Transform Coding and JPEG

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

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

and covariance

$$\begin{aligned} R_u &= \begin{pmatrix} E[(u(0)-\mu_0)(u(0)-\mu_0)] & E[(u(0)-\mu_0)(u(1)-\mu_1)] \\ E[(u(0)-\mu_0)(u(1)-\mu_1)] & E[(u(1)-\mu_1)(u(1)-\mu_1)] \end{pmatrix} \\ &= \begin{pmatrix} E[u(0)^2] & E[u(0)u(1)] \\ E[u(1)u(0)] & E[u(1)^2] \end{pmatrix} \\ &= \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix} \end{aligned}$$

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

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

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

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

The entries of this matrix are computed as follows:

$$v(0) = \frac{\sqrt{3}}{2}u(0) + \frac{1}{2}u(1) \qquad \text{and} \qquad E[v(0)^2] = \frac{3}{4} + \frac{1}{4} + 2\frac{\sqrt{3}}{4}\rho$$

etc. Now if we compare the energy of the two coordinates:

$$\sigma^2_{v(0)} = 1 + \sqrt{3}\rho/2 \qquad \text{and} \qquad \sigma^2_{v(1)} = 1 - \sqrt{3}\rho/2$$

we see that the total average energy is still 2, but now the energy in $v(0)$ is greater than in $v(1)$. If $\rho = 0.95$, then 91.1% of the total average energy has been packed in the first sample.

$$\sigma^2_{v(0)} = 1.82 \qquad \sigma^2_{v(1)} = 0.18$$

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

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

which is less in absolute value than $\mid \rho \mid$ for $\mid \rho \mid < 1$ 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

$$\sigma^2_{v(0)} = 1 + \rho \qquad \text{and} \qquad \sigma^2_{v(1)} = 1 - \rho \qquad \text{and} \qquad \rho_{v(0,1)} = 0$$

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, α is

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

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

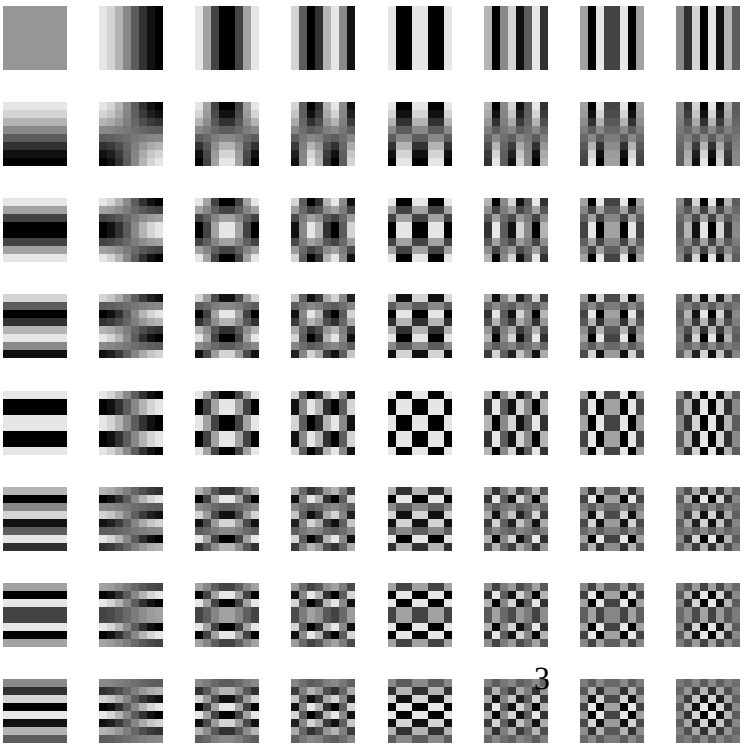
$$W(u) = \begin{cases} \frac{1}{2N} & \text{for } u = 0 \\ \frac{1}{N} & \text{for } u = 1, 2, \ldots N-1. \end{cases}$$

The unitary 2-D DCT pair is:

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

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

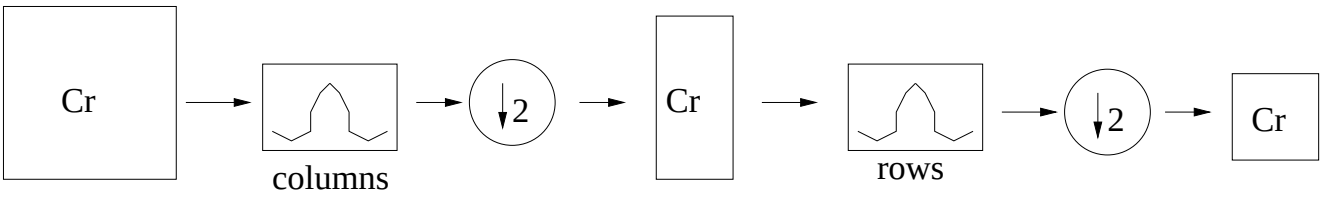
In JPEG, N=8. The 64 DCT coefficients $C(u,v)$ indicate how much of each of the 64 basis images must be used in order to reconstruct the image block:



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:



Example (from Gonzalez and Woods) of how JPEG works:
Consider the 8×8 subimage:

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	154	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

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

-76	-73	-67	-62	-58	-67	-64	-55
-65	-69	-62	-38	-19	-43	-59	-56
-66	-69	-60	-15	16	-24	-62	-55
-65	-70	-57	-6	26	-22	-58	-59
-61	-67	-60	-24	-2	-40	-60	-58
-49	-63	-68	-58	-51	-60	-70	-53
-43	-57	-64	-69	-73	-67	-63	-45
-41	-49	-59	-60	-63	-52	-50	-34

-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

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 Δ tells you that each coefficient is supposed

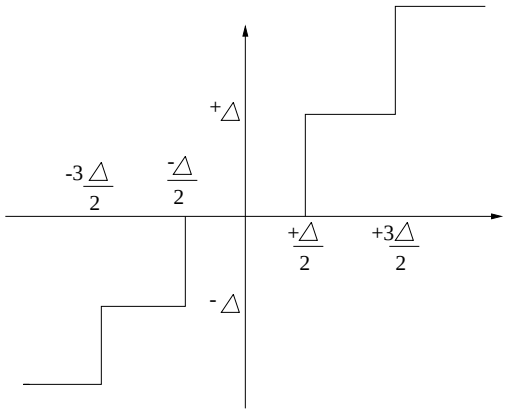
to be subjected to a uniform quantizer as shown below.

Luminance quantization table:

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Chrominance quantization table:

17	18	24	47	99	99	99	99
18	21	26	66	99	99	99	99
24	26	56	99	99	99	99	99
47	66	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99



The index for the coefficient is

$$i = round(\frac{x}{\Delta})$$

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

$$y = \Delta \times i = \Delta \times round(\frac{x}{\Delta})$$

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

$$round\left[\frac{-415}{16}\right] = -26$$

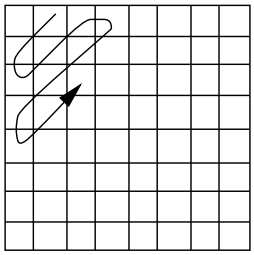
The entire group of coefficients comes out as:

-26	-3	-6	2	2	0	0	0
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

Range	DC Cat.	AC Cat.
0	0	N/A
-1,1	1	1
-3,-2,2,3	2	2
$-7,\ldots,-4,4,\ldots,7$	3	3
$-15,\ldots,-8,8,\ldots,15$	4	4
$\vdots$	$\vdots$	$\vdots$
-32767,...,-16384,16384,...,32767	F	N/A

Category	Base Code	Length
0	010	3
1	011	4
2	100	5
3	00	5
4	101	7
5	110	8
6	1110	10
7	11110	12
8	111110	14
9	1111110	16
A	11111110	18
B	111111110	20

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.

Elts	-15	-14	-13	-12	-11	-10	-9	-8	8	9	10	11	12	13	14	15
Code	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

-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.

Run / Category	Base Code	Length
0/0	1010 (=EOB)	4
0/1	00	3
0/2	01	4
0/3	100	6
$\vdots$	$\vdots$	$\vdots$

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.

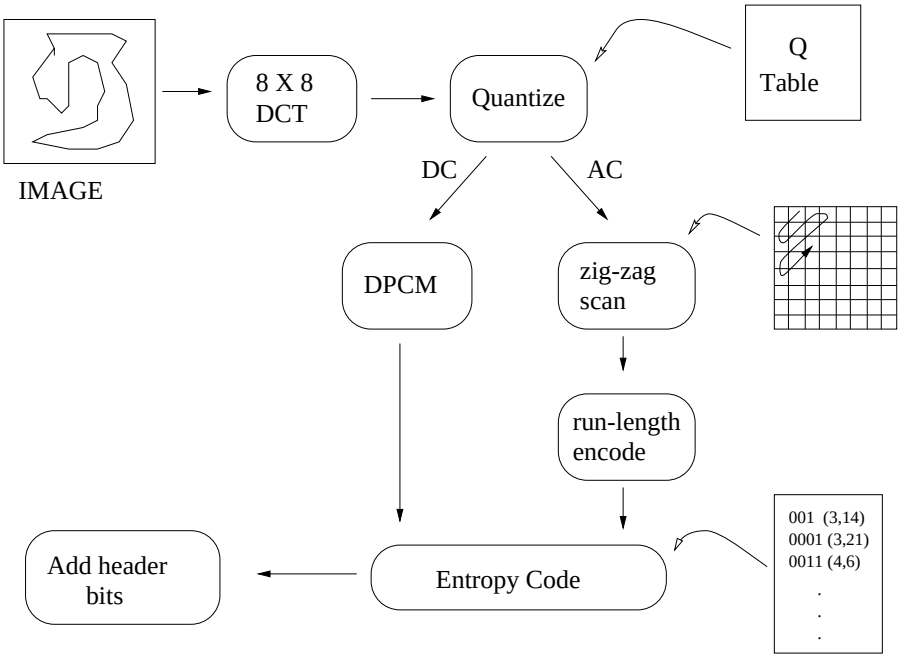
- tailor for low bit rates
- tailor for individual images (2-pass technique)

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:



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:

   $$P(x,y) = A_1x^2y^2 + A_2x^2y + A_3xy^2 + A_4x^2 + A_5xy + A_6y^2 + A_7x + A_8y + A_9$$

   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:

c	b
a	x

$$d = x - f(a,b,c)$$

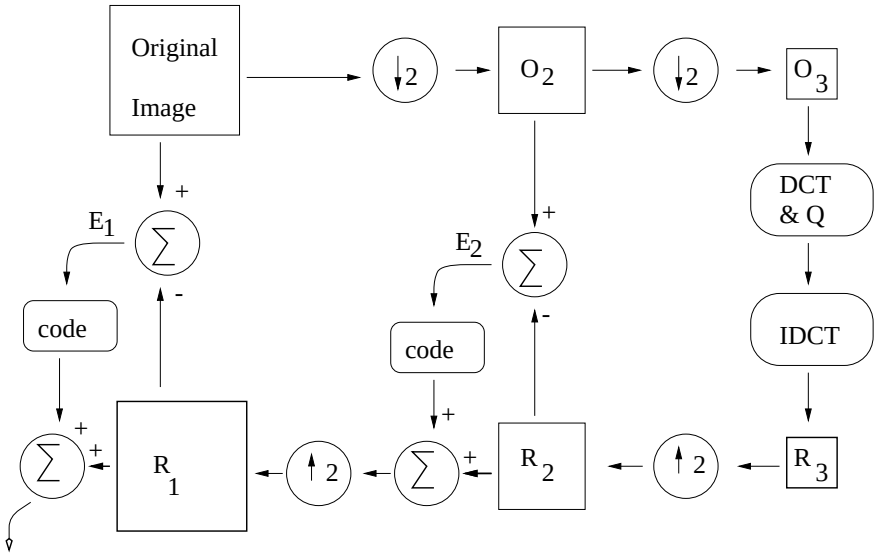
Option	Prediction function $f(a,b,c)$
0	no prediction (no differential coding)
1	a
2	b
3	c
4	a+b−c
5	a+(b−c)/2
6	b+(a−c)/2
7	(a+b)/2

• 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 ≥ 1 bit before losslessly encoding as above.

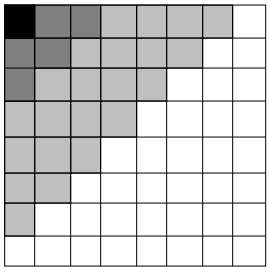
JPEG Hierarchical Mode:



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.