6.3000: Signal Processing

DCT and JPEG

- Block Processing
- Discrete Cosine Transform (DCT)
- JPEG

November 25, 2025

Data Compression

Fourier methods underlie many common data compression schemes, including **JPEG** (for pictures), **MP3** (for audio) and **MPEG** (for video).

Example: **JPEG** (Joint Photographic Experts Group) Encoding

1. color encoding: RGB $\rightarrow$ YCrCb
2. 2D DCT (discrete cosine transform): a kind of Fourier series
3. quantization to achieve perceptual compression (lossy)
4. run-length and Huffman encoding (lossless)

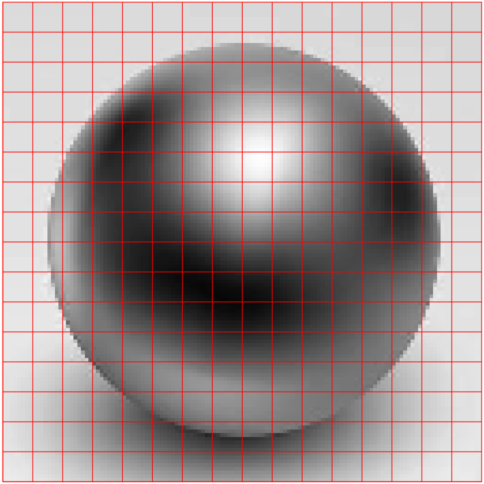
We will focus on steps 2 & 3: the DCT and quantization of its components.

- the image is broken into 8×8 pixel **blocks**
- each block is represented by its 8×8 **DCT** coefficients
- each DCT coefficient is quantized, using higher resolutions for coefficients with greater **perceptual** importance

Block Processing

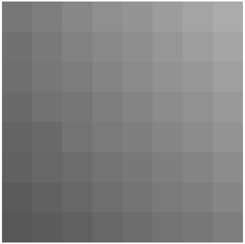
The idea behind block processing is that small patches of an image can be coded efficiently (with just a few bits).

Break the image into blocks.



Compaction

The block has $8 \times 8 = 64$ pixels.

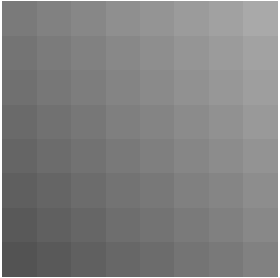


Representing each pixel in a block with an 8-bit number
$\rightarrow$ a total of 64 bytes for this block.

Compaction

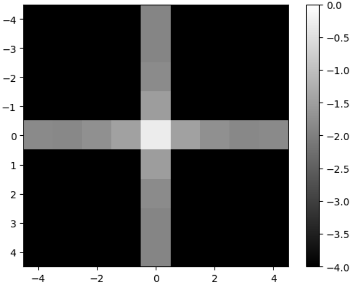
Try coding the 2D DFT instead. Here is the magnitude of the 2D DFT.

$f'[r, c]$



$\xRightarrow{\text{DFT}}$

$\log_{10}|F'[k_r, k_c]|$



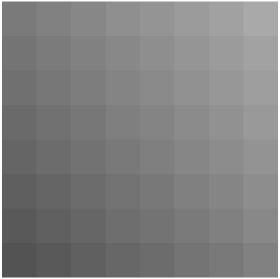
This looks promising. Only the horizontal and vertical frequencies have magnitudes greater than $F[0, 0]/100$.

Retaining just those frequencies introduces little error in the reconstructed signal $f'[r, c]$.

Check Yourself

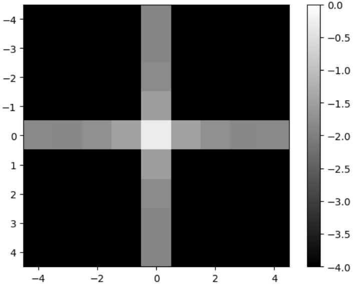
The DFT coefficients are **complex numbers**.

$f'[r,c]$



$\overset{\text{DFT}}{\Longrightarrow}$

$\log_{10}|F'[k_r,k_c]|$



How many real-valued numbers are needed to represent the information contained in the non-black pixels in the right image.

1. 15
2. 16
3. 17
4. 34
5. none of the above

Compaction

We can do even better with a different but related transform.

Discrete Cosine Transform (DCT)

$$f[n] \overset{\text{DCT}}{\Longrightarrow} F_C[k]$$

$$F_C[k] = \frac{1}{N}\sum_{n=0}^{N-1} f[n]\cos\left(\frac{\pi k}{N}\left(n+\frac{1}{2}\right)\right) \qquad \text{(analysis)}$$

$$f[n] = F_C[0] + 2\sum_{k=1}^{N-1} F_C[k]\cos\left(\frac{\pi k}{N}\left(n+\frac{1}{2}\right)\right) \qquad \text{(synthesis)}$$

The DCT ...

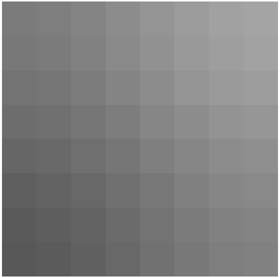
- has cosine terms but no sine terms.
- frequencies at half multiples of the fundamental $2\pi/N$ are analyzed.
- times between integer values of n are analyzed.

The DCT of a real-valued signal is **real-valued**.

Compaction

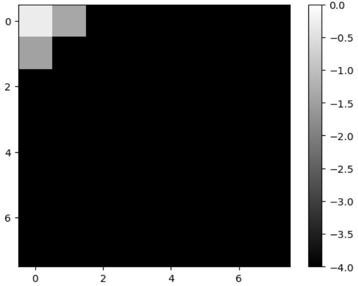
Try coding the 2D DCT instead. Here is the magnitude of the 2D DCT.

$f'[r, c]$



$\overset{\text{DCT}}{\Longrightarrow}$

$\log_{10}|F'_C[k_r, k_c]|$



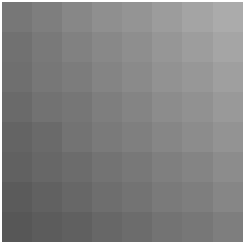
This looks even more promising. Now there are only 3 discrete frequencies with magnitudes greater than $F[0, 0]/100$.

The information in $F'_C[k_r, k_c]$ can be represented by just 3 real numbers: five times fewer bytes than that for the DFT.

What is the DCT and why is its representation more compressed than that of the DFT?

Compaction

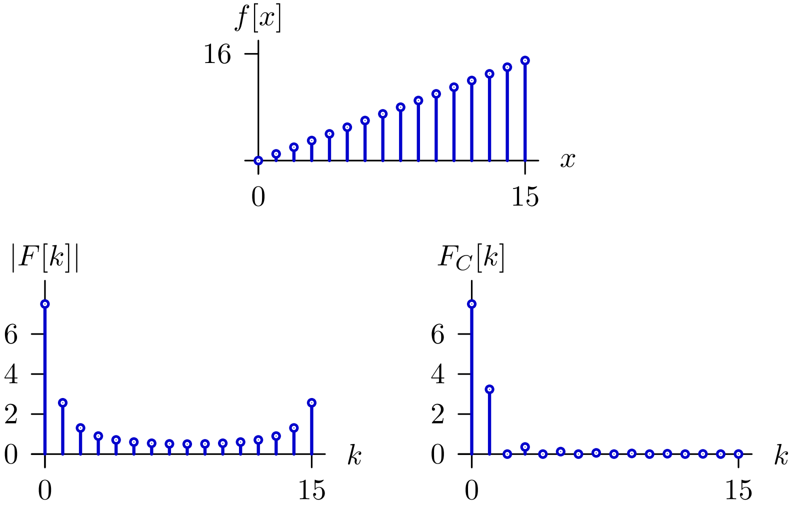
Consider the structure of the patch that we have been examining.



It's basically a 2D ramp: brighter in the upper right than in the lower left. Such blocks are common, and not so easy to compress with the DFT.

Compaction of a Ramp

Compare the DFT and DCT of a "ramp."



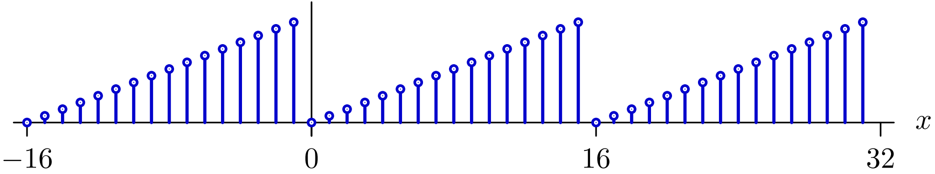
Why are there so many high frequencies in the DFT?

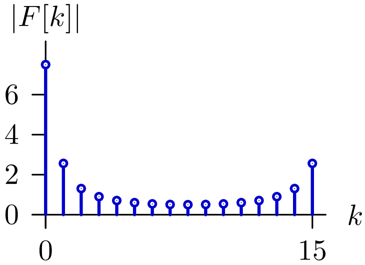
And why are there fewer in the DCT?

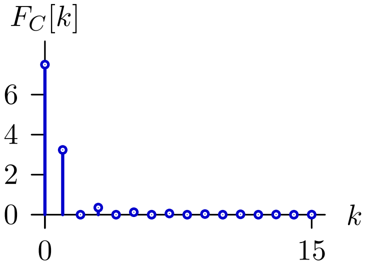
DFT of a Ramp

The DFT is the Fourier series of a periodically extended version of a signal.

$$f_p[x] = \sum_{m=-\infty}^{\infty} f[x+mN]$$





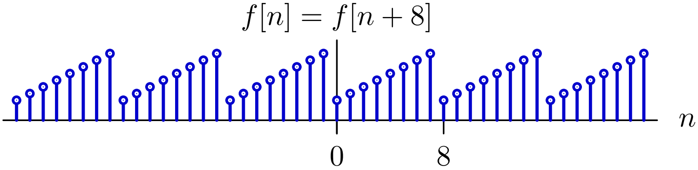


Periodic extension of a ramp results in a sawtooth wave.

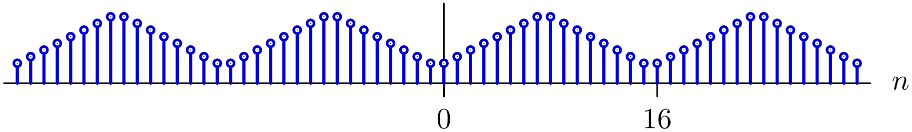
Step discontinuities at the window edges produce high-frequency content.

Discrete Cosine Transform (DCT)

The idea in the Discrete Cosine Transform (DCT) is to avoid introducing step discontinuities in periodic extension:



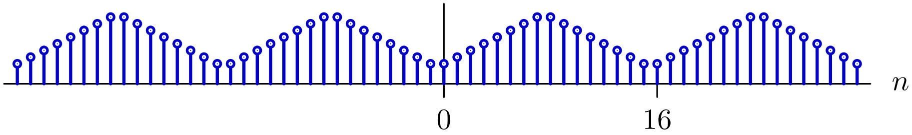
by first replicating one period in reverse order.



The resulting "folded" function does not have a step discontinuity in value (although there is a discontinuity in slope).

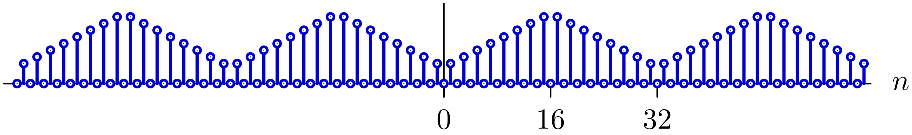
Discrete Cosine Transform (DCT)

The idea in the Discrete Cosine Transform (DCT) is to avoid introducing step discontinuities in periodic extension.



To simplify taking a transform, stretch the folded function in time by inserting zeros between successive samples, double the values (to preserve the DC value), and shift result 1 step right.

$$g[n] = g[n + 4N]$$

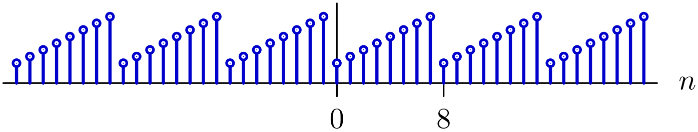


The resulting signal is symmetric about $n = 0$, periodic in $4N$, and contains only odd numbered samples.

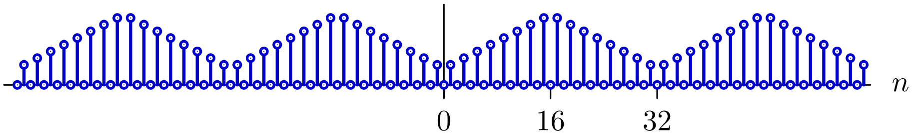
Discrete Cosine Transform (DCT)

The **DFT** of the folded, stretched, doubled, and shifted signal is the **DCT** of the original function.

$$f[n] = f[n+8]$$



$$g[n] = g[n+4N]$$

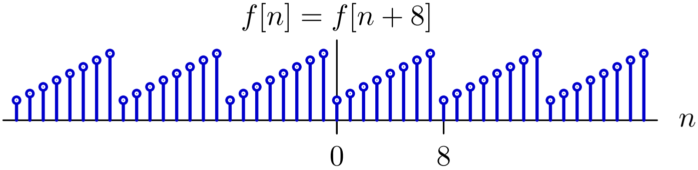


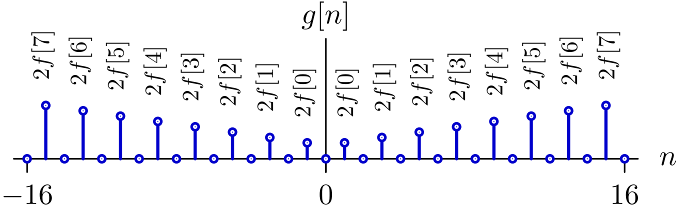
$$g[n] \overset{\text{DFT}}{\Longrightarrow} G[k]$$

$$f[n] \overset{\text{DCT}}{\Longrightarrow} F_C[k] = G[k]$$

Discrete Cosine Transform (DCT)

The **DFT** of the folded, stretched, doubled, and shifted signal is the **DCT** of the original function.





$$G[k] = \frac{1}{4N}\sum_{n=\langle 4N\rangle} g[n]e^{-j\frac{2\pi k}{4N}n} = \frac{1}{4N}\sum_{m=0}^{N-1} 2f[m]\left(e^{-j\frac{2\pi k}{4N}(2m+1)} + e^{j\frac{2\pi k}{4N}(2m+1)}\right)$$

$$= \frac{1}{N}\sum_{m=0}^{N-1} f[m]\cos\left(\frac{2\pi k}{4N}(2m+1)\right) = \underbrace{\frac{1}{N}\sum_{m=0}^{N-1} f[m]\cos\left(\frac{\pi k}{N}\left(m+\frac{1}{2}\right)\right) = F_C[k]}_{\text{analysis formula}}$$

Discrete Cosine transform (DCT)

The DCT of $f[n]$ is equal to the DFT of a folded, stretched, doubled, and shifted version of $f[n]$.

These operations define the DCT **analysis equation**:

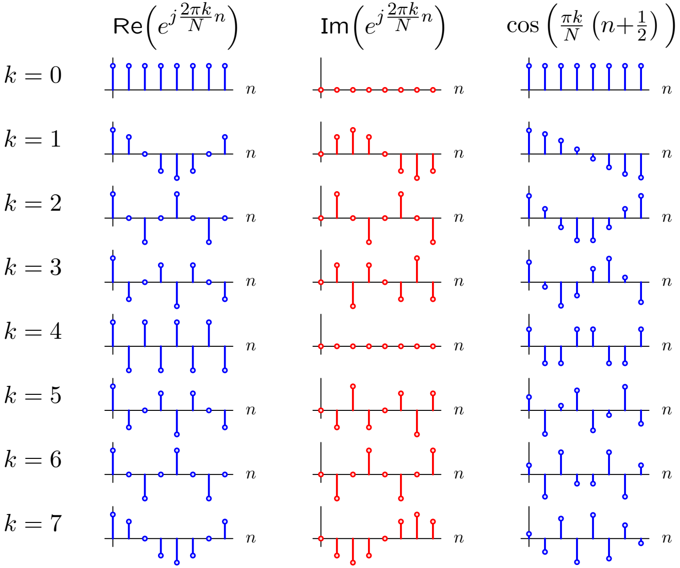
$$F_C[k] = \frac{1}{N} \sum_{n=0}^{N-1} f[n] \cos\left(\frac{\pi k}{N}\left(n + \frac{1}{2}\right)\right)$$

in terms of **basis functions**:

$$\phi_k[n] = \cos\left(\frac{\pi k}{N}\left(n + \frac{1}{2}\right)\right)$$

Comparison of DFT and DCT Basis Functions

DFT basis functions (real and imaginary parts) and DCT basis functions.



The DCT basis functions are symmetric or antisymmetric about $n = 3.5$ at half-integer multiples of the fundamental frequency.

DCT Basis Functions

As with the DFT, the DCT basis functions are **orthogonal** to each other.

$$\phi_k[n] = \cos\left(\frac{\pi k}{N}\left(n+\frac{1}{2}\right)\right)$$

$$\phi_l[n] = \cos\left(\frac{\pi l}{N}\left(n+\frac{1}{2}\right)\right)$$

$$\frac{1}{N}\sum_{n=0}^{N-1}\phi_k^*[n]\phi_l[n] = \frac{1}{N}\sum_{n=0}^{N-1}\cos\left(\frac{\pi k}{N}\left(n+\frac{1}{2}\right)\right)\cos\left(\frac{\pi l}{N}\left(n+\frac{1}{2}\right)\right)$$

$$= \underbrace{\frac{1}{2N}\sum_{n=0}^{N-1}\cos\left(\frac{\pi(k-l)}{N}\left(n+\frac{1}{2}\right)\right)}_{\frac{1}{2}\delta[k-l]} + \underbrace{\frac{1}{2N}\sum_{n=0}^{N-1}\cos\left(\frac{\pi(k+l)}{N}\left(n+\frac{1}{2}\right)\right)}_{\frac{1}{2}\delta[k]\delta[l]}$$

$$= \begin{cases} 1 & \text{if } k = l = 0 \\ 1/2 & \text{if } k = l \neq 0 \\ 0 & k \neq l \end{cases}$$

The sum over time of the product of two different basis functions is zero.

Find DCT Synthesis Equation (using Orthogonality)

We would like to express $f[n]$ as a weighted sum of DCT basis functions.

$$f[n] = \sum_{k=0}^{N-1} a_k \cos\left(\frac{\pi k}{N}\left(n+\frac{1}{2}\right)\right)$$

Multiply both sides by $\phi_l[n]$ and sum over n.

$$\sum_{n=0}^{N-1} f[n] \cos\left(\frac{\pi l}{N}\left(n+\frac{1}{2}\right)\right) = \sum_{n=0}^{N-1}\sum_{k=0}^{N-1} a_k \cos\left(\frac{\pi k}{N}\left(n+\frac{1}{2}\right)\right)\cos\left(\frac{\pi l}{N}\left(n+\frac{1}{2}\right)\right)$$

Left-hand side is $NF_C[l]$. Swap order of summation on the right-hand side.

$$NF_C[l] = \sum_{k=0}^{N-1} a_k \sum_{n=0}^{N-1} \cos\left(\frac{\pi k}{N}\left(n+\frac{1}{2}\right)\right)\cos\left(\frac{\pi l}{N}\left(n+\frac{1}{2}\right)\right)$$

$$= \sum_{k=0}^{N-1} a_k \left(\frac{1}{2}\delta[k-l] + \frac{1}{2}\delta[k]\delta[l]\right) = \begin{cases} Na_0 & \text{if } l = 0 \\ \frac{1}{2}Na_l & \text{otherwise} \end{cases}$$

$$f[n] = \sum_{k=0}^{N-1} a_k \cos\left(\frac{\pi k}{N}\left(n+\frac{1}{2}\right)\right) = F_C[0] + 2\sum_{k=1}^{N-1} F_C[k] \cos\left(\frac{\pi k}{N}\left(n+\frac{1}{2}\right)\right)$$

Discrete Cosine Transform (DCT)

The Discrete Cosine Transform (DCT) is described by analysis and synthesis equations that are analogous to those of the DFT.

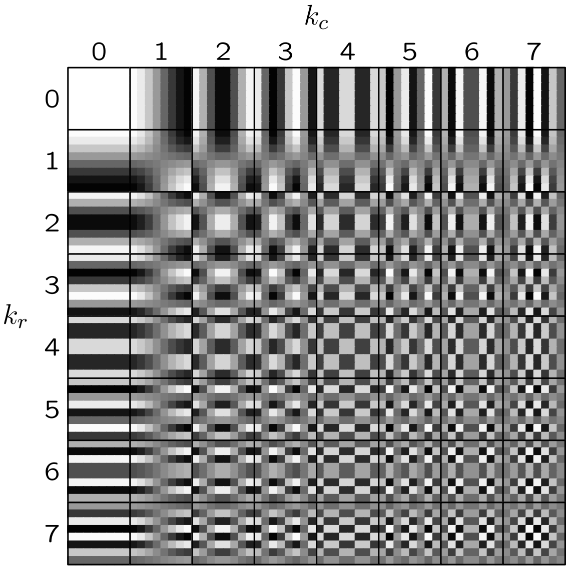
$$F_C[k] = \frac{1}{N} \sum_{n=0}^{N-1} f[n] \cos\left(\frac{\pi k}{N}\left(n+\frac{1}{2}\right)\right) \qquad \text{(analysis)}$$

$$f[n] = F_C[0] + 2 \sum_{k=1}^{N-1} F_C[k] \cos\left(\frac{\pi k}{N}\left(n+\frac{1}{2}\right)\right) \qquad \text{(synthesis)}$$

2D DCT Basis Functions

Grid of 8×8 basis functions organized in rows (k_r) and columns (k_c).

Each basis function has 8×8 elements organized by row r and column c.



Black represents -1, white represents $+1$.

Discrete Cosine Transform

The DCT has a number of useful properties:

- It maps spatial domain to **frequency domain** (much like DFT).
- If input has length N, then the output has **length N**.
- It is purely **real-valued** (unlike DFT).
- It **reduces discontinuities** caused by periodic extension of DFT.

However:

- It does **not have a “filtering” property**.

Basis Functions, Eigenfunctions, and Filtering

The **filtering** property of Fourier transforms results from the **eigenfunction** property of the Fourier basis functions.

Eigenfunction property: If the input to an LTI system is an eigenfunction, then the output is a scaled version of that same eigenfunction.

$$\phi(t) \longrightarrow \boxed{\text{LTI}} \longrightarrow \lambda\phi(t)$$

The basis functions for the Fourier transform are eigenfunctions of linear, time-invariant systems.

$$e^{j\omega_o t} \longrightarrow \boxed{\text{LTI}} \longrightarrow \lambda e^{j\omega_o t}$$

- scaling the amplitude of a complex exponential does not change the shape of the complex exponential
- shifting a complex exponential in time does not change the shape of the complex exponential

Filter property: If we express an input signal as a sum of eigenfunctions, then the output signal is a weighted sum of those same eigenfunctions.

Basis Functions, Eigenfunctions, and Filtering

The DCT cannot be used for filtering because shifts in time result in complicated changes in DCT coefficients.

$$F_c[k] = \frac{1}{N}\sum_{n=0}^{N-1} f[n]\cos\left(\frac{\pi k}{N}\left(n+\frac{1}{2}\right)\right)$$

$$\delta[n] \quad \stackrel{\text{DCT}}{\Longrightarrow} \quad \frac{1}{N}\cos\left(\frac{\pi k}{2N}\right)$$

$$\delta[n-1] \quad \stackrel{\text{DCT}}{\Longrightarrow} \quad \frac{1}{N}\cos\left(\frac{3\pi k}{2N}\right)$$

Delaying the signal changes the basis function used to represent the signal!

Discrete Cosine Transform

The DCT has a number of useful properties:

- It maps spatial domain to **frequency domain** (much like DFT).
- It is purely **real-valued** (unlike DFT).
- If input has length N, then the output has **length N**.
- It **reduces discontinuities** caused by periodic extension of DFT.

However:

- It does **not have a "filtering" property**.

But the DCT represents patches of a smooth image very efficiently. For that reason, it is widely used in audio and image **compression**.

Data Compression

Fourier methods underlie many common data compression schemes, including JPEG (for pictures), MP3 (for audio) and MPEG (for video).

Example: **JPEG** (Joint Photographic Experts Group) Encoding

1. color encoding: RGB $\rightarrow$ YCrCb
2. 2D DCT (discrete cosine transform): a kind of Fourier series
3. quantization to achieve perceptual compression (lossy)
4. run-length and Huffman encoding (lossless)

We will focus on steps 2 & 3: the DCT and quantization of its components.

- the image is broken into 8×8 pixel blocks
- each block is represented by its 8×8 DCT coefficients
- each DCT coefficient is quantized, using higher resolutions for coefficients with greater perceptual importance

Quantization

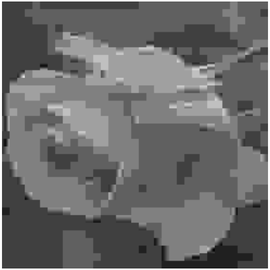
DCT amplitudes are quantized by dividing by a frequency-dependent number $q[k_r, k_c]$ and then rounding to the nearest integer.

$$
\begin{array}{cc}
q[k_r, k_c] & k_c \quad \rightarrow \\
\begin{array}{c} \\ \\ \\ k_r \\ \downarrow \\ \\ \\ \\ \end{array} &
\begin{array}{cccccccc}
16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\
12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\
14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\
14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\
18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\
24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\
49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\
72 & 92 & 95 & 98 & 112 & 100 & 103 & 99
\end{array}
\end{array}
$$

These values were chosen to represent human sensitivities.
High frequencies are more coarsely quantized than low frequencies.

Different tables of this form are used to implement different "qualities."

JPEG: Results



1%: 1666 bytes

10%: 2550 bytes

20%: 3595 bytes

40%: 5318 bytes

80%: 10994 bytes

100%: 47k bytes

Summary

The number of bits used to represent a signal is of critical importance in modern communication systems.

Modern compression systems combine lossless compression techniques (such as LZW, Huffman, and zip) with perceptual (lossy) compression based on Fourier representations.

The Discrete Cosine Transform (DCT) is a close relative of the DFT that is more easily compressed using block coding methods.

The DCT is not useful for filtering because its basis functions are not eigenfunctions of LTI systems.

The DCT does provide significantly improved data compaction and is widely used in both audio and video signal processing.

Question of the Day

Let $\phi_k[n]$ represent the k^{th} basis function for a 1D Discrete Cosine Transform with an analysis window of length $N = 12$.

Sketch $\phi_1[n]$ as a function of n.