Transform Coding and JPEG

Quantitative example of energy compaction and decorrelation:
Let $u$ be a $2 \times 1$ vector with mean zero:

$$u = \begin{bmatrix} u(0) \\ u(1) \end{bmatrix}$$

and covariance

$$R_u = \begin{pmatrix} E[(u(0) - \mu_0)(u(0) - \mu_0)] & E[(u(0) - \mu_0)(u(1) - \mu_1)] \\ E[(u(0) - \mu_0)(u(1) - \mu_1)] & E[(u(1) - \mu_1)(u(1) - \mu_1)] \end{pmatrix}$$

$$= \begin{pmatrix} E[u(0)^2] & E[u(0)u(1)] \\ E[u(1)u(0)] & E[u(1)^2] \end{pmatrix}$$

$$= \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$$

for $0 < \rho < 1$. From the expression for $R_u$, the variances

$$\sigma_{u(0)}^2 = \sigma_{u(1)}^2 = 1$$

that is, the total average energy of 2 is distributed equally between $u(0)$ and $u(1)$.

The parameter $\rho$ gives an indication of the correlation between $u(0)$ and $u(1)$. The correlation is by definition

$$corr[u(0), u(1)] = \frac{E[u(0)u(1)]}{\sigma_{u(0)}\sigma_{u(1)}} = \rho$$

So in this case, the off-diagonal elements of the covariance matrix are in fact exactly equal to the correlation, but that is only because the $\sigma$’s are both equal to one.

Now we transform $u$ as follows:

$$v = \frac{1}{2} \begin{pmatrix} \sqrt{3} & 1 \\ -1 & \sqrt{3} \end{pmatrix} u$$

The covariance matrix of $v$ is then

$$R_v = \begin{pmatrix} 1 + \frac{\sqrt{3}\rho}{2} & \frac{\rho}{2} \\ \frac{\rho}{2} & 1 - \frac{\sqrt{3}\rho}{2} \end{pmatrix}$$

The entries of this matrix are computed as follows:
$$v(0) = \frac{\sqrt{3}}{2} u(0) + \frac{1}{2} u(1) \quad \text{and} \quad E[v(0)^2] = \frac{3}{4} + \frac{1}{4} + 2\sqrt{3}\rho$$

eq 1.82 \quad \text{and} \quad \sigma_{v(1)}^2 = 0.18 \]

The correlation between $v(0)$ and $v(1)$ is given by

$$\rho_{v(0,1)} = \frac{E[v(0)v(1)]}{\sigma_{v(0)}\sigma_{v(1)}} = \frac{\rho}{2(1 - \frac{3}{4}\rho^2)^{1/2}}$$

(1)

which is less in absolute value than $|\rho|$ for $|\rho| < 1$. For $\rho = 0.95$, we find $\rho_{v(0,1)} = 0.83$. Hence the correlation between the transform coefficients has been reduced. If we use the transform matrix

$$A = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

then we find

$$\sigma_{v(0)}^2 = 1 + \rho \quad \text{and} \quad \sigma_{v(1)}^2 = 1 - \rho \quad \text{and} \quad \rho_{v(0,1)} = 0$$

For $\rho = 0.95$, now 97.5% of the energy is packed into $v(0)$, and the two components $v(0)$ and $v(1)$ are uncorrelated.

**Discrete Cosine Transform**

When $f(x)$ is a discrete sequence of length $N$, the unitary one dimensional discrete cosine transform is defined by

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

for $u = 0, 1, 2, \ldots, N - 1$. The inverse DCT is defined by

$$f(x) = \sum_{u=0}^{N-1} \alpha(u) C(u) \cos \left[ \frac{(2x + 1)u\pi}{2N} \right]$$
for $x = 0, 1, 2, \ldots, N - 1$. In these equations, $\alpha$ is

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

A commonly used non-unitary definition of the DCT is:

$$C(u) = \sum_{x=0}^{N-1} 2f(x) \cos \left( \frac{(2x + 1)u\pi}{2N} \right)$$

$$f(x) = \sum_{u=0}^{N-1} W(u)C(u) \cos \left( \frac{(2x + 1)u\pi}{2N} \right)$$

where

$$W(u) = \begin{cases} \frac{1}{2N} & \text{for } u = 0 \\ \frac{1}{N} & \text{for } u = 1, 2, \ldots N - 1. \end{cases}$$

The unitary 2-D DCT pair is:

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

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

In JPEG, $N=8$. The 64 DCT coefficients $C(u, v)$ indicate how much of each of the 64 basis images must be used in order to reconstruct the image block:
**JPEG Color Space:** YCrCb

\[
\begin{bmatrix}
Y \\
Cr \\
Cb
\end{bmatrix} = \begin{bmatrix}
0.299 & 0.587 & 0.114 \\
-0.169 & -0.332 & 0.5 \\
0.5 & -0.419 & -0.081
\end{bmatrix} \begin{bmatrix}
R \\
G \\
B
\end{bmatrix}
\]

Notice Cr is predominantly blue; Cb is predominantly red. Cr and Cb are called the chrominance planes. Y contains “grayscale” values only; it is called the “luminance” plane. Because Cr and Cb contain less information than Y (and are less perceptually important), they are downsampled by a factor of 2 in each direction. They are then interpolated for display. Before downsampling, the images are filtered to reduce aliasing:

![Diagram of filtering and downsampling](image)

**Example (from Gonzalez and Woods) of how JPEG works:**

Consider the 8 × 8 subimage:

\[
\begin{array}{cccccccc}
52 & 55 & 61 & 66 & 70 & 61 & 64 & 73 \\
63 & 59 & 66 & 90 & 109 & 85 & 69 & 72 \\
62 & 59 & 68 & 113 & 144 & 104 & 66 & 73 \\
63 & 58 & 71 & 122 & 144 & 106 & 70 & 69 \\
67 & 61 & 68 & 104 & 126 & 88 & 68 & 70 \\
79 & 65 & 60 & 70 & 77 & 68 & 58 & 75 \\
85 & 71 & 64 & 59 & 55 & 61 & 65 & 83 \\
87 & 79 & 69 & 68 & 65 & 76 & 78 & 94
\end{array}
\]

First we shift the pixel values by \(-2^7 = -128\) gray levels (below left), then transform by forward DCT (below right):

\[
\begin{array}{cccccccc}
-49 & -63 & -68 & -58 & -51 & -60 & -70 & -53 \\
-41 & -49 & -59 & -60 & -63 & -52 & -50 & -34
\end{array}
\]

\[
\begin{array}{cccccccc}
-415 & -29 & -62 & 25 & 55 & -20 & -1 & 3 \\
7 & -21 & -62 & 9 & 11 & -7 & -6 & 6 \\
-46 & 8 & 77 & -25 & -30 & 10 & 7 & -5 \\
-50 & 13 & 35 & -15 & -9 & 6 & 0 & 3 \\
11 & -8 & -13 & -2 & -1 & 1 & -4 & 1 \\
-10 & 1 & 3 & -3 & -1 & 0 & 2 & -1 \\
-4 & -1 & 2 & -1 & 2 & -3 & 1 & -2 \\
-1 & -1 & -1 & -2 & -1 & -1 & 0 & -1
\end{array}
\]

Each pixel will be quantized by a uniform quantizer whose step size is given by the appropriate position in the quantization table (Q table). The matrix of step sizes \(\Delta\) tells you that each coefficient is supposed
to be subjected to a uniform quantizer as shown below.

### Luminance quantization table:

<table>
<thead>
<tr>
<th>16</th>
<th>11</th>
<th>10</th>
<th>16</th>
<th>24</th>
<th>40</th>
<th>51</th>
<th>61</th>
</tr>
</thead>
<tbody>
<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>
</tr>
<tr>
<td>14</td>
<td>13</td>
<td>24</td>
<td>40</td>
<td>57</td>
<td>69</td>
<td>56</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>22</td>
<td>29</td>
<td>51</td>
<td>87</td>
<td>80</td>
<td>62</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>
</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>
</tr>
<tr>
<td>49</td>
<td>64</td>
<td>78</td>
<td>87</td>
<td>103</td>
<td>121</td>
<td>100</td>
<td>103</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>
</tr>
</tbody>
</table>

### Chrominance quantization table:

<table>
<thead>
<tr>
<th>17</th>
<th>18</th>
<th>24</th>
<th>47</th>
<th>99</th>
<th>99</th>
<th>99</th>
<th>99</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>21</td>
<td>26</td>
<td>66</td>
<td>99</td>
<td>99</td>
<td>99</td>
<td>99</td>
</tr>
<tr>
<td>24</td>
<td>26</td>
<td>56</td>
<td>99</td>
<td>99</td>
<td>99</td>
<td>99</td>
<td>99</td>
</tr>
<tr>
<td>47</td>
<td>66</td>
<td>99</td>
<td>99</td>
<td>99</td>
<td>99</td>
<td>99</td>
<td>99</td>
</tr>
</tbody>
</table>

The index for the coefficient is

\[ i = \text{round}\left( \frac{x}{\Delta} \right) \]

When the decoder decodes the index, it will reconstruct the coefficient value as

\[ y = \Delta \times i = \Delta \times \text{round}\left( \frac{x}{\Delta} \right) \]

**Remarks on the Q tables:**

- Tables determined by measuring, experimentally, the sensitivity of human observers to sinusoids of different frequencies in different directions

- The luminance table is not symmetric, as you’d think it would be. It is also not monotonic in either the horizontal or vertical directions. This is because the contrast sensitivity function for human beings is not monotonic— it has a bandpass characteristic. For frequencies lower and higher than the peak, the visual sensitivity is less. JPEG makes assumptions about the viewing distances and pixel sizes, and thereby makes assumptions about which AC coefficients will correspond to the peak of the human contrast sensitivity function.

- Chrominance step size increases more rapidly with frequency than luminance step size. Human beings have little difficulty in resolving shapes when the color info is bleeding slightly across the grayscale edges.
In our example, the DC coefficient becomes
\[
\text{round} \left[ \frac{-415}{16} \right] = -26
\]
-26 -3 -6 2 2 0 0 0
The entire group of coefficients comes out as:
1 -2 -4 0 0 0 0 0
-3 1 5 -1 -1 0 0 0
-4 1 2 -1 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
0 0 0 0 0 0 0 0

How do we choose quantization matrices?
1. Use the recommended matrix. But then we get only a single compression ratio for each image—whatever it turns out to be. Solution: scaling. Allow the user to select a target quality level or target bit rate, and just scale the recommended matrix. Multiply or divide it by some value to achieve the desired quantization.
2. Design your own. Visual detection model predicts sensitivity as a function of display parameters and image content, allowing Q matrices to be designed for specific conditions:
   - conditions related to spatial frequency (pixel size, viewing distance, aspect ratio)
   - ambient lighting
   - image content (background luminance, spatial masking)
   - different color spaces
   - Optimize for whatever processing operation or display comes next: e.g., halftoning and printing

After quantizing: Reorder by zigzag scan:

Results of zigzag ordering:
DC: -26 - previous value
AC: (0,-3),(0,1),(0,-3),(0,-2),(0,-6),(0,2),(0,-4),(0,1), (0,-4),(0,1),(0,1),(0,5),(1,2),(2,-1), (0,2),(5,-1),(0,-1) EOB
where the EOB denotes the end-of-block condition (meaning that all coefficients after that point are zero).

Compute difference between current DC coefficient and one from previous subblock:
\[-26 - (-17) = -9\]
which lies in DC difference category 4. The base code for -9 is 101.

<table>
<thead>
<tr>
<th>Range</th>
<th>DC Cat.</th>
<th>AC Cat.</th>
<th>Category</th>
<th>Base Code</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>N/A</td>
<td>0</td>
<td>010</td>
<td>3</td>
</tr>
<tr>
<td>-1,1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>011</td>
<td>4</td>
</tr>
<tr>
<td>-3, -2, 2, 3</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>100</td>
<td>5</td>
</tr>
<tr>
<td>-7, ..., -4, 4, ..., 7</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>00</td>
<td>5</td>
</tr>
<tr>
<td>-15, ..., -8, 8, ..., 15</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>101</td>
<td>7</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>110</td>
<td>8</td>
</tr>
<tr>
<td>-32767, ..., -16384, 16384, ..., 32767</td>
<td>F</td>
<td>N/A</td>
<td>F</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

and the total length is 7. If we consider the possible elements of category 4, we see there are 16 elements, which can be represented by 4 bits.

<table>
<thead>
<tr>
<th>Elts</th>
<th>-15</th>
<th>-14</th>
<th>-13</th>
<th>-12</th>
<th>-11</th>
<th>-10</th>
<th>-9</th>
<th>-8</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code</td>
<td>0000</td>
<td>0001</td>
<td>0010</td>
<td>0011</td>
<td>0100</td>
<td>0101</td>
<td>0110</td>
<td>0111</td>
<td>1000</td>
<td>1001</td>
<td>1010</td>
<td>1011</td>
<td>1100</td>
<td>1101</td>
<td>1110</td>
<td>1111</td>
</tr>
</tbody>
</table>

-9 corresponds to 0110. So the total codeword is 1010110.

AC coefficients depend on the number of zero coefficients preceding the nonzero coefficient to be coded.

<table>
<thead>
<tr>
<th>Run / Category</th>
<th>Base Code</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>0/0</td>
<td>1010 (=EOB)</td>
<td>4</td>
</tr>
<tr>
<td>0/1</td>
<td>00</td>
<td>3</td>
</tr>
<tr>
<td>0/2</td>
<td>01</td>
<td>4</td>
</tr>
<tr>
<td>0/3</td>
<td>100</td>
<td>6</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

So the first nonzero AC coefficient is -3, coded as 0100. The completely coded sequence is

1010110 0100 001 0100 0101 100001 0110 100011 001 100011 001 001 100101 11100110 110110 0110 11110100 000 1010

92 bits in the sequence. Originally there were \(8 \times 8 \times 8 = 512\) bits, so the compression is 512:92 (about 5.6:1).

This Huffman coded binary sequence is *instantaneous* and *uniquely decodable*. Decoded by table lookup.

Note that the Huffman Tables, like the quantization tables, can be specified by the user.
- tailor for low bit rates
- tailor for individual images (2-pass technique)

The Huffman decoded values are “dequantized” in accordance with earlier table. For example, the DC coefficient is $-26 \times 16 = -416$. The resulting array is shown below left. Lastly we take an inverse DCT and shift by +128 (below right). The errors range from −14 to +11.

```
-416 -33 -60 32 48 0 0 0
12 -24 -56 0 0 0 0 0
-42 13 80 -24 -40 0 0 0
-56 17 44 -29 0 0 0 0
18 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
```

```
58 64 67 64 59 62 70 78
56 55 67 89 98 88 74 69
60 50 70 119 141 116 80 64
69 51 71 128 149 115 77 68
74 53 64 105 115 84 65 72
76 57 56 74 75 57 57 74
83 69 59 60 61 61 67 78
93 81 67 62 69 80 84 84
```

Overall JPEG Diagram:

```
8 X 8
DCT
Quantize

DC
AC

DPCM
zig-zag
scan
run-length
encode

Entropy Code
Add header
bits

IMAGE
```

The header contains various pieces of information such as:
- The number of rows and number of columns, the sample precision (8 bits, 12 bits)
- The mode (sequential, progressive, etc. see below)
- The number of components (3 color planes, etc)
- The interleaving strategy for the components
- Whether the default Q,H table is being used, and if not, what the Q,H table is.
**JPEG ENHANCEMENTS:**
There are a few things one can do to improve the quality of the output picture, while still remaining within the standard.

1. **Removing blocking artifacts with AC prediction:** Suppose the image is coded at a very low rate such that very few of even the lowest AC coefficients are getting coded. In that case, after decoding the entire image, the decoder can consider each block in turn and attempt to predict some of the AC coefficients that were not sent. For example, for a given block, the decoder can look at the DC values in that block and in the 8 neighboring blocks. The decoder can fit a quadratic surface to the array of 9 values:

\[
P(x, y) = A_1 x^2 y^2 + A_2 x^2 y + A_3 x y^2 + A_4 x^2 + A_5 x y + A_6 y^2 + A_7 x + A_8 y + A_9
\]

The \( A_i \) are computed by requiring that the mean values computed for the quadratic surface match the received DC values. As a decoder option, this yields about 2% improvement in quality at low rates.

2. **Approximate Adaptive Quantization:** Also called “lying to the decoder,” this method consists of having the encoder use a priori information to decide that certain portions of the image are not important, and it can zero out coefficients in those portions. The bit rate will thereby go down, and the decoder will receive a valid bit stream and will not know that it has been lied to, and that its reconstruction quality has been selectively adjusted.

More enhancements are possible if the bit stream is not trying to be JPEG compliant.

**JPEG LOSSLESS MODE:**
Pure lossless encoding uses 2-D prediction on the images. No DCT is used. JPEG has 8 options for the prediction function \( f \). The difference \( d \) is losslessly encoded:

\[
d = x - f(a, b, c)
\]

<table>
<thead>
<tr>
<th>Option</th>
<th>Prediction function ( f(a, b, c) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>no prediction (no differential coding)</td>
</tr>
<tr>
<td>1</td>
<td>a</td>
</tr>
<tr>
<td>2</td>
<td>b</td>
</tr>
<tr>
<td>3</td>
<td>c</td>
</tr>
<tr>
<td>4</td>
<td>a+b−c</td>
</tr>
<tr>
<td>5</td>
<td>a+(b−c)/2</td>
</tr>
<tr>
<td>6</td>
<td>b+(a−c)/2</td>
</tr>
<tr>
<td>7</td>
<td>(a+b)/2</td>
</tr>
</tbody>
</table>

- Whenever the prediction scheme involves negative values, we are extrapolating that the new pixel will follow the same trend as the previous row or column.
- Prediction at the array edges just uses value a or b.
- Pseudo-lossless encoding first reduces the input precision by \( \geq 1 \) bit before losslessly encoding as above.
**JPEG Hierarchical Mode:**

```
[Diagram depicting JPEG Hierarchical Mode]
```

**JPEG Progressive mode:**

*Spectral selection*—defines bands of DCT coefficients. Example: 4 bands

- DC coefficient
- AC coefs 1–5
- AC coefs 6–27
- AC coefs 28–63

Band 1 is coded first, for the whole image. Then Band 2 is coded, for the whole image, etc. What price do you pay for doing this? More bits. You are breaking the run-lengths.