(c) $\mathcal{C}_4(\mathbf{v})$ given by (9.43).

9.34 Let $\mathbf{v} \in \mathbf{R}^M$ with $\|\mathbf{v}\|^2 = 1$ and consider $\mathcal{C}(\mathbf{v}) = -\sum_{k=1}^{M} f(v_k^2)$ where

$$f(t) = \begin{cases} \ln(t), & t \neq 0 \\ 0, & t = 0 \end{cases}$$

Show that $\mathcal{C}(\mathbf{v})$ is a cost function.

9.35 Let $\mathbf{v} \in \mathbb{R}^M$ and consider the *nonnormalized Shannon entropy function*

$$\mathcal{C}(\mathbf{v}) = \sum_{k=1}^{M} s(v_k)\, v_k^2 \ln(v_k^2)$$

with the convention that $0\ln(0) = 0$. Here $s(t)\colon [0,\infty) \to \{-1, 1\}$ is given by

$$s(t) = \begin{cases} -1, & 0 \leq t < 1 \\ 1, & t \geq 1 \end{cases}$$

Show that $\mathcal{C}(\mathbf{v})$ is a cost function.

9.36 Is the entropy function (4.9) from Definition 4.3 a cost function? Either prove it is or provide a counterexample to show that it is not.

9.37 Write a best basis algorithm (use Algorithm 9.1 as a guide) for matrix input.

9.4 THE FBI FINGERPRINT COMPRESSION STANDARD

According to its Web site [26], the Federal Bureau of Investigation (FBI) started collecting and using fingerprints in 1902, and the Identification Division of the FBI was established in 1921. Bradley, Brislawn, and Hopper [8] reported that the FBI had about 810,000 fingerprint cards in 1924. Each card consists of 14 prints. Each finger is printed, flat impressions are taken of both thumbs, and each hand is printed simultaneously. In 1992, the Identification Division was renamed the Criminal Justice Information Services (CJIS) Division. In 1996, the CJIS fingerprint collection consisted of over 200 million print cards (see Bradley, Brislawn, Onyshczak, and Hopper [9]) and now totals over 250 million cards. The CJIS Web site [26] reports that about 80 million of these cards are digitized and that they add or digitally convert about 7000 new cards daily to their electronic database.

Each digitized print (an example is plotted in Figure 9.27) from a fingerprint card is scanned at 500 dots per inch (dpi) and each print is of varying size (e. g., a flat thumbprint measuring 0.875×1.875 inches is stored in a matrix of dimensions 455×975 pixels). According to Bradley, Brislawn, and Hopper [7], about 10.7 megabytes is necessary to store each fingerprint card. Thus over 800 terabytes of space would be needed to store the digitized cards as raw data. Given the daily

Figure 9.27 A digital fingerprint of size 832×832 pixels.

growth rate of the digitized card database, it is clear why the CJIS Division decided to use image compression to store the fingerprint cards.

The FBI first considered the JPEG image compression standard [46] but ultimately decided not to use it for fingerprint compression. The JPEG standard starts by partitioning the image into 8×8 blocks and then applying the *discrete cosine transform* to each block. Quantization is applied individually to each block, which is then encoded using a version of Huffman coding. The compressed image is recovered by unencoding the image data and then applying the discrete cosine transform to each 8×8 block. The method is highly effective, but the decoupling of the image into 8×8 blocks often gives the compressed image a "blocky" look. The image in Figure 9.27 was compressed using JPEG, and the center 96×96 portion is enlarged in Figure 9.28. The blocky structure of the compression method is evident in this figure.

In 1993 the FBI adopted the *Wavelet/Scalar Quantization Standard* (WSQ) [7]. This standard was developed by Thomas Hopper of the FBI in conjunction with researchers Jonathan Bradley and Christopher Brislawn at Los Alamos National Laboratory [8, 9, 10].

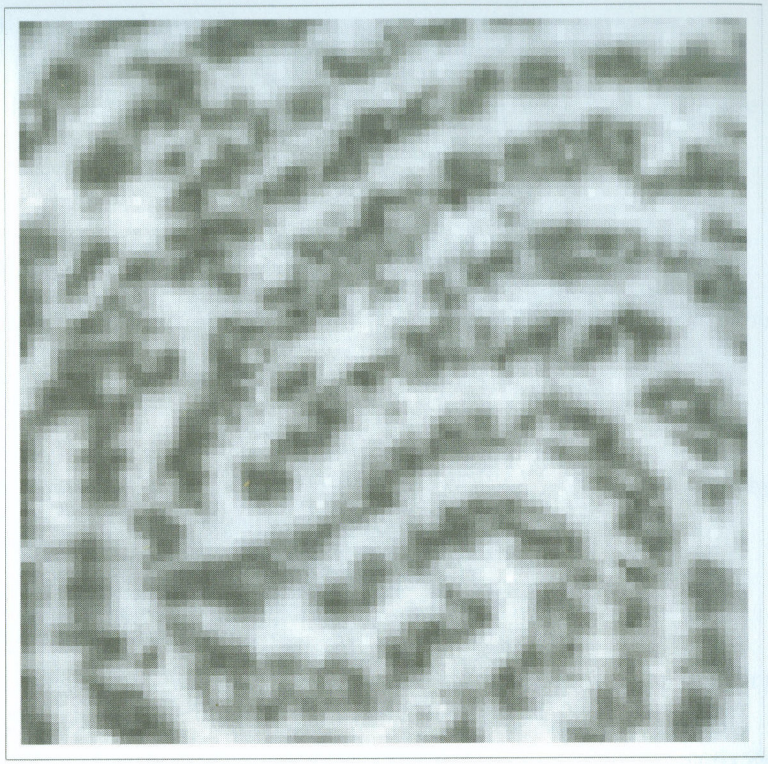

Figure 9.28 The enlarged 96×96 center of the JPEG-compressed fingerprint from Figure 9.27. The decoupling of 8×8 blocks is evident in this image.

Our interest in this standard is the fact that it uses the CDF97 biorthogonal filter pair developed in Section 8.3 and the discrete wavelet packet transform from Section 9.3. Unlike the image compression schemes described earlier in the book, this standard allows the user to set the compression rate *before* the quantization step. Although we can set the compression rate to whatever value we desire, we must be aware of the fact that there is an objective test for the effectiveness of this standard. In a US court of law, experts use 10 points on a fingerprint to uniquely identify the person whose finger produced the print (see Hawthorne [33]). If the compressed version of a digitized fingerprint does not preserve these 10 points accurately, then the compression scheme is useless in a court of law. Remarkably, a typical compression rate used by the FBI is 0.75 bit per pixel, or a 10.7:1 ratio if we consider 8 bits per pixel a 1:1 compression ratio.

The Basic Wavelet/Scalar Quantization Standard Algorithm

We now present the basic algorithm for implementing the FBI WSQ standard.

Algorithm 9.2 (FBI Wavelet/Scalar Quantization Standard) *This algorithm gives a basic description of the FBI WSQ standard developed by Bradley, Brislawn and Hopper [7, 8, 9, 10] for compressing digitized fingerprints. Suppose that matrix A holds a grayscale version of a digitized fingerprint image.*

1. *Normalize A.*
2. *Apply the discrete wavelet packet transform using the CDF97 biorthogonal filter pair. At each iteration of the packet transform, use the modified transformation (see Algorithm 8.3) to better handle boundary effects. Rather than using a best basis algorithm, use a prescribed basis for the packet transform.*
3. *Perform quantization on each portion of the transform.*
4. *Apply adaptive Huffman coding.*

■

Note: The material that follows is based on the work [7, 8, 9, 10] of the designers Bradley, Brislawn and Hopper of the WSQ standard.

We will look at each step of Algorithm 9.2 in more detail using the fingerprint plotted in Figure 9.27 as a running example. Each step contains some computations that we will not attempt to justify — to do so would be beyond the scope of this book. The standard designers have done much empirical work and analysis in developing the first three steps of the algorithm. For more details the interested reader is encouraged to see the references cited in the note above.

Normalizing the Image

The first step in Algorithm 9.2 is to normalize the digitized fingerprint image A. If μ is the mean value of the elements of A and m_1 and M_1 are the minimum and maximum elements of A, then we normalize A to obtain $\tilde{A}$, where the elements of $\tilde{A}$ are

$$\tilde{A}_{i,j} = \frac{A_{i,j} - \mu}{R}$$

Here

$$R = \frac{1}{128}\max\{M_1 - \mu, \mu - m_1\} \tag{9.46}$$

Bradley, Brislawn, and Hopper [7] remark that this normalization produces a mean of approximately zero in the approximation portion of the iterated packet transformation

(region 0 in Figure 9.29) that is computed in the second step of the algorithm. In this way the distribution of values in this portion of the transformation is similar to the distribution of other portions of the transformation, making the Huffman coding in the final step more efficient. For the fingerprint plotted in Figure 9.27, we have $\mu = 193.074$, $m_1 = 22$, and $M_1 = 253$. The mean of $\tilde{A}$ is 1.22171×10^{-15}.

Applying the Discrete Wavelet Packet Transform

The second step in Algorithm 9.2 is the application of a discrete wavelet packet transformation to $\tilde{A}$. This packet transform is constructed from the CDF97 biorthogonal filter pair from Section 8.3 and is modified via Algorithm 8.3 to better handle the boundary effects produced by the transform matrix. The similar lengths of each filter in the pair and the balanced smoothness of the corresponding scaling functions were desirable features for the designers of the standard.

The number of possible representations of the two-dimensional transform grows quickly as the number of iterations increase,[27] and it is thus interesting to learn how the standard designers choose the "best" packet representation. We introduced the best basis algorithm in Section 9.3 and it would seem natural to expect it to be used in conjunction with some cost function that measures the effectiveness of each portion of the transformation with regard to image compression. But the designers of the standard did not use the best basis algorithm — undoubtedly, the cost of computing a best basis for each fingerprint weighed in their decision not to use it. Unlike the compression of generic images, the fingerprint compression considers a class of images with many similar traits and tendencies. The authors of the standard [7] used empirical studies and analysis of the *power spectral density*[28] of fingerprint images to determine a static packet representation that could be used for *all* digitized fingerprints. The power spectral density analysis gives information about the frequencies at which values in the packet transform occur. This information is useful when designing a quantization method that will be most effective with the static wavelet packet representation selected by the standard designers.

For fingerprint compression, the static representation used in the WSQ standard is displayed in Figure 9.29. It is interesting to note that the schematic in Figure 9.29 implies that five iterations of the discrete wavelet packet transform are computed for the standard. The 64 portions of this discrete wavelet packet transform representation are numbered and referred to as the *bands* of the transform. Note that no portions of the first and third iterations are used, and only four bands of the fifth iteration are computed. We recognize bands 0 through 3 as the fifth iteration of the modified biorthogonal wavelet transform from Algorithm 8.3 applied to $\tilde{A}$. The transform of the fingerprint image from Figure 9.27 is plotted in Figure 9.30.

[27] Walnut remarks [62] that if A is $M \times M$ with $M = 2^i$, then there are more than $2^{M^2/2}$ possible wavelet packet representations.

[28] The power spectral density of a continuous signal can be viewed as the norm squared of its Fourier transform. If the signal is discrete, then the power spectral density is in terms of the norm squared of the Fourier series built from the signal.

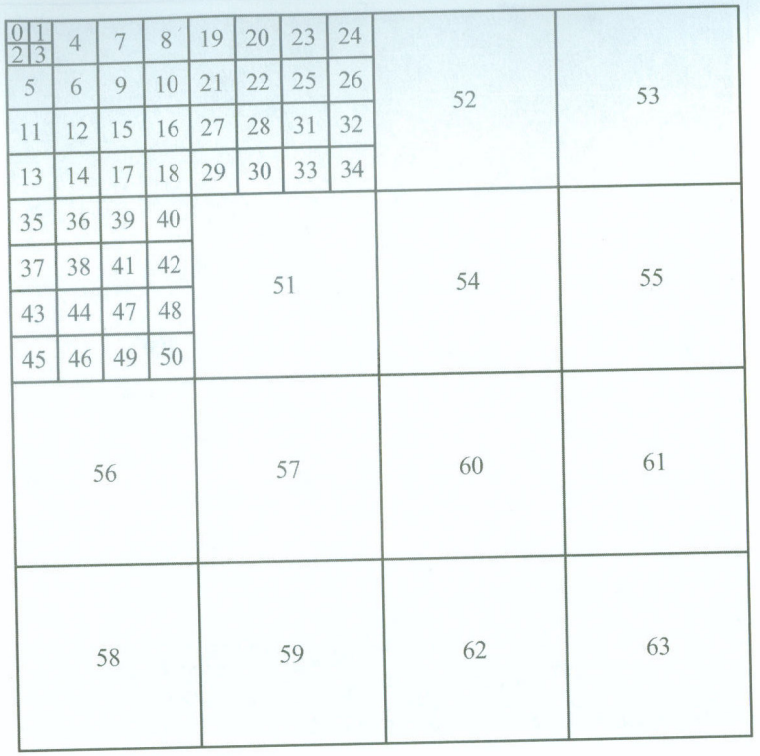

Figure 9.29 The wavelet packet representation used for the second step of Algorithm 9.2.

Performing Quantization

Perhaps the most important step in the WSQ standard algorithm is quantization. Remarkably, bands 60 through 63 of the transform are discarded. Mathematically, we can think of the elements in these bands as quantized to zero. Quantization is performed on bands 0 through 59 using the piecewise constant function

$$f(t,Q,Z) = \begin{cases} \left\lfloor \dfrac{t - Z/2}{Q} \right\rfloor + 1, & t > Z/2 \\ 0, & -Z/2 \le t \le Z/2 \\ \left\lceil \dfrac{t + Z/2}{Q} \right\rceil - 1, & t < -Z/2 \end{cases} \tag{9.47}$$

Here Z and Q are positive numbers. The breakpoints of this function are $\pm\left(\frac{Z}{2} + kQ\right)$, $k \in \mathbb{Z}$. The value Z is called the *zero bin width* since it determines the range of values quantized to zero. The value Q is called the *bin width*. The quantization function is plotted in Figure 9.31.

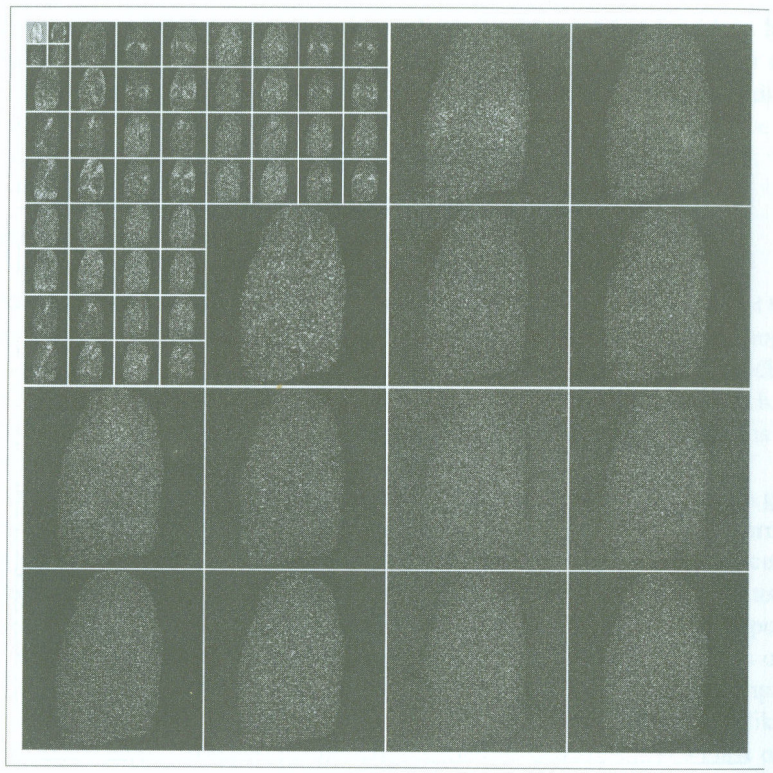

Figure 9.30 The wavelet packet transformation of the fingerprint from Figure 9.27.

For $k = 0, \ldots, 59$ we compute the bin widths Q_k and the zero bin width Z_k for band k and use these values to compute the quantized transform elements for each band. Indeed, if $w_{i,j}^k$ are the elements in band k, then we compute the quantized values

$$\hat{w}_{i,j}^k = f\left(w_{i,j}^k, Q_k, Z_k\right)$$

The zero bin width Z_k is assigned the value

$$Z_k = 1.2Q_k \tag{9.48}$$

for $k = 0, \ldots, 59$ and Q_k is computed using

$$Q_k = \begin{cases} \dfrac{1}{q}, & 0 \le k \le 3 \\ \dfrac{10}{qA_k \ln\left(\sigma_k^2\right)}, & 4 \le k \le 59 \end{cases} \tag{9.49}$$

The values Q_k depend on values q, A_k, and σ_k. The constants $A_k \approx 1$ chosen by the FBI are listed in Table 9.4.

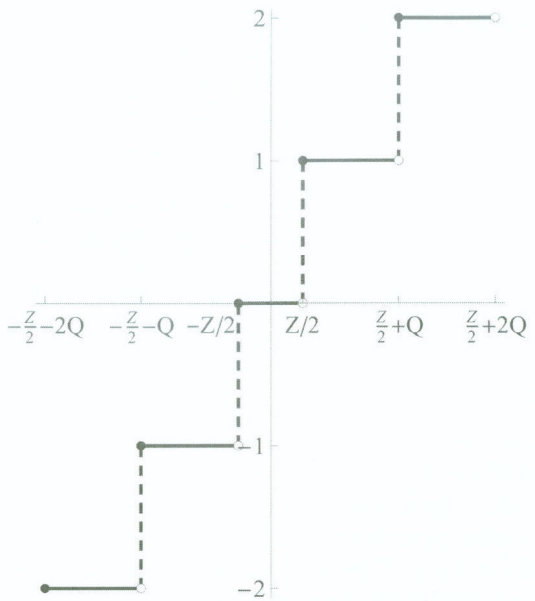

Figure 9.31 The quantization function for the FBI WSQ algorithm.

Table 9.4 The weights A_k used to determine the bin width (9.49).

Band	A_k
$4, \ldots, 51$	1.00
52, 56	1.32
53, 55, 58, 59	1.08
54, 57	1.42

The value σ_k^2 is the variance of a modified version of band k. In particular, we compute σ_k^2 from a subregion of band k. Suppose that the elements of band k are $\hat{w}_{i,j}^k$, where $0 \le i < M_k$ and $0 \le j < N_k$. Instead of using the entire $M_k \times N_k$ matrix, we instead set

$$\tilde{M}_k = \left\lfloor \frac{3M_k}{4} \right\rfloor \qquad \tilde{N}_k = \left\lfloor \frac{7N_k}{16} \right\rfloor \tag{9.50}$$

and consider the subregion with indices

$$i_{0,k} = \left\lfloor \frac{M_k}{8} \right\rfloor \le i \le i_{0,k} + \tilde{M}_k - 1 = i_{1,k}$$

$$j_{0,k} = \left\lfloor \frac{9N_k}{32} \right\rfloor \le j \le j_{0,k} + \tilde{N}_k - 1 = j_{1,k}$$

We compute σ_k^2 using the formula

$$\sigma_k^2 = \frac{1}{\tilde{M}_k \tilde{N}_k - 1} \sum_{i=i_{0,k}}^{i_{1,k}} \sum_{j=j_{0,k}}^{j_{1,k}} \left(\hat{w}_{i,j}^k - \mu_k\right)^2 \tag{9.51}$$

where μ_k is the mean of band k.

For example, band 39 is a square matrix with $M_{39} = N_{39} = 52$. Using (9.50) we see that $\tilde{M}_{39} = 39$ and $\tilde{N}_{39} = 22$. We compute the subregion boundary indices to be

$$i_{0,39} = \left\lfloor \frac{52}{8} \right\rfloor = 6, \quad i_{1,39} = 6 + 39 - 1 = 44$$

and

$$j_{0,39} = \left\lfloor \frac{9 \cdot 52}{32} \right\rfloor = 14, \quad j_{1,39} = 14 + 22 - 1 = 35$$

Using (9.51) we compute $\sigma_{39}^2 = 244.707$. If the variance $\sigma_k^2 < 1.01$ for any band k, then that band has all elements set to 0.

The final value that is used to compute the bin widths Q_k in (9.49) is the parameter q. This important value depends on the rate r selected for the compression process. A typical value of r is $r = 0.75$bpp and that is what we use for our example.

To compute q, we need some more values for each band. For $k = 0, \ldots, 59$, let

$$P_k = qQ_k = \begin{cases} 1, & 0 \le k \le 3 \\ \dfrac{10}{A_k \ln(\sigma_k^2)}, & 4 \le k \le 59 \end{cases}$$

If the dimensions of A are $M \times N$, then we set

$$m_k = \frac{MN}{M_k N_k}$$

where the dimensions of the matrix in band k are $M_k \times N_k$. In engineering terms we can think of m_k as the *downsampling* rate for band k. For example, the dimensions of the fingerprint in Figure 9.27 are 832×832 and the dimensions of band 39 are $M_{39} \times N_{39} = 52 \times 52$. Thus $m_k = \frac{832^2}{52^2} = 256$. The values of m_k for the fingerprint in Figure 9.27 are given in Table 9.5.

Table 9.5 The values m_k for the fingerprint in Figure 9.27.

Band	m_k
$0, \ldots, 3$	1024
$4, \ldots, 50$	256
$51, \ldots 59$	16

For $S = \sum_{k=0}^{59} \frac{1}{m_k}$, Bradley and Brislawn [5, 6] showed that if q is defined by

$$q = (0.4) \cdot 2^{r/S-1} \left(\prod_{k=0}^{59} \left(\frac{\sigma_k}{P_k} \right)^{1/m_k} \right)^{-1/S} \tag{9.52}$$

then the quantization scheme will produce a bit rate of r bits per pixel when the quantized data are encoded. Using Table 9.5 we find that $S = \frac{3}{4}$. Plugging these values and the values for σ_k, P_k, and $r = 0.75$ into (9.52) gives $q = 0.0375$. So that we can get a sense of the size of the bin widths Q_k, we have listed them in Table 9.6 for the fingerprint in Figure 9.27.

Table 9.6 The bin widths Q_k from (9.49) for the fingerprint in Figure 9.27.

Band	Q_k	Band	Q_k	Band	Q_k	Band	Q_k
0	26.702	15	3.513	30	3.869	45	2.844
1	26.702	16	2.966	31	3.474	46	2.956
2	26.702	17	3.007	32	3.723	47	3.453
3	26.702	18	2.475	33	3.090	48	2.989
4	2.779	19	4.569	34	3.173	49	3.703
5	2.480	20	3.939	35	4.224	50	3.126
6	2.594	21	4.592	36	4.244	51	4.196
7	2.735	22	3.880	37	3.641	52	4.978
8	2.639	23	2.820	38	3.741	53	4.294
9	2.858	24	3.101	39	4.855	54	5.871
10	2.521	25	2.909	40	4.361	55	5.015
11	2.647	26	3.100	41	4.496	56	4.486
12	2.796	27	5.103	42	3.775	57	5.341
13	2.355	28	4.553	43	2.731	58	4.052
14	2.397	29	4.552	44	2.863	59	4.830

Using the values of Q_k from Table 9.6 and $Z_k = 1.2Q_k$ in the quantization function (9.47), we quantize the first 60 bands of the discrete packet transform and replace the elements in the remaining four bands with zeros. The quantized packet transformation is plotted in Figure 9.32.

Encoding the Quantized Transform

The final step in Algorithm 9.2 is to encode the values in the quantized transformation. The WSQ standard uses an adaptive Huffman coding method that is much like the one used by the JPEG group (see Pennebaker and Mitchell [46]) for its compression standard for this task. This coding scheme is a bit more sophisticated than the basic Huffman coding method detailed in Appendix A. The interested reader is referred to Gersho and Gray [28]. Recall that the last four bands are not even included in the modified transform. The image is coded at 0.75 bit per pixel or at a compression ratio of 10.7 : 1.

Recovering the Compressed Image

To recover the compressed image, we perform the following steps:

1. Decode the Huffman codes.

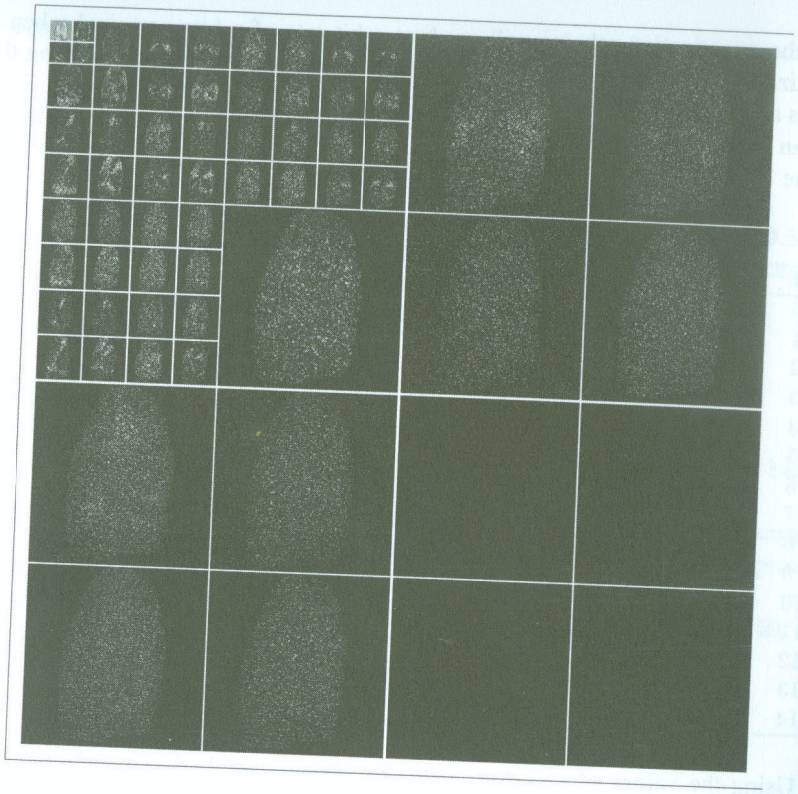

Figure 9.32 The quantized wavelet packet transformation of the fingerprint from Figure 9.27.

2. Apply a *dequantization function.*
3. Compute the inverse wavelet packet transform to obtain the normalized approximation $\tilde{B}$ to $\tilde{A}$.
4. For μ the mean of A and R given by (9.46), we compute

$$B_{i,j} = R\tilde{B}_{i,j} + \mu \tag{9.53}$$

The dequantization function is interesting and we describe it now. The quantization function $y = f(t, Q, Z)$ (9.47) maps values larger than $Z/2$ to positive numbers, values between and including $-Z/2$ and $Z/2$ to zero, and values smaller than $-Z/2$ to negative numbers. Then it makes sense that the dequantization function $d(y, Q, Z)$ should be piecewise defined with different formulas for positive, negative, and zero values. The easiest case is $y = 0$. Here we define $d(0, Q, Z) = 0$, since we have no way of knowing which $t \in [-Z/2, Z/2]$ yields $f(t, Q, Z) = 0$. If $t > Z/2$, we have

$$y = f(t, Q, Z) = \left\lfloor \frac{t - Z/2}{Q} \right\rfloor + 1 \quad \text{or} \quad y - 1 = \left\lfloor \frac{t - Z/2}{Q} \right\rfloor$$

Exact inversion is impossible due to the floor function. Instead of subtracting 1 from y, the designers of the standard subtracted a value $0 < C < 1$ instead to account for the floor function. So for $y > 0$, we define

$$d(y, Q, Z) = (y - C)Q + Z/2$$

We can add the same "fudge factor" in the case when $y < 0$ and thus define our dequantization function as

$$d(y, Q, Z) = \begin{cases} (y - C)Q + Z/2, & y > 0 \\ 0, & y = 0 \\ (y + C)Q - Z/2, & y < 0 \end{cases} \tag{9.54}$$

We apply $f\,(y, Q_k, Z_k)$ to each element in band k, $k = 0, \ldots, 59$. A typical value for C used by the FBI is $C = 0.44$. We have applied $d\,(y, Q_k, Z_k)$ to the packet transformation in Figure 9.32. The result is plotted in Figure 9.33. Note that we have added zero matrices in the place of bands 60 through 63.

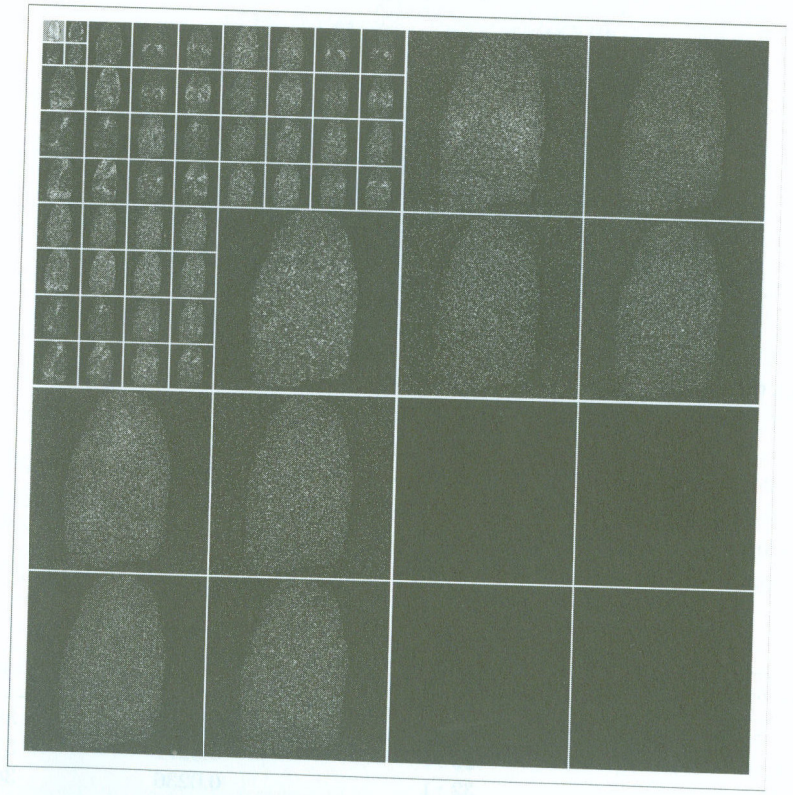

Figure 9.33 The dequantized wavelet packet transformation constructed by applying $d\,(y, Q_k, Z_k)$ to the matrix in Figure 9.32.

The last step in recovering the compressed fingerprint image is to apply (9.53) to each element in the image matrix in Figure 9.33. The result is plotted in Figure 9.34.

Figure 9.34 The fingerprint from Figure 9.27 compressed at 0.75 bit per pixel or at a ratio of 10.7 : 1.

The images in Figures 9.27 and 9.34 are difficult to distinguish. Indeed, the PSNR of the two images is 38.5462. In Figures 9.35 and 9.36 we have plotted the fingerprint of Figure 9.27 compressed at 0.40 and 0.25 bits per pixel, respectively. Table 9.7 summarizes the results.

Table 9.7 Different compression results for the fingerprint in Figure 9.27.

Bits per Pixel	Compression Ratio	q	PSNR
0.75	10.7 : 1	0.0375	38.5462
0.40	20 : 1	0.0271	37.3248
0.25	32 : 1	0.0236	36.7256

Figure 9.35 The fingerprint from Figure 9.27 compressed at 0.40 bit per pixel or at a ratio of 20 : 1.

Figure 9.36 The fingerprint from Figure 9.27 compressed at 0.25 bit per pixel or at a ratio of 32 : 1.

Appendix A

Huffman Coding

The material that follows is adapted from Section 3.4 of Discrete Wavelet Transformations: An Elementary Approach with Applications, *by Patrick J. Van Fleet [60] with permission from John Wiley & Sons, Inc.*

In this appendix we discuss a simple method for reducing the number of bits needed to represent a signal or digital image.

Huffman coding, introduced by David Huffman [37], is an example of lossless compression. The routine can be applied to integer-valued data and the basic idea is quite simple. The method exploits the fact that signals and digital images often contain elements that occur with a much higher frequency than other elements (consider, for example, the digital image and its corresponding histogram in Figures 7.20 and 7.21). Recall that intensity values from grayscale images are integers that range from 0 to 255, and each integer can be represented with an 8-bit ASCII code (see, e. g., Oualline

* *Discrete Wavelet Transformations: An Elementary Approach with Applications.* By D. K. Ruch and P. J. Van Fleet