Block Transform Coding

Section 8.2.8

Transform coding

Blocking artifact: boundaries between subimages become visible
Transform Selection

- DFT
- Discrete Cosine Transform (DCT)
- Wavelet transform
- Karhunen-Loeve Transform (KLT)

\[ T(u,v) = \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} g(x,y) r(x,y,u,v) \quad (8.2.10) \]

\[ g(x,y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} T(u,v) s(x,y,u,v) \quad (8.2.11) \]

Transform Kernels

- Separable if \( r(x,y,u,v) = r_1(x,u)r_2(y,v) \quad (8.2.12) \)
- E.g. DFT: \( r(x,y,u,v) = \exp(-j2\pi(ux + vy) / n) \)
- E.g. Walsh-Hadamard transform (see page 568, text)
- E.g. DCT
Approximations using DFT, Hadamard and DCT, and the scaled error images

**Discrete Cosine Transform**

1-D Case: Extended 2N Point Sequence

Consider 1-D first; Let \( x(n) \) be a N point sequence \( 0 \leq n \leq N - 1 \).

\[
x(n) \quad \leftrightarrow \quad 2 \text{ - } N \text{ point } \quad DFT \quad 2 \text{ - } N \text{ point } \quad Y(u) \quad \leftrightarrow \quad N \text{ - point } \quad C(u)
\]

\[
y(n) = x(n) + x(2N - 1 - n) = \begin{cases} 
  x(n), & 0 \leq n \leq N - 1 \\
  x(2N - 1 - n), & N \leq n \leq 2N - 1
\end{cases}
\]

**DCT & DFT**

\[
Y(u) = \sum_{n=0}^{2N-1} y(n) \exp \left( -j \frac{2\pi}{2N} u n \right)
\]

\[
= \sum_{n=0}^{N-1} x(n) \exp \left( -j \frac{2\pi}{2N} u n \right) + \sum_{n=N}^{2N-1} x(2N - 1 - n) \exp \left( -j \frac{2\pi}{2N} u n \right)
\]

\[
= \sum_{n=0}^{N-1} x(n) \exp \left( -j \frac{2\pi}{2N} u n \right) + \sum_{m=0}^{N-1} x(m) \exp \left( -j \frac{2\pi}{2N} u (2N - 1 - m) \right)
\]

\[
= \exp \left( j \frac{\pi}{2N} u \right) \sum_{n=0}^{N-1} x(n) \exp \left( -j \frac{\pi}{2N} u - j \frac{2\pi}{2N} u n \right)
\]

\[
+ \exp \left( j \frac{\pi}{2N} u \right) \sum_{n=0}^{N-1} x(n) \exp \left( j \frac{\pi}{2N} u + j \frac{2\pi}{2N} u n \right)
\]

\[
= \exp \left( j \frac{\pi}{2N} u \right) \sum_{n=0}^{N-1} 2x(n) \cos \left( \frac{\pi}{2N} u (2n + 1) \right).
\]
The N-point DCT of \( x(n) \), \( C(u) \), is given by
\[
C(u) = \begin{cases} 
\exp \left( -j \frac{\pi u}{2N} \right) Y(u), & 0 \leq u \leq N-1 \\
0 & \text{otherwise.}
\end{cases}
\]

The unitary DCT transformations are:
\[
F(u) = \alpha(u) \sum_{n=0}^{N-1} f(n) \cos \left( \frac{\pi}{2N} (2n+1)u \right), \quad 0 \leq u \leq N-1, \text{ where}
\]
\[
\alpha(0) = \frac{1}{\sqrt{N}}, \quad \alpha(u) = \frac{2}{\sqrt{N}} \quad \text{for} \quad 1 \leq k \leq N-1.
\]

The inverse transformation is
\[
f(n) = \sum_{u=0}^{N-1} \alpha(u) F(u) \cos \left( \frac{\pi}{2N} (2n+1)u \right), \quad 0 \leq u \leq N-1.
\]

**Discrete Cosine Transform—in 2-D**

\[
C(u, v) = \alpha(u) \alpha(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y) \cos \left( \frac{(2x+1)u\pi}{2N} \right) \cos \left( \frac{(2y+1)v\pi}{2N} \right)
\]

for \( u, v = 0, 1, 2, \ldots, N-1 \), where
\[
\alpha(u) = \begin{cases} 
\sqrt{\frac{1}{N}} & \text{for} \quad u = 0 \\
\sqrt{\frac{2}{N}} & \text{for} \quad u = 1, 2, \ldots, N-1.
\end{cases}
\]

\[
f(x, y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \alpha(u) \alpha(v) C(u, v) \cos \left( \frac{(2x+1)u\pi}{2N} \right) \cos \left( \frac{(2y+1)v\pi}{2N} \right)
\]

for \( x, y = 0, 1, 2, \ldots, N-1 \).
DCT Summary

\[ T(u,v) = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) r(x,y,u,v) \]  
\[ r(x,y,u,v) = s(x,y,u,v) \]

\[ = \alpha(u)\alpha(v) \cos\left(\frac{(2x+1)u\pi}{2N}\right) \cos\left(\frac{(2y+1)v\pi}{2N}\right) \]  

\[ \text{.........8.2.18} \]

DCT Basis functions

**Figure 8.30** Discrete-cosine basis functions for \( N = 4 \). The origin of each block is at its top left.
Implicit Periodicity-DFT vs DCT (Fig 8.32)

**Why DCT?**

- Blocking artifacts less pronounced in DCT than in DFT.
- Good approximation to the Karhunen-Loeve Transform (KLT) but with basis vectors fixed.
- DCT is used in JPEG image compression standard.
Sub-image size selection (fig 8.26)

Different sub-image sizes

FIGURE 8.34 Approximations of Fig. 8.23 using 25% of the DCT coefficients: (a) and (b) 8 x 8 subimage results; (c) centered original; (d) 2 x 2 result; (e) 4 x 4 result; and (f) 8 x 8 result.
Bit Allocation/Threshold Coding

- # of coefficients to keep
- How to quantize them

**Threshold coding**

For each subimage i
- Arrange the transform coefficients in decreasing order of magnitude
- Keep only the top X% of the coefficients and discard rest.
- Code the retained coefficient using variable length code.

**Zonal Coding**

- Compute the variance of each of the transform coeff; use the subimages to compute this.
- Keep X% of their coeff. which have maximum variance.
- Variable length coding (proportional to variance)

Bit allocation: In general, let the number of bits allocated be made proportional to the variance of the coefficients. Suppose the total number of bits per block is B. Let the number of retained coefficients be M. Let \( v(i) \) be variance of the i-th coefficient. Then

\[
b(i) = \frac{B}{M} + \frac{1}{2} \log_2 v(i) - \frac{1}{2M} \sum_{i=1}^M \log_2 v(i)
\]
Zonal Mask & bit allocation: an example

Typical Masks (Fig 8.36)

 Typical (a) zonal mask, (b) zonal bit allocation, (c) threshold mask, and (d) thresholded coefficient ordering sequence. Shading highlights the coefficients that are retained.
Image Approximations

The JPEG standard

- The following is mostly from Tekalp’s book, Digital Video Processing by M. Tekalp (Prentice Hall).
- For the new JPEG-2000 check out the web site www.jpeg.org.
- See Example 8.17 in the text
JPEG (contd.)

- JPEG is a lossy compression standard using DCT.
- Four modes of operation: Sequential (baseline), hierarchical, progressive, and lossless.
- Arbitrary image sizes; DCT mode 8-12 bits/sample. Luminance and chrominance channels are separately encoded.
- We will only discuss the baseline method.

JPEG-baseline.

- DCT: The image is divided into 8x8 blocks. Each pixel is level shifted by $2^{n-1}$ where $2^n$ is the maximum number of gray levels in the image. Thus for 8 bit images, you subtract 128. Then the 2-D DCT of each block is computed. For the baseline system, the input and output data precision is restricted to 8 bits and the DCT values are restricted to 11 bits.
- Quantization: the DCT coefficients are threshold coded using a quantization matrix, and then reordered using zig-zag scanning to form a 1-D sequence.
- The non-zero AC coefficients are Huffman coded. The DC coefficients of each block are DPCM coded relative to the DC coefficient of the previous block.
JPEG -color image

- RGB to Y-Cr-Cb space
  - \( Y = 0.3R + 0.6G + 0.1B \)
  - \( Cr = 0.5 (B-Y) + 0.5\)
  - \( Cb = \frac{1}{1.6} (R-Y) + 0.5\)

- Chrominance samples are sub-sampled by 2 in both directions.

<table>
<thead>
<tr>
<th>Y1</th>
<th>Y2</th>
<th>Y3</th>
<th>Y4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y5</td>
<td>Y6</td>
<td>Y7</td>
<td>Y8</td>
</tr>
<tr>
<td>Y9</td>
<td>Y10</td>
<td>Y11</td>
<td>Y12</td>
</tr>
<tr>
<td>Y13</td>
<td>Y14</td>
<td>Y15</td>
<td>Y16</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cr1</th>
<th>Cr2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cr3</td>
<td>Cr4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cb1</th>
<th>Cb2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cb3</td>
<td>Cb4</td>
</tr>
</tbody>
</table>

Non-Interleaved
Scan 1: Y1, Y2, ..., Y16
Scan 2: Cr1, Cr2, Cr3, Cr4
Scan 3: Cb1, Cb2, Cb3, Cb4

Interleaved: Y1, Y2, Y3, Y4, Cr1, Cb1, Y5, Y6, Y7, Y8, Cr2, Cb2, ...

JPEG – quantization matrices

- Check out the matlab workspace (dctex.mat)
- Quantization table for the luminance channel.
- Quantization table for the chrominance channel.
- JPEG baseline method
  - Consider the 8x8 image (matlab: array s.)
  - Level shifted (s-128=sd).
  - 2d-DCT: dct2(sd)= dcts
  - After dividing by quantization matrix qmat: dctshat.
  - Zigzag scan as in threshold coding.

\[ [20, 5, -3, -1, -2, -3, 1, 1, -1, 0, 0, 1, 2, 3, -2, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, EOB]. \]
An 8x8 sub-image (s)

$$s = (8x8\text{block})$$

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>183</td>
<td>160</td>
<td>94</td>
<td>153</td>
<td>194</td>
<td>163</td>
<td>132</td>
<td>165</td>
<td></td>
<td></td>
</tr>
<tr>
<td>183</td>
<td>153</td>
<td>116</td>
<td>176</td>
<td>187</td>
<td>166</td>
<td>130</td>
<td>169</td>
<td></td>
<td></td>
</tr>
<tr>
<td>179</td>
<td>168</td>
<td>171</td>
<td>182</td>
<td>179</td>
<td>170</td>
<td>131</td>
<td>167</td>
<td></td>
<td></td>
</tr>
<tr>
<td>177</td>
<td>177</td>
<td>179</td>
<td>177</td>
<td>179</td>
<td>165</td>
<td>131</td>
<td>167</td>
<td></td>
<td></td>
</tr>
<tr>
<td>178</td>
<td>178</td>
<td>179</td>
<td>176</td>
<td>182</td>
<td>164</td>
<td>130</td>
<td>171</td>
<td></td>
<td></td>
</tr>
<tr>
<td>179</td>
<td>180</td>
<td>180</td>
<td>179</td>
<td>183</td>
<td>164</td>
<td>130</td>
<td>171</td>
<td></td>
<td></td>
</tr>
<tr>
<td>179</td>
<td>179</td>
<td>180</td>
<td>182</td>
<td>183</td>
<td>170</td>
<td>129</td>
<td>173</td>
<td></td>
<td></td>
</tr>
<tr>
<td>180</td>
<td>179</td>
<td>181</td>
<td>179</td>
<td>181</td>
<td>170</td>
<td>130</td>
<td>169</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

sd = (level shifted)

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>55</td>
<td>32</td>
<td>-34</td>
<td>25</td>
<td>66</td>
<td>35</td>
<td>4</td>
<td>37</td>
<td></td>
<td></td>
</tr>
<tr>
<td>55</td>
<td>25</td>
<td>-12</td>
<td>48</td>
<td>59</td>
<td>38</td>
<td>2</td>
<td>41</td>
<td></td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>40</td>
<td>43</td>
<td>54</td>
<td>51</td>
<td>42</td>
<td>3</td>
<td>39</td>
<td></td>
<td></td>
</tr>
<tr>
<td>49</td>
<td>49</td>
<td>51</td>
<td>49</td>
<td>51</td>
<td>37</td>
<td>3</td>
<td>39</td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>50</td>
<td>51</td>
<td>48</td>
<td>54</td>
<td>36</td>
<td>2</td>
<td>43</td>
<td></td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>52</td>
<td>52</td>
<td>51</td>
<td>55</td>
<td>36</td>
<td>2</td>
<td>43</td>
<td></td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>51</td>
<td>52</td>
<td>54</td>
<td>55</td>
<td>42</td>
<td>1</td>
<td>45</td>
<td></td>
<td></td>
</tr>
<tr>
<td>52</td>
<td>51</td>
<td>53</td>
<td>51</td>
<td>53</td>
<td>42</td>
<td>2</td>
<td>41</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2D DCT (dcts) and the quantization matrix (qmat)

dcts=

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>312</td>
<td>56</td>
<td>-27</td>
<td>17</td>
<td>79</td>
<td>-60</td>
<td>26</td>
<td>-26</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-38</td>
<td>-28</td>
<td>13</td>
<td>45</td>
<td>31</td>
<td>-1</td>
<td>-24</td>
<td>-10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-20</td>
<td>-18</td>
<td>10</td>
<td>33</td>
<td>21</td>
<td>-6</td>
<td>-16</td>
<td>-9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-11</td>
<td>-7</td>
<td>9</td>
<td>15</td>
<td>10</td>
<td>-11</td>
<td>-13</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-6</td>
<td>1</td>
<td>6</td>
<td>5</td>
<td>-4</td>
<td>-7</td>
<td>-5</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>0</td>
<td>-2</td>
<td>-7</td>
<td>-4</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>0</td>
<td>-4</td>
<td>-8</td>
<td>-1</td>
<td>2</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>-1</td>
<td>-2</td>
<td>-3</td>
<td>-1</td>
<td>4</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

qmat=

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>11</td>
<td>10</td>
<td>16</td>
<td>24</td>
<td>40</td>
<td>51</td>
<td>61</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>14</td>
<td>19</td>
<td>26</td>
<td>58</td>
<td>60</td>
<td>55</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>13</td>
<td>16</td>
<td>24</td>
<td>40</td>
<td>57</td>
<td>69</td>
<td>56</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>17</td>
<td>22</td>
<td>29</td>
<td>51</td>
<td>87</td>
<td>80</td>
<td>62</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>22</td>
<td>37</td>
<td>56</td>
<td>68</td>
<td>109</td>
<td>103</td>
<td>77</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>35</td>
<td>55</td>
<td>64</td>
<td>81</td>
<td>104</td>
<td>113</td>
<td>92</td>
<td></td>
<td></td>
</tr>
<tr>
<td>49</td>
<td>64</td>
<td>78</td>
<td>87</td>
<td>103</td>
<td>121</td>
<td>120</td>
<td>101</td>
<td></td>
<td></td>
</tr>
<tr>
<td>72</td>
<td>92</td>
<td>95</td>
<td>98</td>
<td>112</td>
<td>100</td>
<td>103</td>
<td>99</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Division by qmat \((dcthat)=dcts/qmat\)

\[
dcthat=\begin{bmatrix}
20 & 5 & -3 & 1 & 3 & -2 & 1 & 0 \\
-3 & -2 & 1 & 2 & 1 & 0 & 0 & 0 \\
-1 & -1 & 1 & 1 & 1 & 0 & 0 & 0 \\
-1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 
\end{bmatrix}
\]

\[
dcts=\begin{bmatrix}
312 & 56 & -27 & 17 & 79 & -60 & 26 & -26 \\
-38 & -28 & 13 & 45 & 31 & -1 & -24 & -10 \\
-20 & -18 & 10 & 33 & 21 & -6 & -16 & -9 \\
-11 & -7 & 9 & 15 & 10 & -11 & -13 & 1 \\
-6 & 1 & 6 & 5 & -4 & -7 & -5 & 5 \\
3 & 3 & 0 & -2 & -7 & -4 & 1 & 2 \\
3 & 5 & 0 & -4 & -8 & -1 & 2 & 4 \\
3 & 1 & -1 & -2 & -3 & -1 & 4 & 1 
\end{bmatrix}
\]

Zig-zag scan of dcthat

\[
dcthat=\begin{bmatrix}
20 & 5 & -3 & 1 & 3 & -2 & 1 & 0 \\
-3 & -2 & 1 & 2 & 1 & 0 & 0 & 0 \\
-1 & -1 & 1 & 1 & 1 & 0 & 0 & 0 \\
-1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 
\end{bmatrix}
\]

Zigzag scan as in threshold coding.
\([20, 5, -3, -1, -2, -3, 1, 1, -1, -1, 0, 0, 1, 2, 3, -2, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, EOB]\)
Threshold coding -revisited

Zig-zag scanning of the coefficients.

The coefficients along the zig-zag scan lines are mapped into [run, level] where the level is the value of non-zero coefficient, and run is the number of zero coeff. preceding it. The DC coefficients are usually coded separately from the rest.

JPEG – baseline method example

Zigzag scan as in threshold coding.

[20, 5, -3, -1, -2, -3, 1, 1, -1, 0, 0, 1, 2, 3, -2, 1, 1, 0, 0, 0, 0, 1, 0, 1, EOB].

- The DC coefficient is DPCM coded (difference between the DC coefficient of the previous block and the current block.)
- The AC coeff. are mapped to run-level pairs.
  (0,5), (0,-3), (0,-1), (0,-2),(0,-3),(0,1),(0,1),(0,-1), (2,1), (0,2), (0,3), (0,-2),(0,1),(0,1),(6,1),(0,1),(1,1),EOB.
- These are then Huffman coded (codes are specified in the JPEG scheme.)
- The decoder follows an inverse sequence of operations. The received coefficients are first multiplied by the same quantization matrix.
  (recddetshat=dctshat.*qmat.)
- Compute the inverse 2-D dct. (recdsd=idct2(recddetshat); add 128 back. (recs=recds+128.)
**Decoder**

\[
\text{Reccddcthat} = \text{dcthat} \times qmat
\]

\[
\begin{array}{cccccccccc}
320 & 55 & -30 & 16 & 72 & -80 & 51 & 0 \\
-36 & -24 & 14 & 38 & 26 & 0 & 0 & 0 \\
-14 & -13 & 16 & 24 & 40 & 0 & 0 & 0 \\
-14 & 0 & 0 & 29 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

\[
\text{Recdsd} =
\]

\[
\begin{array}{cccccccccc}
67 & 12 & -9 & 20 & 69 & 43 & -8 & 42 \\
58 & 25 & 15 & 30 & 65 & 40 & -4 & 47 \\
46 & 41 & 44 & 40 & 59 & 38 & 0 & 49 \\
41 & 52 & 59 & 43 & 57 & 42 & 3 & 42 \\
44 & 54 & 58 & 40 & 58 & 47 & 3 & 33 \\
49 & 52 & 53 & 40 & 61 & 47 & 1 & 33 \\
53 & 50 & 53 & 46 & 63 & 41 & 0 & 45 \\
55 & 50 & 56 & 53 & 64 & 34 & -1 & 57 \\
\end{array}
\]

**Received signal**

\[
\text{Received signal}
\]

\[
\text{Reconstructed S} = 
\]

\[
\begin{array}{cccccccc}
195 & 140 & 119 & 148 & 197 & 171 & 120 & 170 \\
186 & 153 & 143 & 158 & 193 & 168 & 124 & 175 \\
174 & 169 & 172 & 168 & 187 & 166 & 128 & 177 \\
169 & 180 & 187 & 171 & 185 & 170 & 131 & 170 \\
172 & 182 & 186 & 168 & 186 & 175 & 131 & 161 \\
177 & 180 & 181 & 168 & 189 & 175 & 129 & 161 \\
181 & 178 & 181 & 174 & 191 & 169 & 128 & 173 \\
183 & 178 & 184 & 181 & 192 & 162 & 127 & 185 \\
\end{array}
\]

\[
s = (8 \times 8 \text{block})
\]

\[
\begin{array}{cccccccc}
183 & 160 & 94 & 153 & 194 & 163 & 132 & 165 \\
183 & 153 & 116 & 176 & 187 & 166 & 130 & 169 \\
179 & 168 & 171 & 182 & 179 & 170 & 131 & 167 \\
177 & 177 & 179 & 177 & 179 & 165 & 131 & 167 \\
178 & 178 & 179 & 176 & 182 & 164 & 130 & 171 \\
179 & 180 & 180 & 179 & 183 & 164 & 130 & 171 \\
179 & 179 & 180 & 182 & 183 & 170 & 129 & 173 \\
180 & 179 & 181 & 179 & 181 & 170 & 130 & 169 \\
\end{array}
\]
Example

Figure 8.38: Left column: Approximations of Fig. 8.2 using the DCT and normalization array of Fig. 8.37(b). Right column: Similar results for 4L.

Wavelet Compression

Figure 8.39: A wavelet coding system: (a) encoder; (b) decoder.
FIGURE 8.41 (a), (c), and (e) Wavelet coding results with a compression ratio of 108 to 1; (b), (d), and (f) similar results for a compression of 167 to 1.

FIGURE 8.42 Wavelet transforms of Fig. 8.23 with respect to (a) Haar wavelets, (b) Daubechies wavelets, (c) symlets, and (d) Cohen-Daubechies-Feuveau biorthogonal wavelets.
Image Compression: Summary

- Data redundancy
- Self-information and Entropy
- Error-free compression
  - Huffman coding, Arithmetic coding, LZW coding, Run-length encoding
  - Predictive coding
- Lossy coding techniques
  - Predictive coding (Lossy)
  - Transform coding
    - DCT, DFT, KLT, ...
- JPEG image compression standard