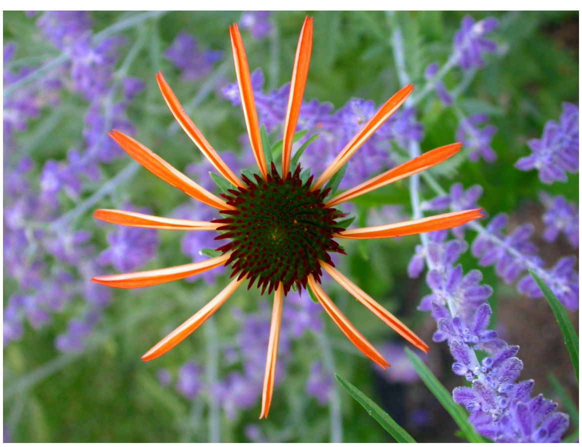
WHAT IS... JPEG?

David Austin

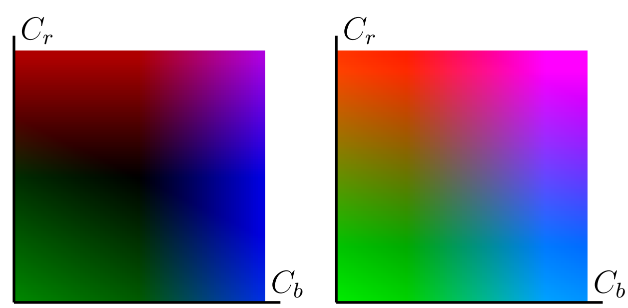
The image below consists of a rectangular array of 3,871,488 pixels. The color of each pixel is determined by red, green, and blue components, each requiring one byte of computer memory. Naively, we expect the image to require 11,614,464 bytes. However, the size of the JPEG file containing this image is only 734,268 bytes, roughly 16 times smaller. We will describe the compression algorithm, developed by the Joint Photographic Experts Group (JPEG), that allows the image to be stored so compactly.



Rather than specifying the red, green, and blue components of a color, it is convenient to use three different quantities: luminance Y, which is closely related to the brightness of the color, and blue and red chrominances C_b and C_r, which roughly determine the hue. An invertible affine transform translates between the two representations; for instance, to recover the red R, green G, and blue B components, we use

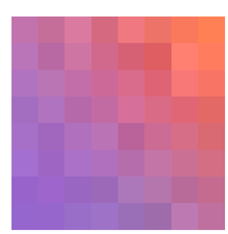
$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.0 & 0.0 & 1.40210 \\ 1.0 & -0.34414 & -0.71414 \\ 1.0 & 1.77180 & 0.0 \end{bmatrix} \times \begin{bmatrix} Y \\ C_b - 128 \\ C_r - 128 \end{bmatrix}.$$

Notice that the luminance contributes equally to the three color components. To help visualize this transformation, we show the colors that result from fixing the luminance and mixing various chrominance values.



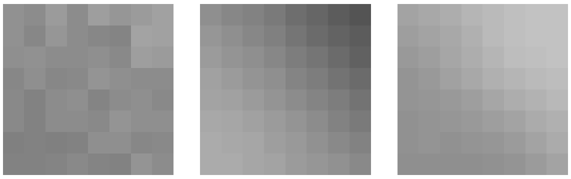
$Y = 0$ $\qquad$ $Y = 100$

The algorithm proceeds by dividing the image into 8 by 8 blocks of pixels that are independently processed. Here is a sample block.



The (Y, C_b, C_r) components in our sample 8 by 8 block are shown below; lighter regions correspond to larger values.

David Austin is professor of mathematics at Grand Valley State University. His email address is `david@merganser.math.gvsu.edu`.



Y C_b C_r

Notice that the luminance values produce a grayscale version of the image. As pyscho-visual experiments show that the human eye is most sensitive to the luminance values, the color transform concentrates the most important information into a single component. Color television uses a similar color model, enabling black and white televisions to display efficiently color images that are broadcast.

For reasons that will be explained later, we now express the component values as a linear combination of cosine functions of increasing frequency. For instance, if $Y_{x,y}$ is the luminance in column x and row y of our block, we write

$$Y_{x,y} = \sum_{u=0}^{7}\sum_{v=0}^{7} C_{u,v}F_{u,v}\cos\left[\frac{(2x+1)u\pi}{16}\right] \times \cos\left[\frac{(2y+1)v\pi}{16}\right].$$

(The normalizing constants $C_{u,v}$ need not concern us.) The coefficients $F_{u,v}$ are found by the two-dimensional Discrete Cosine Transform (DCT)

$$F_{u,v} = C_{u,v}\sum_{x=0}^{7}\sum_{y=0}^{7} Y_{x,y}\cos\left[\frac{(2x+1)u\pi}{16}\right] \times \cos\left[\frac{(2y+1)v\pi}{16}\right]$$

and efficiently computed using a version of the Fast Fourier Transform.

The values of the components in most blocks do not change rapidly, and the human eye is not particularly sensitive to these changes. Therefore, the DCT coefficients corresponding to higher frequencies will likely be small and may be ignored without affecting our perception of the image. This observation motivates our method for quantizing the DCT coefficients so they may be stored as integers.

Two ingredients are used in the quantization process. The first is a parameter α, chosen by the user to control the amount of compression and the quality of the image. Larger values of α lead to smaller files and lower quality images.

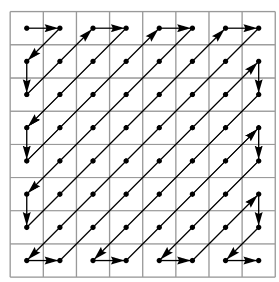
The second ingredient is an 8 by 8 matrix $Q = [Q_{u,v}]$ with the coefficients quantized by rounding $F_{u,v}/\alpha Q_{u,v}$. The $Q_{u,v}$ are chosen empirically and typically have larger values for higher frequencies so as to deemphasize those frequencies. For instance, a typical matrix used for quantizing the luminance DCT coefficients is

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

Recognizing that the luminance carries more important visual information, we use different matrices for quantizing the coefficients describing luminance and chrominance. Processing our sample block with an intermediate value of α, the quantized luminance coefficients for our sample block are shown below.

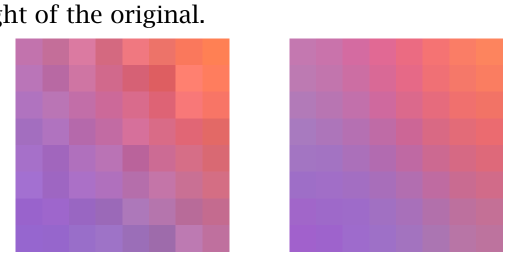
$$\begin{array}{c|cccccccc} & & & & u \rightarrow & & & & \\ \hline v\downarrow & 7 & -2 & 1 & 0 & 0 & 0 & 0 & 0 \\ & 4 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

The quantized coefficients are ordered by following the arrows so that lower frequencies appear first.

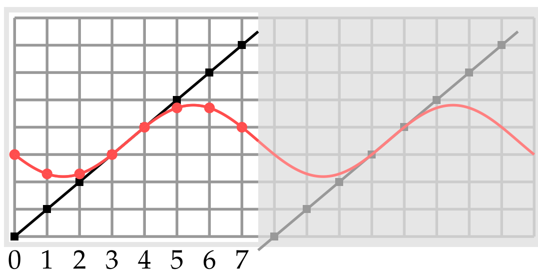


The quantized coefficients for the luminance component in our sample then produce the sequence 7, −2, 4, 1, 0, 1, 0, 1, −1 followed by 55 zeroes. Rather than storing each zero, we simply record the number of zeroes, which reduces the storage requirement significantly. Further compression is obtained by using a Huffman code to store the sequence of coefficients compactly.

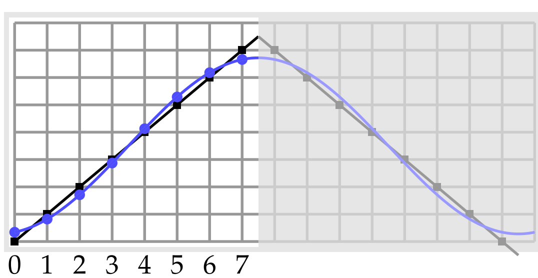
The image is reconstructed by reversing this process. The quantized coefficients give approximate values of $F_{u,v}$. These, in turn, give the (Y, C_b, C_r) components and the (R, G, B) components. The reconstructed block is shown to the right of the original.



The Discrete Fourier transform (DFT) may appear preferable to the DCT since it is more easily computed; however, our choice of the DCT is explained by our desire to concentrate the information in the low-frequency coefficients. For instance, consider the values, y_x, of one component in one row of an 8 by 8 block. The DFT expresses y_x as a linear combination of functions whose period is 8 resulting in a periodic extension of y_x. The transform therefore unnecessarily tracks the change between y_7 and $y_8 = y_0$, which can lead to significant high-frequency contributions. In the figure below, the y_x values are shown in black while the approximations given by the three lowest-frequency terms of the Fourier transform are in red.



By comparison, the DCT expresses y_x as a linear combination of functions whose period is 16 and that are symmetric about $x = 7.5$. This smoothes out the appropriate extension of y_x so that the DCT requires a relatively small contribution from high-frequency terms. Below we see the analogous approximation given by the DCT and note the improvement in the approximation. (These figures appear in [2].)

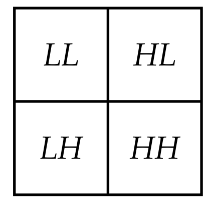


Since the 8 by 8 blocks are processed independently, discontinuities at the edges of the blocks become apparent at high compression ratios. In addition, it is often desirable to have the ability to reconstruct efficiently the image at intermediate resolutions. These reasons and others led to the creation of the JPEG 2000 compression algorithm. Among other differences, JPEG 2000 replaces the DCT with a discrete wavelet transform.

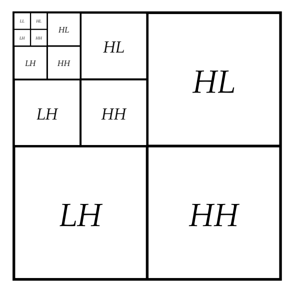
The JPEG 2000 algorithm divides the image into larger blocks, perhaps of size 256 by 256. To illustrate the wavelet transform, fix one row of pixels in a block and let y_x represent the values of one of the components. Now form wavelet coefficients

$$\begin{aligned} l_x &= (y_{2x} + y_{2x+1})/2 \\ h_x &= (y_{2x} - y_{2x+1})/2. \end{aligned}$$

The h_x are called "high-pass" coefficients, since they detect high-frequency changes, while the l_x are "low-pass" coefficients. Order them by listing all the low-pass coefficients followed by the high-pass coefficients and perform the same operation on the columns of wavelet coefficients to obtain blocks of coefficients:



The coefficients in the LL sub-block are obtained by averaging over 2 by 2 blocks of pixels and hence represent a lower resolution version of the image. The other three sub-blocks describe the changes necessary to construct the image at the higher resolution. We iterate this process on the LL sub-block thus storing the image at increasingly lower resolutions.



A quantization procedure detects regions where the values do not change significantly so that high-pass coefficients may be safely ignored.

Rather than the wavelet transform described above, which averages two adjacent values, the JPEG 2000 algorithm uses the Cohen-Daubechies-Feauveau (9, 7) wavelet transform, which finds a weighted average over nine adjacent values and thus produces smoother images.

The complexity of the JPEG 2000 algorithm, compared to that of the original JPEG algorithm, is an order of magnitude greater, and at low and medium compression ratios, the quality of the images produced by JPEG 2000 is not substantially better. At very high compression ratios, however, where JPEG's use of 8 by 8 blocks can cause the image quality to deteriorate severely, JPEG 2000 offers significantly better results.

Since JPEG 2000 asks us to work harder to produce images of generally comparable quality, it is not a clearly superior choice over JPEG. Indeed, only a few web browsers are currently capable of displaying JPEG 2000 images. Its principal advantage lies in providing a much more flexible format for working with images in environments where the increased complexity is not problematic.

For instance, the ability to reconstruct efficiently the image at different resolutions, which results from the use of the wavelet transform, allows users to search visually through many images at a low resolution quickly. JPEG 2000 also permits regions, designated by a user perhaps, to be displayed at a higher resolution, a reason that it is commonly used in medical imagery. Finally, it is possible for digital photographs, stored on a camera's memory card in the JPEG 2000 format, to be converted efficiently to a lower resolution to reduce memory usage after they are taken. JPEG 2000, being designed roughly ten years after JPEG, also includes other features, such as the ability to encrypt images, whose need was not anticipated earlier.

> David Austin also wrote about JPEG in "Image Compression: Seeing What's Not There", the September 2007 installment of the *Feature Column* on the AMS website (see `http://www.ams.org/featurecolumn/archive/image-compression.html`). The *Feature Column*, which appears monthly, presents accessible essays about mathematics that are designed for those who have already discovered the joys of mathematics as well as for those who are just getting to know its charms. The *Feature Column* is freely available at `http://www.ams.org/featurecolumn`.

Further Reading

[1] T. Archarya and P.-S. Tsai, *JPEG2000 Standard for Image Compression: Concepts, Algorithms and VLSI Architectures*, Wiley, Hoboken, 2005.

[2] J. Blinn, What's the Deal with the DCT?, *IEEE Computer Graphics and Applications* **13** (4) (1993), 78–83.

[3] Joint Photographic Experts Group, `http://www.jpeg.org/`.