Algorithms of Scientific Computing

Discrete Cosine Transform

Exercise 1: Simple JPEG Encoder

The Discrete Cosine Transform is the key to JPEG compression. Blocks of 8 × 8 pixels are transformed to the frequency domain to be in a format which is more suited for compression. The code template provides you with the pixel values of the given 8 × 8 block of the well-known test image Lena.

a) In a greyscale image, the pixel values are usually encoded with 8 bit with values in [0, 255]. The DCT, however, works on [−128, 127]. Write a function that normalises the pixel values.

b) The DCT transforms an 8 × 8 block from the spatial domain to the frequency domain. Use

\[
F_{uv} = \frac{1}{\sqrt{2N}} c_u c_v \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f_{xy} \cos \left( \frac{(2x + 1)u\pi}{2N} \right) \cos \left( \frac{(2y + 1)v\pi}{2N} \right)
\]

where

\[
c_{uv} = \begin{cases} 
\frac{1}{\sqrt{2}} & \text{if } u, v = 0 \\
1 & \text{otherwise.}
\end{cases}
\]

What is the complexity of your DCT routine? What is the meaning of \(F_{00}\)?

c) The coefficients \(F_{uv}\) are divided by quantisation values \(Q_{uv}\) and rounded to the nearest integer. Quantisation is a lossy process; high quantisation coefficients result in a high compression factor,
though at the expense of image quality. Implement the quantisation step. A common choice for the quantisation matrix is given in the code template. As mentioned in the lecture, a full JPEG encoder would now apply further compression techniques.

d) Derive the IDCT and write a decoder for our simple JPEG block encoder. Compare the original image with the subsequently encoded and then decoded image.

Exercise 2: Discrete Cosine Transform

We start with a dataset \((f_{-N+1}, \ldots, f_N) \in \mathbb{R}^{2N}\), which fulfils the following symmetry constraint:

\[ f_{-n} = f_n \quad \text{for } n = 1, \ldots, N - 1 \]

a) Show that the corresponding Fourier coefficients

\[ F_k = \frac{1}{2N} \sum_{n=-N+1}^{N} f_n \omega_{2N}^{-kn} \]  

are real values only and can be written as:

\[ F_k = \frac{1}{N} \left( \frac{1}{2} f_0 + \sum_{n=1}^{N-1} f_n \cos \left( \frac{\pi nk}{N} \right) + \frac{1}{2} f_N \cos(\pi k) \right). \]

b) Show that the \( F_k \) is symmetric too.

c) Let \( \text{FFT}(f, N) \) be a procedure that computes the coefficients \( F_k \) efficiently (according to equation (1)) from a field \( f \) which consists of \( 2N \) values \( f_n \). (The result is written back to field \( f \))

Write a short procedure \( \text{DCT}(g, N) \) which uses procedure \( \text{FFT} \) to compute the coefficients \( F_k \) for \( k = 0, \ldots, N \) from equation (2) for the (non-symmetrical) data \( f_0, \ldots, f_N \), stored in the parameter field \( g \).

Exercise 3: Fast Discrete Cosine Transform

Determine the butterfly scheme for equation (1) from the previous exercise. Divide the dataset \( f_n \) of length \( 2N \) into a dataset \( g_n := f_{2n} \), containing all values with an even index, and a dataset \( h_n := f_{2n-1} \), with all values with an odd index. Which symmetries can be found in \( g_n \) and \( h_n \)? Of which kind (Cosine/Sine Transformation, DFT with real data) are the according DFTs of length \( N \)? Which symmetries can be found if the dataset \( f_n \) fulfils the following symmetry constraint:

\[ f_{-n} = f_{n+1}. \]