Chapter 8
Lossy Compression Algorithms

8.1 Introduction

- Lossless compression algorithms do not deliver *compression ratios* that are high enough. Hence, most multimedia compression algorithms are *lossy*.

- What is *lossy compression*?
  - The compressed data is not the same as the original data, but a close approximation of it.
  - Yields a much higher compression ratio than that of lossless compression.

8.2 Distortion Measures

- The three most commonly used distortion measures in image compression are:

  – *mean square error* (MSE) σ^2,

$$\sigma^2 = \frac{1}{N}\sum_{n=1}^{N}(x_n - y_n)^2 \tag{8.1}$$

  where x_n, y_n, and N are the input data sequence, reconstructed data sequence, and length of the data sequence respectively.

  – *signal to noise ratio* (SNR), in decibel units (dB),

$$SNR = 10\log_{10}\frac{\sigma_x^2}{\sigma_d^2} \tag{8.2}$$

  where σ_x^2 is the average square value of the original data sequence and σ_d^2 is the MSE.

  – *peak signal to noise ratio* (PSNR),

$$PSNR = 10\log_{10}\frac{x_{peak}^2}{\sigma_d^2} \tag{8.3}$$

8.3 The Rate-Distortion Theory

- Provides a framework for the study of tradeoffs between Rate and Distortion.

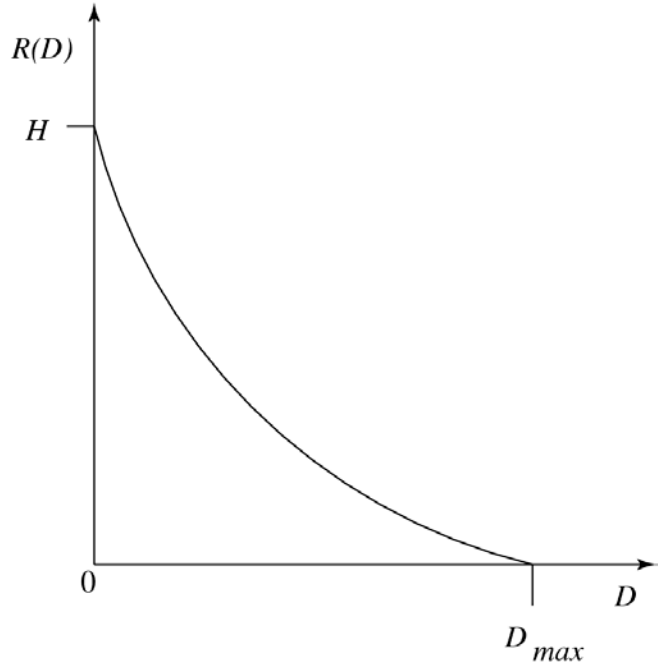


Fig. 8.1: Typical Rate Distortion Function.

8.4 Quantization

- Reduce the number of distinct output values to a much smaller set.

- Main source of the "loss" in lossy compression.

- Three different forms of quantization.
  - Uniform: midrise and midtread quantizers.
  - Nonuniform: companded quantizer.
  - Vector Quantization.

8.4.1 Uniform Scalar Quantization

- A uniform scalar quantizer partitions the domain of input values into equally spaced intervals, except possibly at the two outer intervals.
  - The output or reconstruction value corresponding to each interval is taken to be the midpoint of the interval.
  - The length of each interval is referred to as the *step size*, denoted by the symbol Δ.

- Two types of uniform scalar quantizers:
  - Midrise quantizers have even number of output levels.
  - Midtread quantizers have odd number of output levels, including zero as one of them (see Fig. 8.2).

- For the special case where $\Delta = 1$, we can simply compute the output values for these quantizers as:

$$Q_{midrise}(x) = \lceil x \rceil - 0.5 \tag{8.4}$$

$$Q_{midtread}(x) = \lfloor x + 0.5 \rfloor \tag{8.5}$$

- Performance of an M level quantizer. Let $B = \{b_0, b_1, \ldots, b_M\}$ be the set of decision boundaries and $Y = \{y_1, y_2, \ldots, y_M\}$ be the set of reconstruction or output values.

- Suppose the input is uniformly distributed in the interval $[-X_{max}, X_{max}]$. The rate of the quantizer is:

$$R = \lceil \log_2 M \rceil \tag{8.6}$$

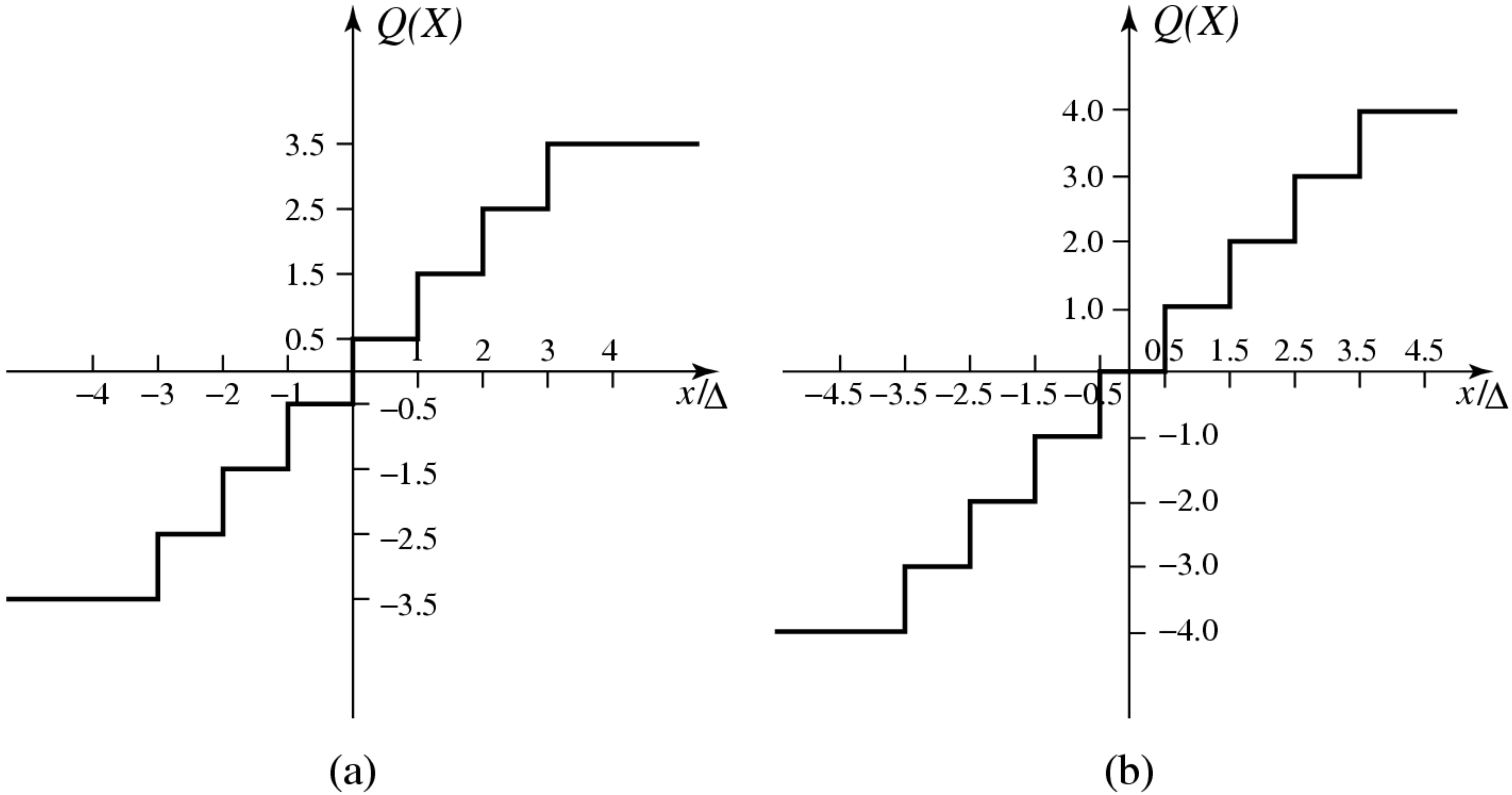


Fig. 8.2: Uniform Scalar Quantizers: (a) Midrise, (b) Midtread.

Quantization Error of Uniformly Distributed Source

- **Granular distortion**: quantization error caused by the quantize for bounded input.

  - To get an overall figure for granular distortion, notice that decision boundaries b_i for a midrise quantizer are $[(i-1)\Delta, i\Delta]$, $i = 1..M/2$, covering positive data X (and another half for negative X values).

  - Output values y_i are the midpoints $i\Delta - \Delta/2$, $i = 1..M/2$, again just considering the positive data. The total distortion is twice the sum over the positive data, or

$$D_{gran} = 2\sum_{i=1}^{\frac{M}{2}} \int_{(i-1)\Delta}^{i\Delta} \left(x - \frac{2i-1}{2}\Delta\right)^2 \frac{1}{2X_{max}}\, dx \tag{8.8}$$

- Since the reconstruction values y_i are the midpoints of each interval, the quantization error must lie within the values $[-\frac{\Delta}{2}, \frac{\Delta}{2}]$. For a uniformly distributed source, the graph of the quantization error is shown in Fig. 8.3.

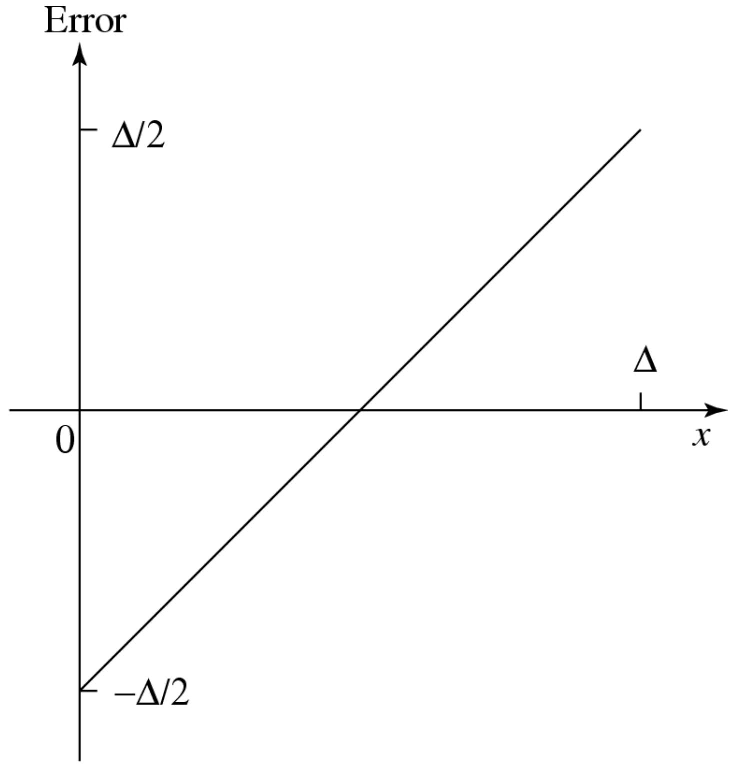


Fig. 8.3: Quantization error of a uniformly distributed source.

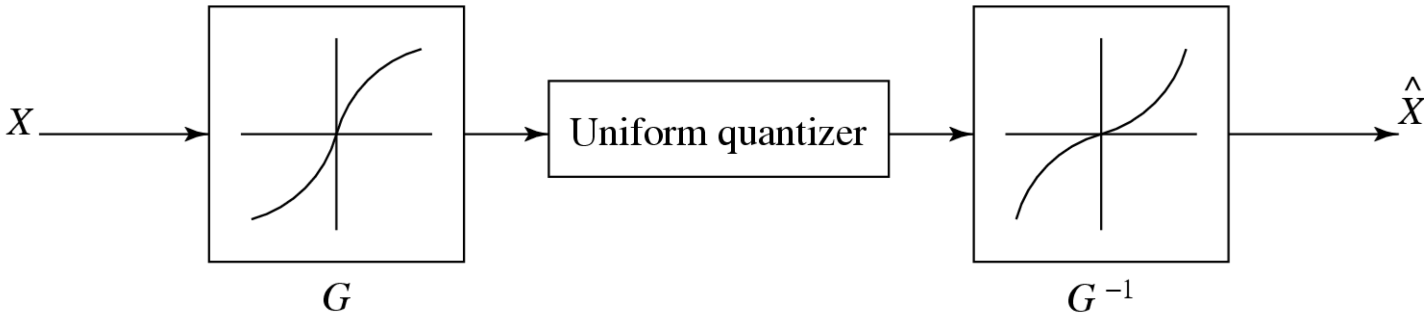


Fig. 8.4: Companded quantization.

- *Companded quantization* is **nonlinear**.

- As shown above, a *compander* consists of a *compressor function* G, a uniform quantizer, and an *expander function* G^{-1}.

- The two commonly used companders are the μ-law and A-law companders.

8.4.3 Vector Quantization (VQ)

- According to Shannon's original work on information theory, any compression system performs better if it operates on vectors or groups of samples rather than individual symbols or samples.
- Form vectors of input samples by simply concatenating a number of consecutive samples into a single vector.
- Instead of single reconstruction values as in scalar quantization, in VQ code *vectors* with n components are used. A collection of these code vectors form the *codebook*.

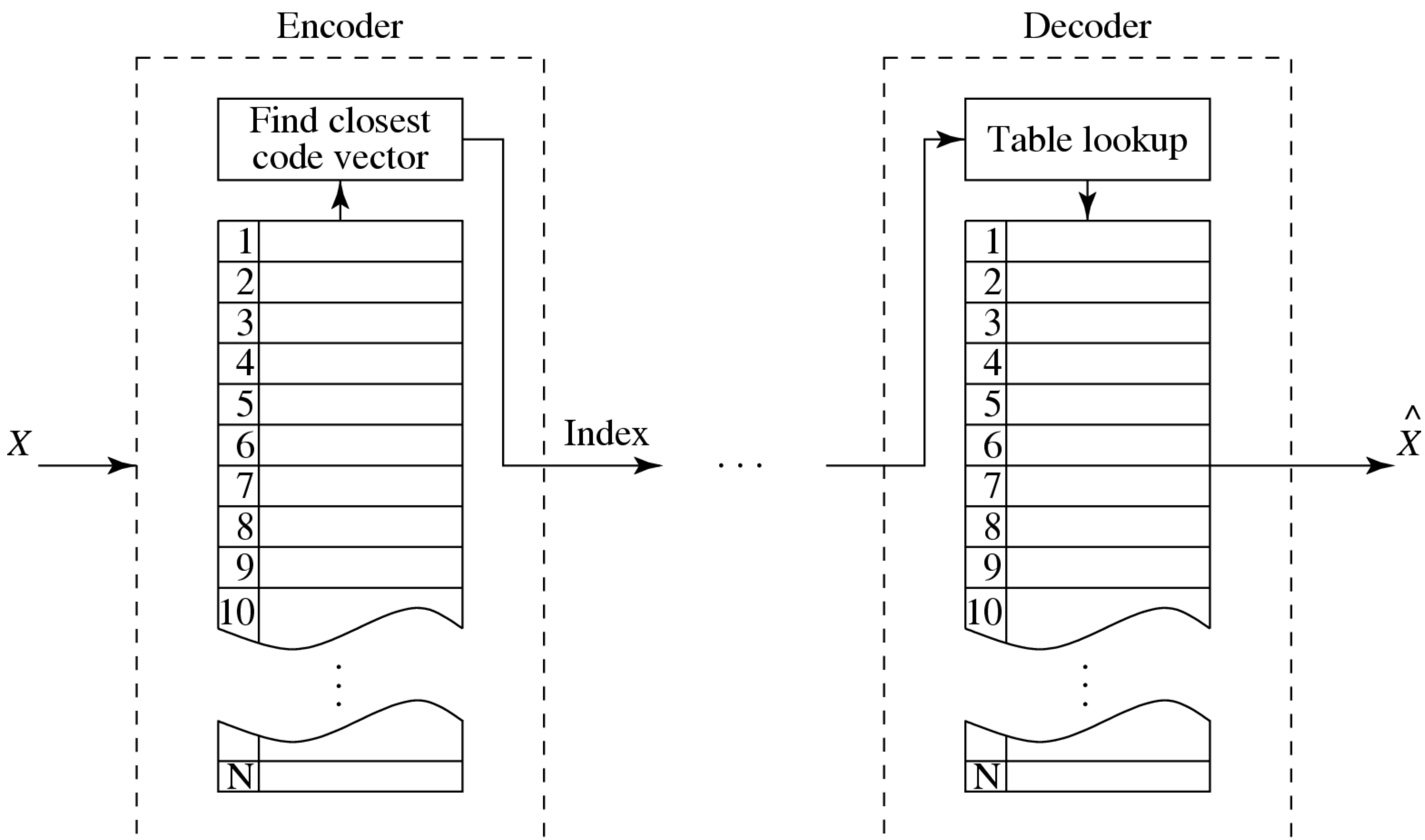


Fig. 8.5: Basic vector quantization procedure.

8.5 Transform Coding

- **The rationale behind transform coding**:

- If $\mathbf{Y}$ is the result of a linear transform $\mathbf{T}$ of the input vector $\mathbf{X}$ in such a way that the components of $\mathbf{Y}$ are much less correlated, then $\mathbf{Y}$ can be coded more efficiently than $\mathbf{X}$.

- If most information is accurately described by the first few components of a transformed vector, then the remaining components can be coarsely quantized, or even set to zero, with little signal distortion.

- Discrete Cosine Transform (DCT) will be studied first. In addition, we will examine the Karhunen-Loève Transform (KLT) which *optimally* decorrelates the components of the input $\mathbf{X}$.

8.5.1 Spatial Frequency and DCT

- *Spatial frequency* indicates how many times pixel values change across an image block.
- The DCT formalizes this notion with a measure of how much the image contents change in correspondence to the number of cycles of a cosine wave per block.
- The role of the DCT is to *decompose* the original signal into its DC and AC components; the role of the IDCT is to *reconstruct* (re-compose) the signal.

Definition of DCT:

- Given an input function $f(i, j)$ over two integer variables i and j (a piece of an image), the 2D DCT transforms it into a new function $F(u, v)$, with integer u and v running over the same range as i and j. The general definition of the transform is:

$$F(u,v) = \frac{2C(u)C(v)}{\sqrt{MN}} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \cos\frac{(2i+1)\cdot u\pi}{2M} \cdot \cos\frac{(2j+1)\cdot v\pi}{2N} \cdot f(i,j) \qquad (8.15)$$

where i, u = 0, 1, … , M − 1; j, v = 0, 1, … , N − 1; and the constants $C(u)$ and $C(v)$ are determined by

$$C(\xi) = \begin{cases} \frac{\sqrt{2}}{2} & if \quad \xi = 0, \\ 1 & otherwise. \end{cases} \qquad (8.16)$$

2D Discrete Cosine Transform (2D DCT):

$$F(u,v) = \frac{C(u)C(v)}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} \cos\frac{(2i+1)u\pi}{16} \cos\frac{(2j+1)v\pi}{16} f(i,j) \qquad (8.17)$$

where i, j, u, v = 0, 1, . . . , 7, and the constants $C(u)$ and $C(v)$ are determined by Eq. (8.16).

2D Inverse Discrete Cosine Transform (2D IDCT):

The inverse function is almost the same, with the roles of $f(i, j)$ and $F(u, v)$ reversed, except that now $C(u)C(v)$ must stand inside the sums:

$$\tilde{f}(i,j) = \sum_{u=0}^{7} \sum_{v=0}^{7} \frac{C(u)C(v)}{4} \cos\frac{(2i+1)u\pi}{16} \cos\frac{(2j+1)v\pi}{16} F(u,v) \qquad (8.18)$$

where i, j, u, v = 0, 1, . . . , 7.

1D Discrete Cosine Transform (1D DCT):

$$F(u) = \frac{C(u)}{2} \sum_{i=0}^{7} \cos \frac{(2i+1)u\pi}{16} f(i) \qquad (8.19)$$

where i = 0, 1, . . . , 7, u = 0, 1, . . . , 7.

1D Inverse Discrete Cosine Transform (1D IDCT):

$$\tilde{f}(i) = \sum_{u=0}^{7} \frac{C(u)}{2} \cos \frac{(2i+1)u\pi}{16} F(u) \qquad (8.20)$$

where i = 0, 1, . . . , 7, u = 0, 1, . . . , 7.

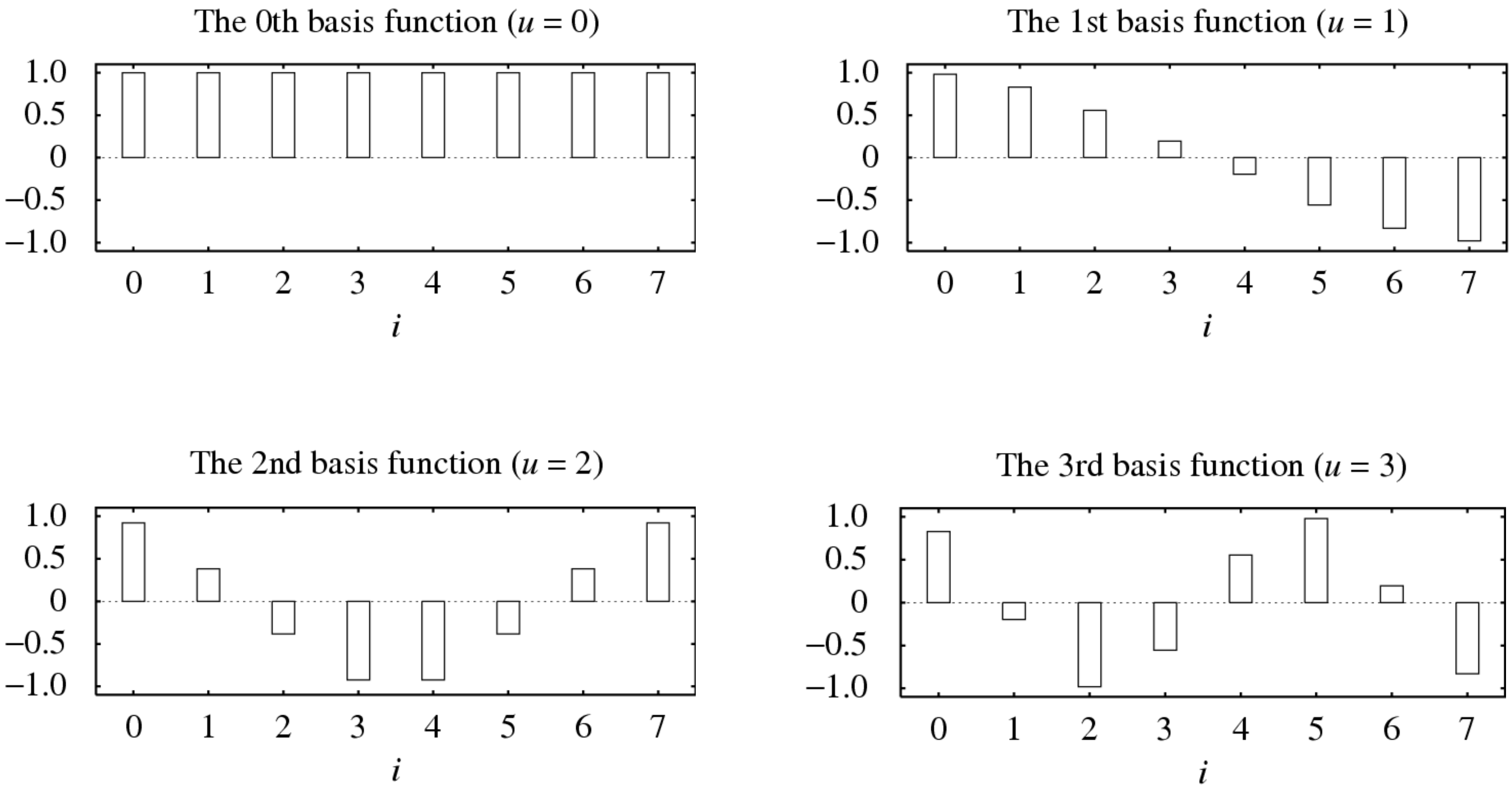


Fig. 8.6: The 1D DCT basis functions.

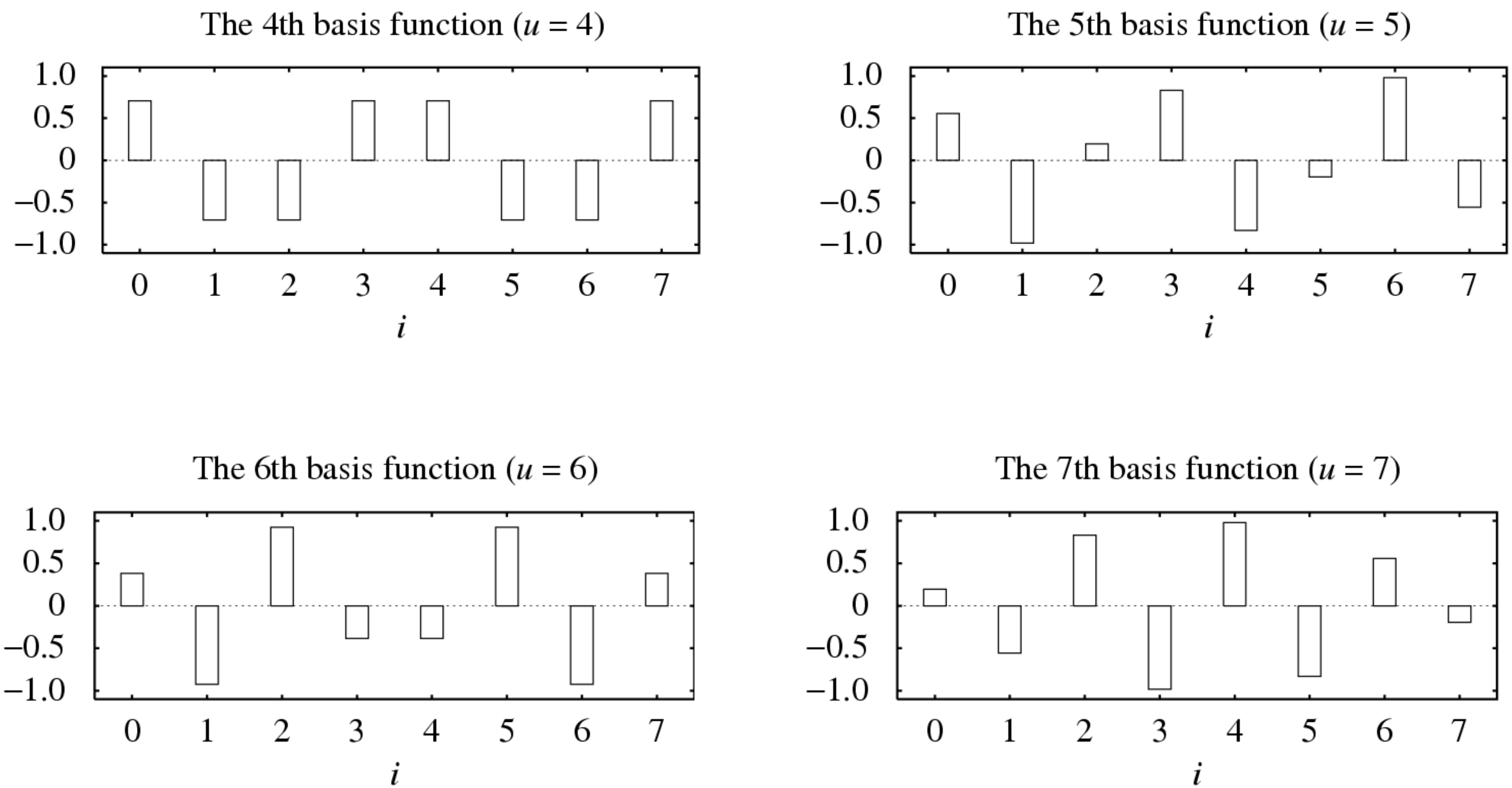


Fig. 8.6 (Cont'd): The 1D DCT basis functions.

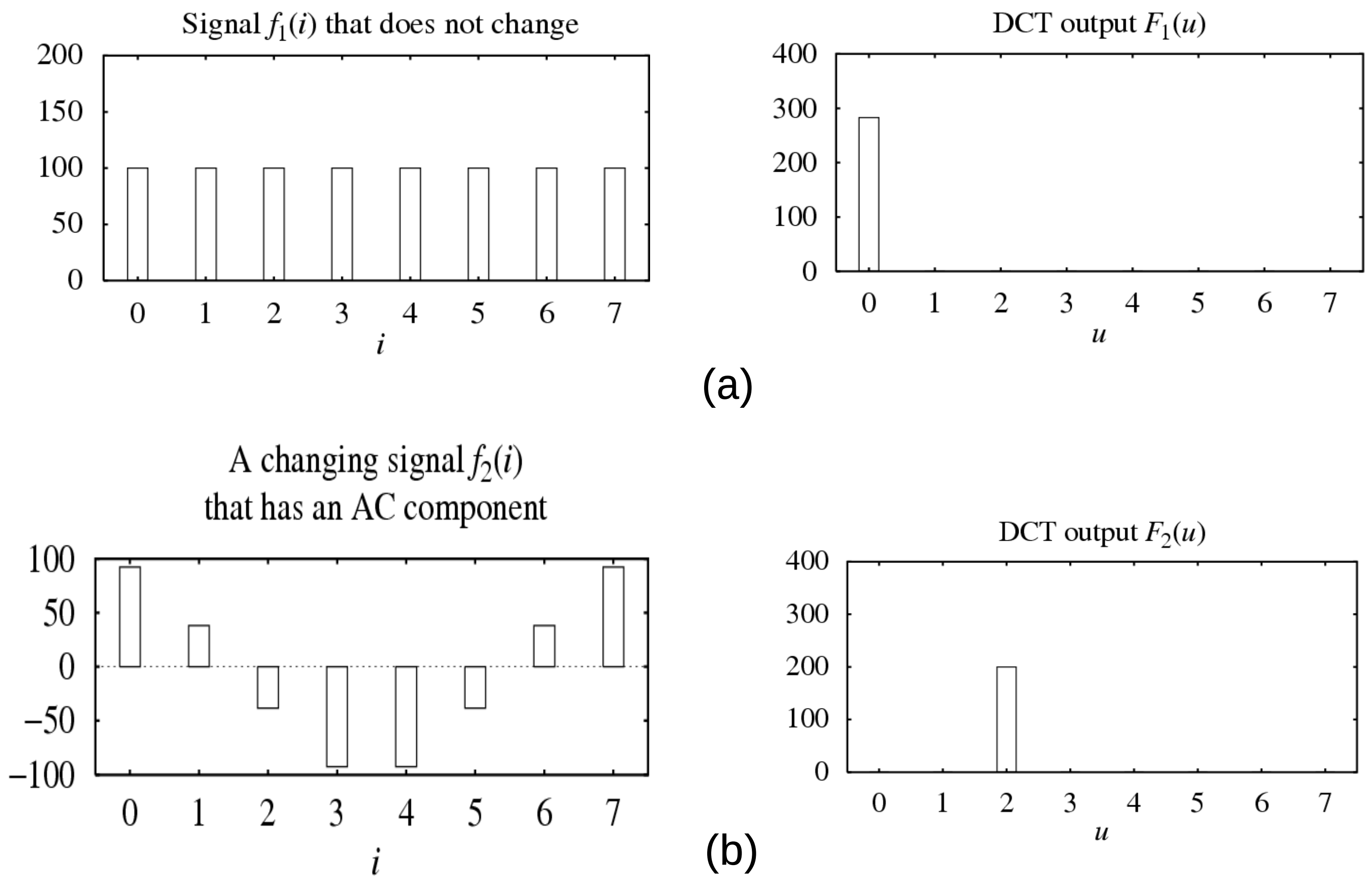
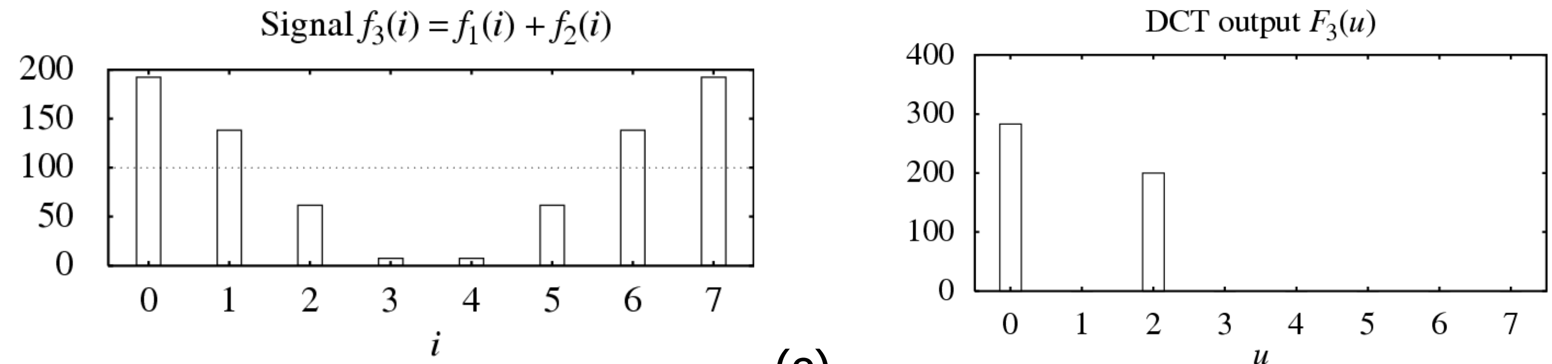
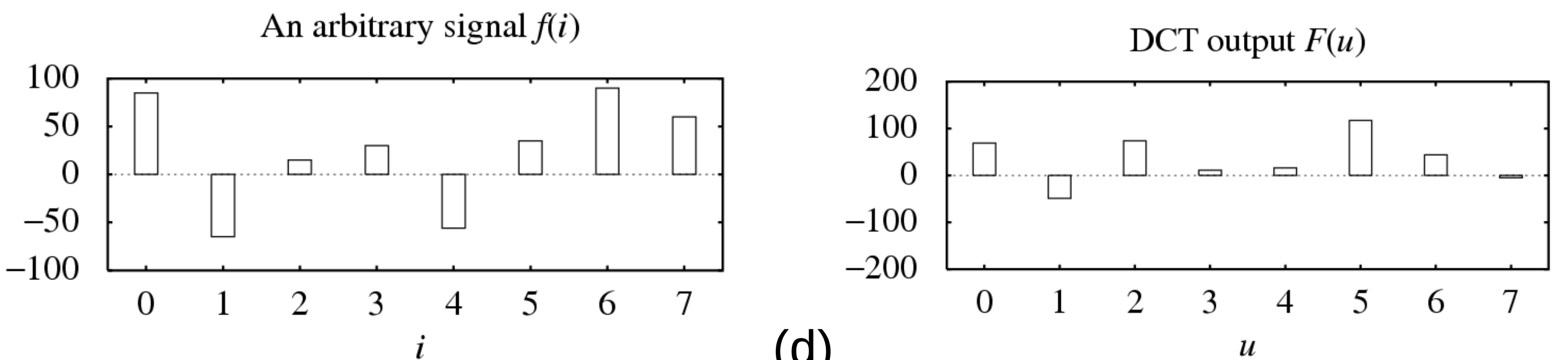


Fig. 8.7: Examples of 1D Discrete Cosine Transform:
(a) A DC signal $f_1(i)$, **(b)** An AC signal $f_2(i)$.



(c)



(d)

Fig. 8.7 (Cont'd): Examples of 1D Discrete Cosine Transform: **(c)** $f_3(i) = f_1(i)+f_2(i)$, and **(d)** an arbitrary signal $f(i)$.

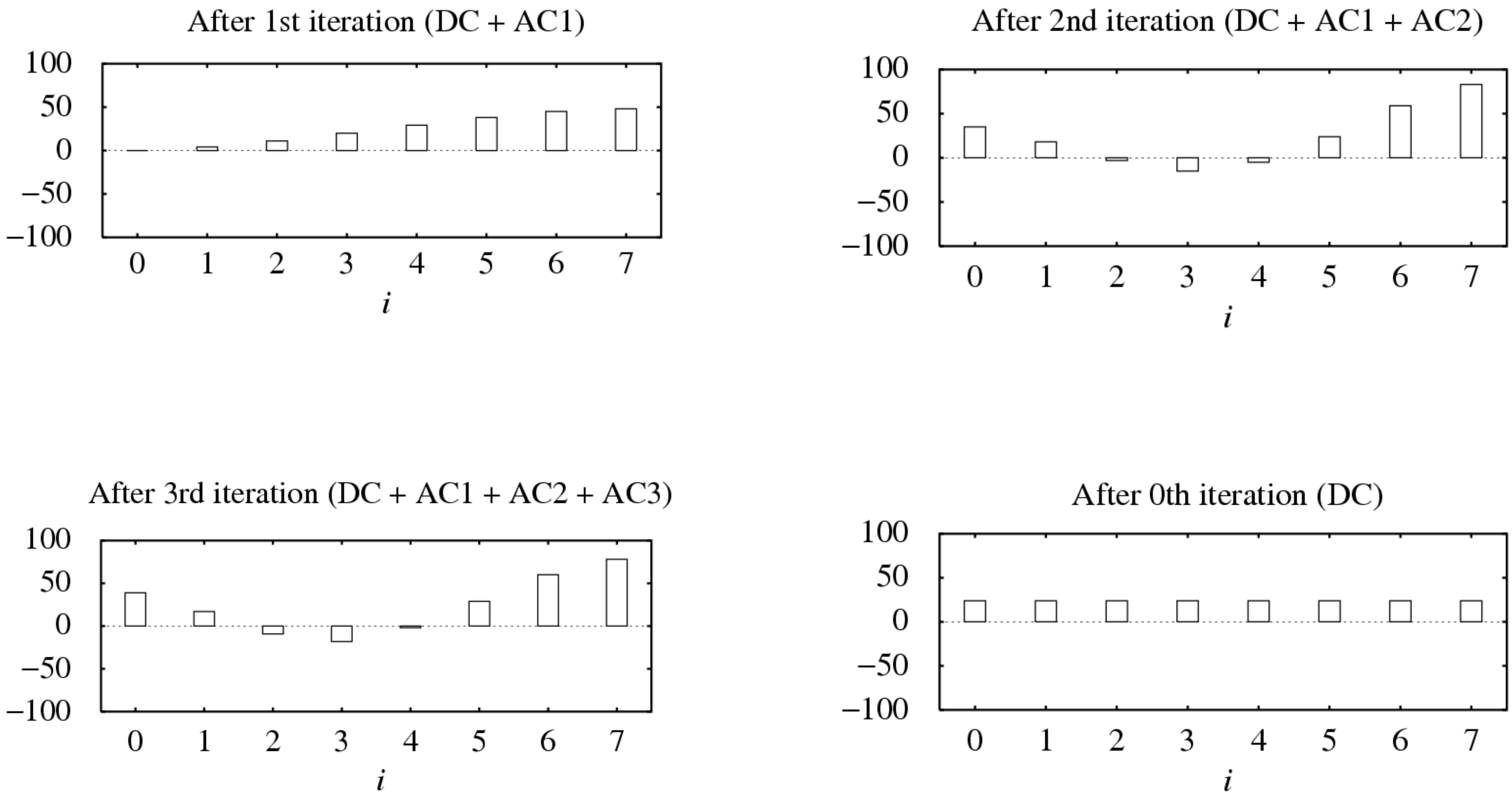


Fig. 8.8: An example of 1D IDCT.

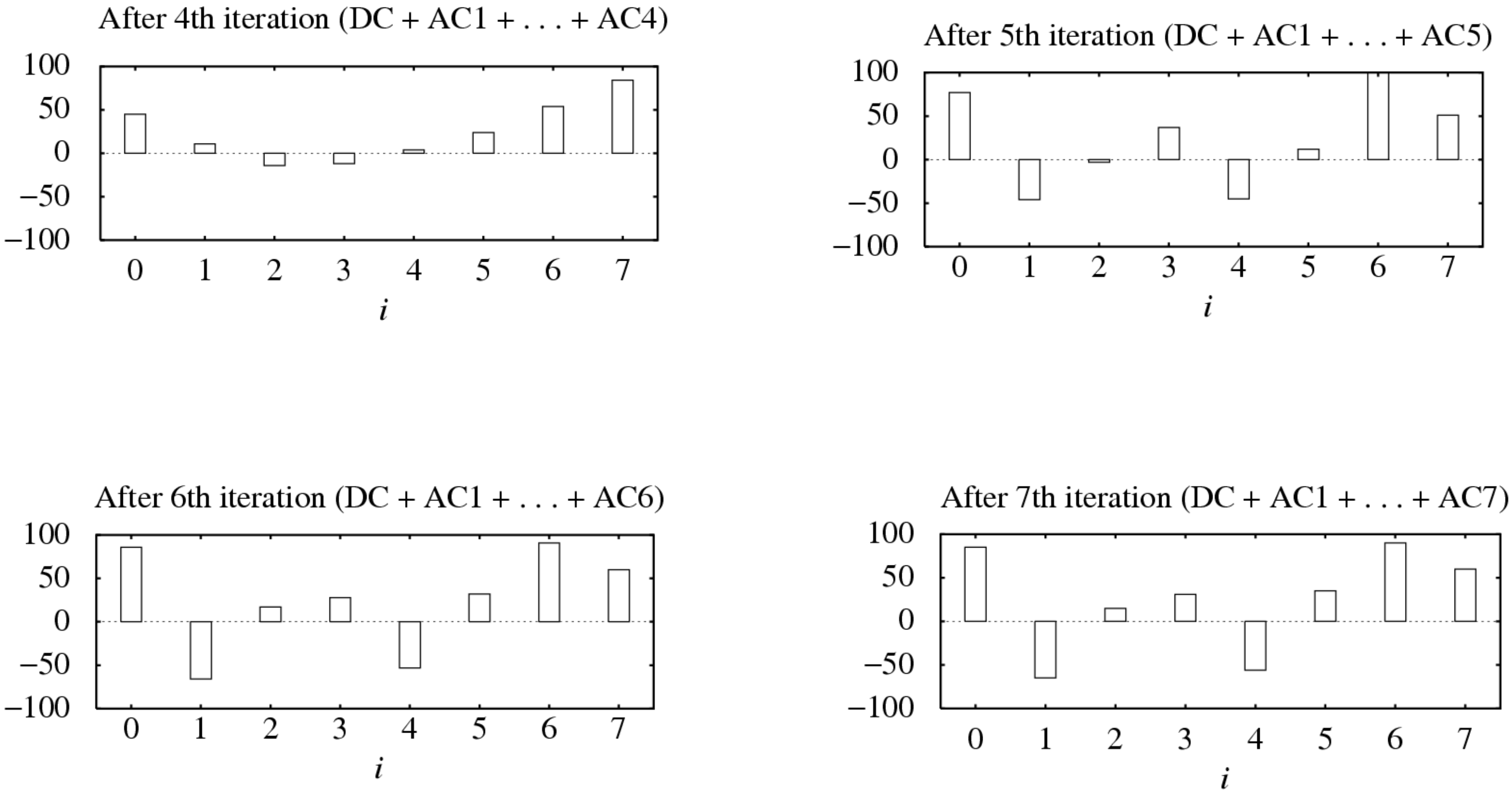


Fig. 8.8 (Cont'd): An example of 1D IDCT.

The DCT is a linear transform:

- In general, a transform T (or function) is linear, iff

$$\mathcal{T}(\alpha p + \beta q) = \alpha \mathcal{T}(p) + \beta \mathcal{T}(q), \qquad (8.21)$$

where α and β are constants, p and q are any functions, variables or constants.

- From the definition in Eq. 8.17 or 8.19, this property can readily be proven for the DCT because it uses only simple arithmetic operations.

The Cosine Basis Functions

- Function $B_p(i)$ and $B_q(i)$ are *orthogonal*, if

$$\sum_i [B_p(i) \cdot B_q(i)] = 0 \qquad if \quad p \neq q \tag{8.22}$$

- Function $B_p(i)$ and $B_q(i)$ are *orthonormal*, if they are orthogonal and

$$\sum_i [B_p(i) \cdot B_q(i)] = 1 \qquad if \quad p = q \tag{8.23}$$

- It can be shown that:

$$\sum_{i=0}^{7} \left[\cos\frac{(2i+1)\cdot p\pi}{16} \cdot \cos\frac{(2i+1)\cdot q\pi}{16} \right] = 0 \qquad if \;\; p \neq q$$

$$\sum_{i=0}^{7} \left[\frac{C(p)}{2} \cos\frac{(2i+1)\cdot p\pi}{16} \cdot \frac{C(q)}{2} \cos\frac{(2i+1)\cdot q\pi}{16} \right] = 1 \qquad if \;\; p = q$$

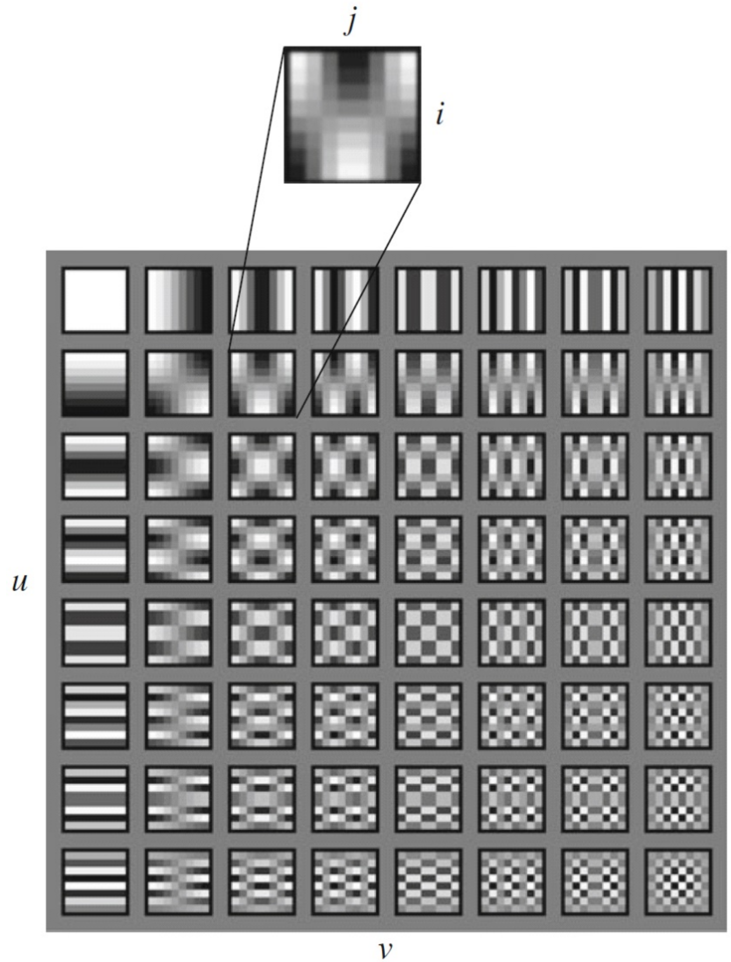


Fig. 8.9: Graphical Illustration of 8 × 8 2D DCT basis.

2D Basis Functions

- For a particular pair of *u* and *v*, the respective 2D basis function is:

$$\cos \frac{(2i+1) \cdot u\pi}{16} \cdot \cos \frac{(2j+1) \cdot v\pi}{16}, \qquad (8.24)$$

- The enlarged block shown in Fig. 8.9 is for the basis function:

$$\cos \frac{(2i+1) \cdot 1\pi}{16} \cdot \cos \frac{(2j+1) \cdot 2\pi}{16}.$$

2D Separable Basis

- The 2D DCT can be *separated* into a sequence of two, 1D DCT steps:

$$G(u, j) = \frac{1}{2} C(u) \sum_{i=0}^{7} \cos \frac{(2i + 1)u\pi}{16} f(i, j). \tag{8.25}$$

$$F(u, v) = \frac{1}{2} C(v) \sum_{j=0}^{7} \cos \frac{(2j + 1)v\pi}{16} G(u, j). \tag{8.26}$$

- It is straightforward to see that this simple change saves many arithmetic steps. The number of iterations required is reduced from 8 × 8 to 8+8.

2D DCT Matrix Implementation

- The above factorization of a 2D DCT into two 1D DCTs can be implemented by two consecutive matrix multiplications:

$$F(u, v) = \mathbf{T} \cdot f(i, j) \cdot \mathbf{T}^T. \tag{8.27}$$

- We will name $\mathbf{T}$ the *DCT-matrix*.

$$\mathbf{T}[i, j] = \begin{cases} \frac{1}{\sqrt{N}}, & \text{if } i = 0 \\ \sqrt{\frac{2}{N}} \cdot \cos\frac{(2j+1)\cdot i\pi}{2N}, & \text{if } i > 0 \end{cases} \tag{8.28}$$

Where i = 0, ... , N-1 and j = 0, ... , N-1 are the row and column indices, and the block size is N x N.

When $N = 8$, we have:

$$\mathbf{T_8}[i, j] = \begin{cases} \frac{1}{2\sqrt{2}}, & \text{if } i = 0 \\ \frac{1}{2} \cdot \cos \frac{(2j+1) \cdot i\pi}{16}, & \text{if } i > 0. \end{cases} \tag{8.29}$$

$$\mathbf{T_8} = \begin{bmatrix} \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & \cdots & \frac{1}{2\sqrt{2}} \\ \frac{1}{2} \cdot \cos \frac{\pi}{16} & \frac{1}{2} \cdot \cos \frac{3\pi}{16} & \frac{1}{2} \cdot \cos \frac{5\pi}{16} & \cdots & \frac{1}{2} \cdot \cos \frac{15\pi}{16} \\ \frac{1}{2} \cdot \cos \frac{\pi}{8} & \frac{1}{2} \cdot \cos \frac{3\pi}{8} & \frac{1}{2} \cdot \cos \frac{5\pi}{8} & \cdots & \frac{1}{2} \cdot \cos \frac{15\pi}{8} \\ \frac{1}{2} \cdot \cos \frac{3\pi}{16} & \frac{1}{2} \cdot \cos \frac{9\pi}{16} & \frac{1}{2} \cdot \cos \frac{15\pi}{16} & \cdots & \frac{1}{2} \cdot \cos \frac{45\pi}{16} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{1}{2} \cdot \cos \frac{7\pi}{16} & \frac{1}{2} \cdot \cos \frac{21\pi}{16} & \frac{1}{2} \cdot \cos \frac{35\pi}{16} & \cdots & \frac{1}{2} \cdot \cos \frac{105\pi}{16} \end{bmatrix}. \tag{8.30}$$

2D IDCT Matrix Implementation

The 2D IDCT matrix implementation is simply:

$$f(i, j) = \mathbf{T}^T \cdot F(u, v) \cdot \mathbf{T}. \tag{8.31}$$

- See the textbook for step-by-step derivation of the above equation.
  - The key point is: the DCT-matrix is orthogonal, hence,

$$\mathbf{T}^T = \mathbf{T}^{-1}.$$

Comparison of DCT and DFT

- The discrete cosine transform is a close counterpart to the Discrete Fourier Transform (DFT). DCT is a transform that only involves the real part of the DFT.
- For a continuous signal, we define the continuous Fourier transform F as follows:

$$\mathcal{F}(\omega) = \int_{-\infty}^{\infty} f(t) e^{-i\omega t}\, \mathrm{d}t. \tag{8.32}$$

Using Euler's formula, we have

$$e^{ix} = \cos(x) + i\sin(x) \tag{8.33}$$

- Because the use of digital computers requires us to discretize the input signal, we define a DFT that operates on 8 samples of the input signal $\{f_0, f_1, \ldots, f_7\}$ as:

$$F_\omega = \sum_{x=0}^{7} f_x \cdot e^{-\frac{2\pi i \omega x}{8}} \tag{8.34}$$

Writing the sine and cosine terms explicitly, we have

$$F_\omega = \sum_{x=0}^{7} f_x \cos\left(\frac{2\pi\omega x}{8}\right) - i\sum_{x=0}^{7} f_x \sin\left(\frac{2\pi\omega x}{8}\right) \qquad (8.35)$$

- The formulation of the DCT that allows it to use only the cosine basis functions of the DFT is that we can cancel out the imaginary part of the DFT by making a symmetric copy of the original input signal.

- DCT of 8 input samples corresponds to DFT of the 16 samples made up of original 8 input samples and a symmetric copy of these, as shown in Fig. 8.10.

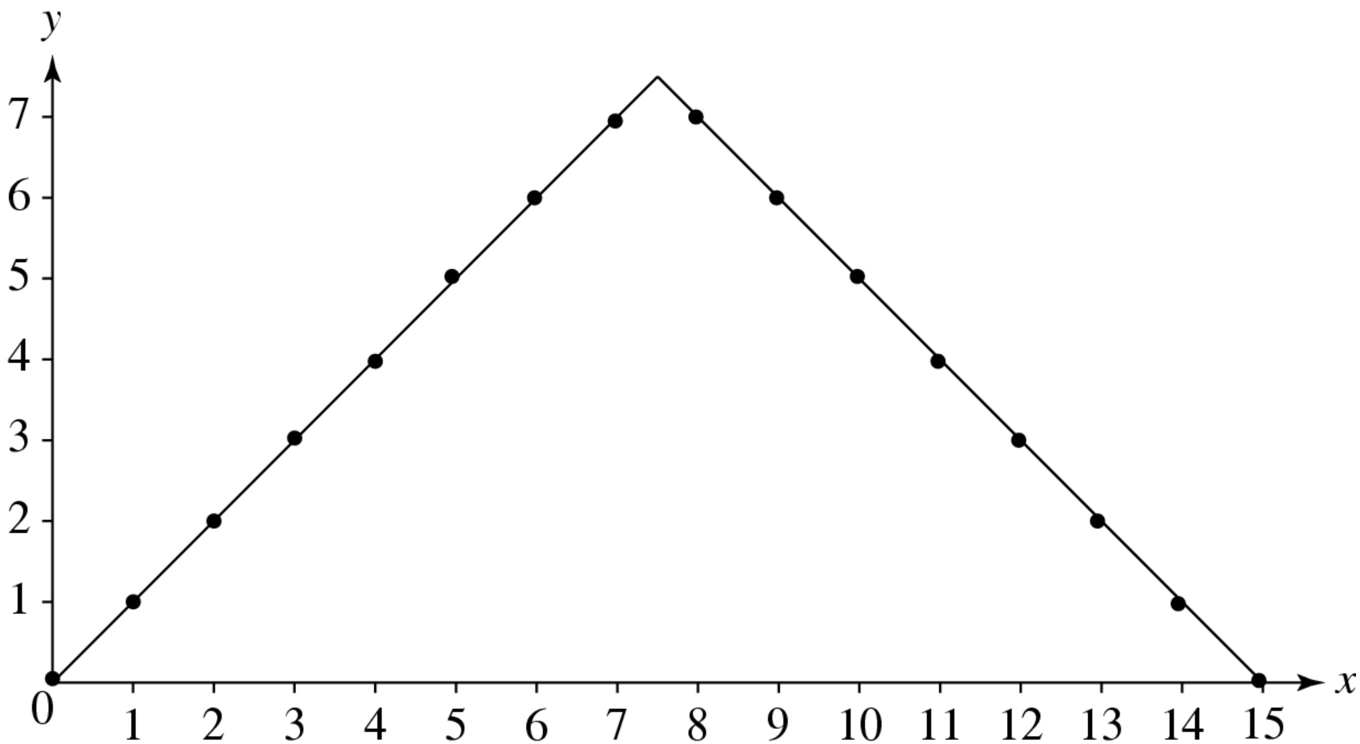


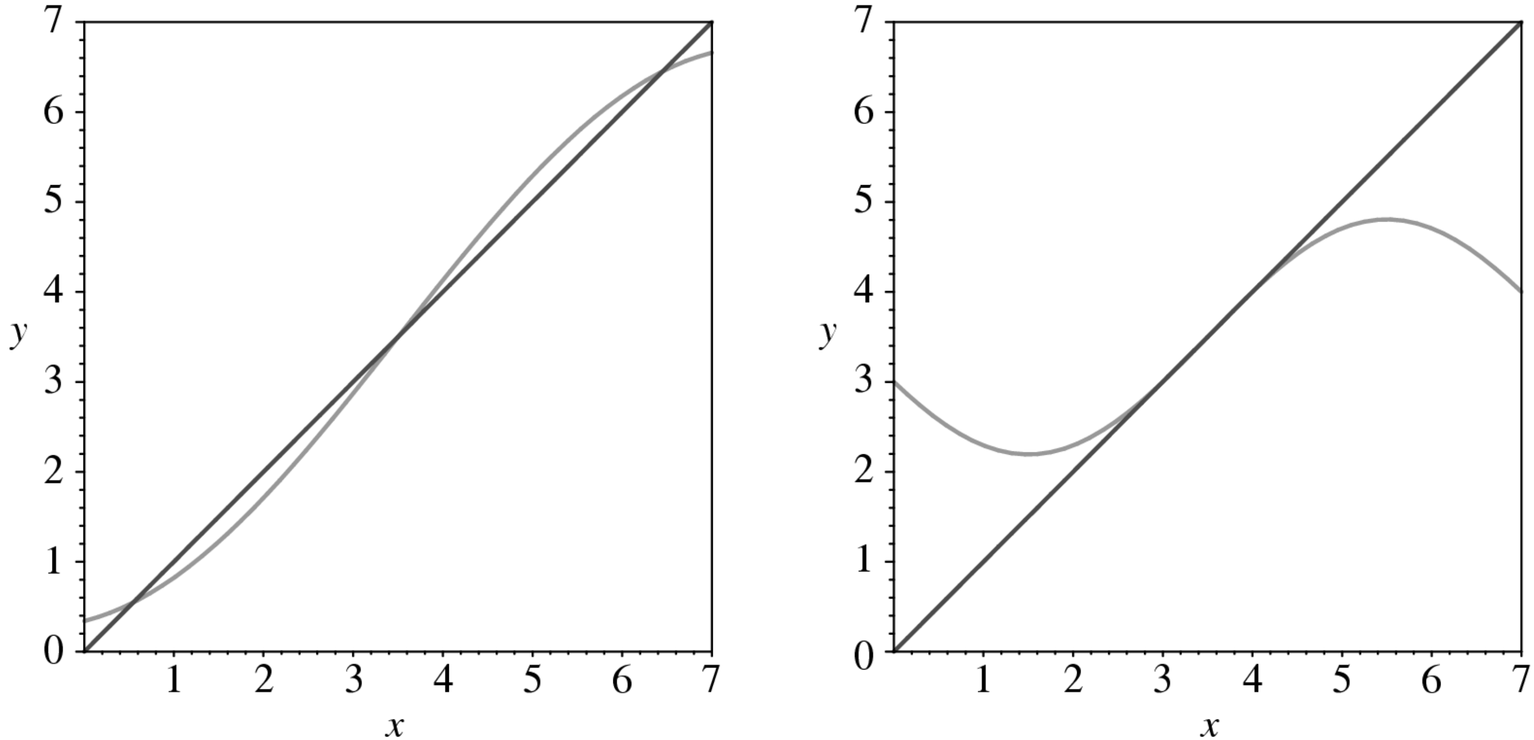
Fig. 8.10: Symmetric extension of the ramp function.

A Simple Comparison of DCT and DFT

Table 8.1 and Fig. 8.11 show the comparison of DCT and DFT on a ramp function, if only the first three terms are used.

Table 8.1: DCT and DFT coefficients of the ramp function

Ramp	DCT	DFT
0	9.90	28.00
1	-6.44	-4.00
2	0.00	9.66
3	-0.67	-4.00
4	0.00	4.00
5	-0.20	-4.00
6	0.00	1.66
7	-0.51	-4.00



(a) (b)

Fig. 8.11: Approximation of the ramp function: (a) 3 Term DCT Approximation, (b) 3 Term DFT Approximation.