Zerotree coding

  HHTH HLLH HLHL

It's insignificant and it's four descendants also insignificant.

Root of a zerotree (T).



All coefficients are insignificant or zero, then this is a zerotree.

Zerotree coding

- Wavelet coefficient could be represented in one of the four data types:
  - T: root of zerotree (it is insignificant but its descendant is also insignificant)
  - Z: isolated zero (it is insignificant but its descendant is not)
  - P: positive significant
  - N: negative significant

EZW

- Initial threshold $T_0 = 2^{\left\lfloor \log_2\left(\max\left(abs\left(w_{x,y}\right)\right)\right)\right\rfloor}$
- What is inside the algorithm?

```
threshold = initial_threshold T0;
 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.

Dominant Pass

- All the coefficients are scanned in a special order
- If the coefficient is a zero tree root, it will be encoded as T. 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 Z (isolated zero).
- If the coefficient is significant then it is encoded as P (positive) or N (negative) depends on its sign.
- Use ZTTT if no coefficient is significant in current dominant pass.
- All the coefficients that are in absolute value larger than or equal to (>=) the current threshold are extracted and placed without their sign on the subordinate list.
- This encoding of the zero tree produces significant compression because gray level natural images result in DWT with many T symbols. Each T indicates that no more bits are needed for encoding the descendants of the corresponding coefficient.

Subordinate Pass

- All the values in the subordinate list are refined. It outputs next most significant bit of all the coefficients in the subordinate list.

EZW Example

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

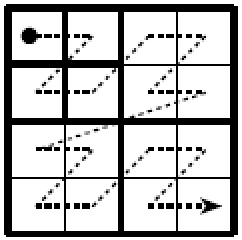
$$T_0 = 2^{\left\lfloor \log_2\left(\max\left(abs\left(w_{x,y}\right)\right)\right)\right\rfloor} = 32$$

EZW example -- first pass

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

D1: PNZT PTTT TZTT TTTT TPTT

S1: 1010



63	-34	49	47
1	1	1	1
1	0	1	0
1	0	0	1
1	0	0	1
1	1	0	1
1	0	1	1

EZW example – second pass

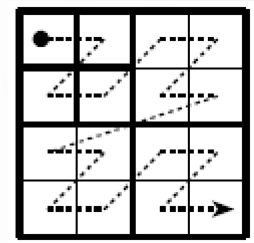
63	-34	49	10	7	13	-12	7
-31	23	14	-13	3	4	6	-1
15	14	3	-12	5	-7	3	9
-9	-7	-14	8	4	-2	3	2
-5	9	-1	47	4	6	-2	2
3	0	-3	2	3	-2	0	4
2	-3	6	-4	3	6	3	6
5	11	5	6	0	3	-4	4

D1: PNZT PTTT TZTT TTTT TPTT

S1: 1010

D2: ZTNP TTTT TTTT

S2: 100110



This is how progressive transmission can be done.

63	-34	49	47	-31	23
1	1	1	1		
1	0	1	0	1	1
1	0	0	1	1	0
1	0	0	1	1	1
1	1	0	1	1	1
1	0	1	1	1	1

EZW example

D1: PNZT PTTT TZTT TTTT TPTT *threshold 32

S1: 1010

D2: ZTNP TTTT TTTT *threshold 16

S2: 100110

D3: ZZZZ ZPPN PPNT TNNP TPTT NTTT TTTT TPTT TPTT TTTT TTTP TTTT TTTT TTTT *threshold 8

S3: 10011101111011011000

D4: ZZZZ ZZZT ZTZN ZZZZ PTTP TPPT PNPT NTTT TTPT PNPP PPTT TTTP TPTT TPNP *threshold 4

S4: 11011111011001000001110110100010010101100

D5: ZZZZ ZTZZ ZZZT PZZZ TTPT TTTN PTPP TTPT TTNP PNTT TTPN NPTT PTTP PTTT *threshold 2

S5: 10111100110100010111110101101100100000000110110110011000111

D6: ZZZT TZTT TZTT TTTN NTTT

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

Huffman code to represent the symbol:
T: 0
Z: 10
N: 110
P: 1110

Last level only encode 1 and -1,
No subordinate pass S6 is required.
‘0” is default for left over coefficients.

Reference

- Matlab wavelet toolbox.
- Adam Drozdek, Elements of data compression
- Z.N. Li and M.S. Drew, Fundamentals of multimedia,
- G. Strang and T. Nguyen, Wavelets and filter banks,
- Algazi, V. R. and R.R. Estes, Analysis based coding of image transform and subband coefficients, proceedings of the SPIE, vol. 2564 (1995), p. 11-21.
- C. D. Creusere, A new method of robust image compression based on the embedded zerotree wavelet algorithm, ieee transactions on image processing, vol. 6, no. 10 (1997), p. 1436-1442.
- J. M. Shapiro, Embedded image coding using zerotrees of wavelet coefficients, ieee transactions on signal processing, vol. 41, no. 12 (1993), p. 3445-3462.