


9

Scalar Quantization

9.1 Overview

 In this chapter we begin our study of quantization, one of the simplest and most general ideas in lossy compression. We will look at scalar quantization in this chapter and continue with vector quantization in the next chapter. First, the general quantization problem is stated, then various solutions are examined, starting with the simpler solutions, which require the most assumptions, and proceeding to more complex solutions that require fewer assumptions. We describe uniform quantization with fixed-length codewords, first assuming a uniform source, then a source with a known probability density function (*pdf*) that is not necessarily uniform, and finally a source with unknown or changing statistics. We then look at *pdf*-optimized nonuniform quantization, followed by companded quantization. Finally, we return to the more general statement of the quantizer design problem and study entropy-coded quantization.

9.2 Introduction

In many lossy compression applications we are required to represent each source output using one of a small number of codewords. The number of possible distinct source output values is generally much larger than the number of codewords available to represent them. The process of representing a large—possibly infinite—set of values with a much smaller set is called *quantization*.

Consider a source that generates numbers between -10.0 and 10.0 . A simple quantization scheme would be to represent each output of the source with the integer value closest to it. (If the source output is equally close to two integers, we will randomly pick one of them.) For example, if the source output is 2.47 , we would represent it as 2 , and if the source output is 3.1415926 , we would represent it as 3 .

This approach reduces the size of the alphabet required to represent the source output; the infinite number of values between -10.0 and 10.0 are represented with a set that contains only 21 values ($\{-10, \dots, 0, \dots, 10\}$). At the same time we have also forever lost the original value of the source output. If we are told that the reconstruction value is 3, we cannot tell whether the source output was 2.95, 3.16, 3.057932, or any other of an infinite set of values. In other words, we have lost some information. This loss of information is the reason for the use of the word “lossy” in many lossy compression schemes.

The set of inputs and outputs of a quantizer can be scalars or vectors. If they are scalars, we call the quantizers *scalar quantizers*. If they are vectors, we call the quantizers *vector quantizers*. We will study scalar quantizers in this chapter and vector quantizers in Chapter 10.

9.3 The Quantization Problem

Quantization is a very simple process. However, the design of the quantizer has a significant impact on the amount of compression obtained and loss incurred in a lossy compression scheme. Therefore, we will devote a lot of attention to issues related to the design of quantizers.

In practice, the quantizer consists of two mappings: an encoder mapping and a decoder mapping. The encoder divides the range of values that the source generates into a number of intervals. Each interval is represented by a distinct codeword. The encoder represents all the source outputs that fall into a particular interval by the codeword representing that interval. As there could be many—possibly infinitely many—distinct sample values that can fall in any given interval, the encoder mapping is irreversible. Knowing the code only tells us the interval to which the sample value belongs. It does not tell us which of the many values in the interval is the actual sample value. When the sample value comes from an analog source, the encoder is called an analog-to-digital (A/D) converter.

The encoder mapping for a quantizer with eight reconstruction values is shown in Figure 9.1. For this encoder, all samples with values between -1 and 0 would be assigned the code 011. All values between 0 and 1.0 would be assigned the codeword 100, and so on. On the two boundaries, all inputs with values greater than 3 would be assigned the code 111, and all inputs with values less than -3.0 would be assigned the code 000. Thus, any input

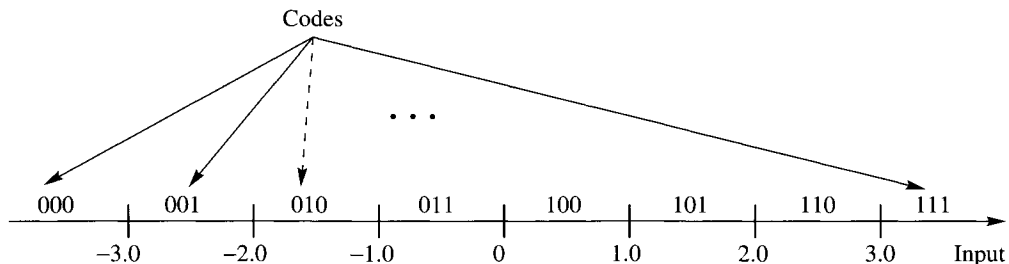


FIGURE 9.1 Mapping for a 3-bit encoder.

Input Codes	Output
000	-3.5
001	-2.5
010	-1.5
011	-0.5
100	0.5
101	1.5
110	2.5
111	3.5

FIGURE 9.2 Mapping for a 3-bit D/A converter.

that we receive will be assigned a codeword depending on the interval in which it falls. As we are using 3 bits to represent each value, we refer to this quantizer as a 3-bit quantizer.

For every codeword generated by the encoder, the decoder generates a reconstruction value. Because a codeword represents an entire interval, and there is no way of knowing which value in the interval was actually generated by the source, the decoder puts out a value that, in some sense, best represents all the values in the interval. Later, we will see how to use information we may have about the distribution of the input in the interval to obtain a representative value. For now, we simply use the midpoint of the interval as the representative value generated by the decoder. If the reconstruction is analog, the decoder is often referred to as a digital-to-analog (D/A) converter. A decoder mapping corresponding to the 3-bit encoder shown in Figure 9.1 is shown in Figure 9.2.

Example 9.3.1:

Suppose a sinusoid $4\cos(2\pi t)$ was sampled every 0.05 second. The sample was digitized using the A/D mapping shown in Figure 9.1 and reconstructed using the D/A mapping shown in Figure 9.2. The first few inputs, codewords, and reconstruction values are given in Table 9.1. Notice the first two samples in Table 9.1. Although the two input values are distinct, they both fall into the same interval in the quantizer. The encoder, therefore, represents both inputs with the same codeword, which in turn leads to identical reconstruction values.

TABLE 9.1 Digitizing a sine wave.

t	$4\cos(2\pi t)$	A/D Output	D/A Output	Error
0.05	3.804	111	3.5	0.304
0.10	3.236	111	3.5	-0.264
0.15	2.351	110	2.5	-0.149
0.20	1.236	101	1.5	-0.264



Construction of the intervals (their location, etc.) can be viewed as part of the design of the encoder. Selection of reconstruction values is part of the design of the decoder. However, the fidelity of the reconstruction depends on both the intervals and the reconstruction values. Therefore, when designing or analyzing encoders and decoders, it is reasonable to view them as a pair. We call this encoder-decoder pair a *quantizer*. The quantizer mapping for the 3-bit encoder-decoder pair shown in Figures 9.1 and 9.2 can be represented by the input-output map shown in Figure 9.3. The quantizer accepts sample values, and depending on the interval in which the sample values fall, it provides an output codeword and a representation value. Using the map of Figure 9.3, we can see that an input to the quantizer of 1.7 will result in an output of 1.5, and an input of -0.3 will result in an output of -0.5 .

From Figures 9.1–9.3 we can see that we need to know how to divide the input range into intervals, assign binary codes to these intervals, and find representation or output values for these intervals in order to specify a quantizer. We need to do all of this while satisfying distortion and rate criteria. In this chapter we will define distortion to be the average squared difference between the quantizer input and output. We call this the mean squared quantization error (msqe) and denote it by σ_q^2 . The rate of the quantizer is the average number of bits

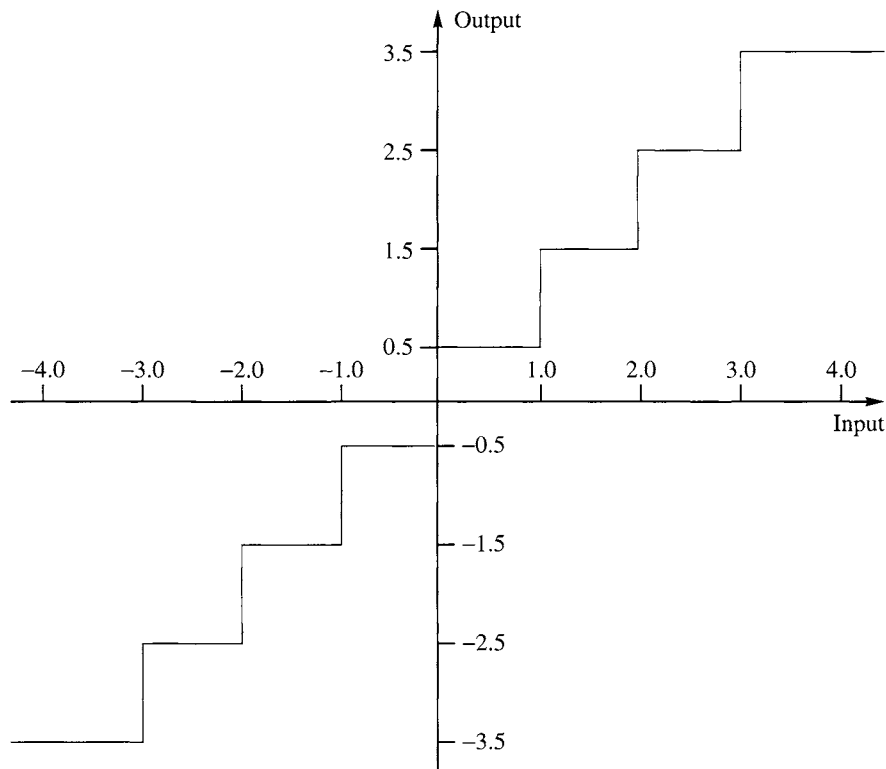


FIGURE 9.3 Quantizer input-output map.

required to represent a single quantizer output. We would like to get the lowest distortion for a given rate, or the lowest rate for a given distortion.

Let us pose the design problem in precise terms. Suppose we have an input modeled by a random variable X with pdf $f_X(x)$. If we wished to quantize this source using a quantizer with M intervals, we would have to specify $M + 1$ endpoints for the intervals, and a representative value for each of the M intervals. The endpoints of the intervals are known as *decision boundaries*, while the representative values are called *reconstruction levels*. We will often model discrete sources with continuous distributions. For example, the difference between neighboring pixels is often modeled using a Laplacian distribution even though the differences can only take on a limited number of discrete values. Discrete processes are modeled with continuous distributions because it can simplify the design process considerably, and the resulting designs perform well in spite of the incorrect assumption. Several of the continuous distributions used to model source outputs are unbounded—that is, the range of values is infinite. In these cases, the first and last endpoints are generally chosen to be $\pm\infty$.

Let us denote the decision boundaries by $\{b_i\}_{i=0}^M$, the reconstruction levels by $\{y_i\}_{i=1}^M$, and the quantization operation by $Q(\cdot)$. Then

$$Q(x) = y_i \quad \text{iff} \quad b_{i-1} < x \leq b_i. \quad (9.1)$$

The mean squared quantization error is then given by

$$\sigma_q^2 = \int_{-\infty}^{\infty} (x - Q(x))^2 f_X(x) dx \quad (9.2)$$

$$= \sum_{i=1}^M \int_{b_{i-1}}^{b_i} (x - y_i)^2 f_X(x) dx. \quad (9.3)$$

The difference between the quantizer input x and output $y = Q(x)$, besides being referred to as the quantization error, is also called the *quantizer distortion* or *quantization noise*. But the word “noise” is somewhat of a misnomer. Generally, when we talk about noise we mean a process external to the source process. Because of the manner in which the quantization error is generated, it is dependent on the source process and, therefore, cannot be regarded as external to the source process. One reason for the use of the word “noise” in this context is that from time to time we will find it useful to model the quantization process as an additive noise process as shown in Figure 9.4.

If we use fixed-length codewords to represent the quantizer output, then the size of the output alphabet immediately specifies the rate. If the number of quantizer outputs is M , then the rate is given by

$$R = \lceil \log_2 M \rceil. \quad (9.4)$$

For example, if $M = 8$, then $R = 3$. In this case, we can pose the quantizer design problem as follows:

Given an input pdf $f_X(x)$ and the number of levels M in the quantizer, find the decision boundaries $\{b_i\}$ and the reconstruction levels $\{y_i\}$ so as to minimize the mean squared quantization error given by Equation (9.3).

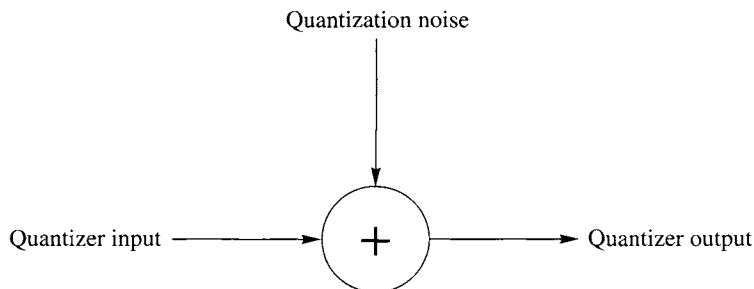


FIGURE 9.4 Additive noise model of a quantizer.

TABLE 9.2 Codeword assignment for an eight-level quantizer.

y_1	1110
y_2	1100
y_3	100
y_4	00
y_5	01
y_6	101
y_7	1101
y_8	1111

However, if we are allowed to use variable-length codes, such as Huffman codes or arithmetic codes, along with the size of the alphabet, the selection of the decision boundaries will also affect the rate of the quantizer. Consider the codeword assignment for the output of an eight-level quantizer shown in Table 9.2.

According to this codeword assignment, if the output y_4 occurs, we use 2 bits to encode it, while if the output y_1 occurs, we need 4 bits to encode it. Obviously, the rate will depend on how often we have to encode y_4 versus how often we have to encode y_1 . In other words, the rate will depend on the probability of occurrence of the outputs. If l_i is the length of the codeword corresponding to the output y_i , and $P(y_i)$ is the probability of occurrence of y_i , then the rate is given by

$$R = \sum_{i=1}^M l_i P(y_i). \quad (9.5)$$

However, the probabilities $\{P(y_i)\}$ depend on the decision boundaries $\{b_i\}$. For example, the probability of y_i occurring is given by

$$P(y_i) = \int_{b_{i-1}}^{b_i} f_X(x) dx.$$

Therefore, the rate R is a function of the decision boundaries and is given by the expression

$$R = \sum_{i=1}^M l_i \int_{b_{i-1}}^{b_i} f_X(x) dx. \quad (9.6)$$

From this discussion and Equations (9.3) and (9.6), we see that for a given source input, the partitions we select and the representation for those partitions will determine the distortion incurred during the quantization process. The partitions we select and the binary codes for the partitions will determine the rate for the quantizer. Thus, the problem of finding the optimum partitions, codes, and representation levels are all linked. In light of this information, we can restate our problem statement:

Given a distortion constraint

$$\sigma_q^2 \leq D^* \quad (9.7)$$

find the decision boundaries, reconstruction levels, and binary codes that minimize the rate given by Equation (9.6), while satisfying Equation (9.7).

Or, given a rate constraint

$$R \leq R^* \quad (9.8)$$

find the decision boundaries, reconstruction levels, and binary codes that minimize the distortion given by Equation (9.3), while satisfying Equation (9.8).

This problem statement of quantizer design, while more general than our initial statement, is substantially more complex. Fortunately, in practice there are situations in which we can simplify the problem. We often use fixed-length codewords to encode the quantizer output. In this case, the rate is simply the number of bits used to encode each output, and we can use our initial statement of the quantizer design problem. We start our study of quantizer design by looking at this simpler version of the problem, and later use what we have learned in this process to attack the more complex version.

9.4 Uniform Quantizer

The simplest type of quantizer is the uniform quantizer. All intervals are the same size in the uniform quantizer, except possibly for the two outer intervals. In other words, the decision boundaries are spaced evenly. The reconstruction values are also spaced evenly, with the same spacing as the decision boundaries; in the inner intervals, they are the midpoints of the intervals. This constant spacing is usually referred to as the step size and is denoted by Δ . The quantizer shown in Figure 9.3 is a uniform quantizer with $\Delta = 1$. It does not have zero as one of its representation levels. Such a quantizer is called a *midrise quantizer*. An alternative uniform quantizer could be the one shown in Figure 9.5. This is called a *midtread quantizer*. As the midtread quantizer has zero as one of its output levels, it is especially useful in situations where it is important that the zero value be represented—for example,

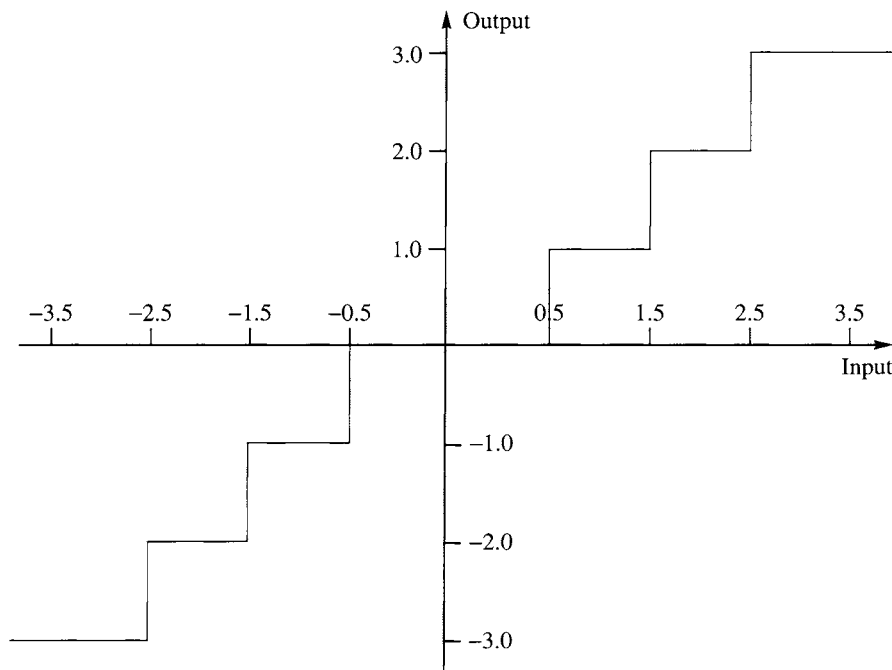


FIGURE 9.5 A midtread quantizer.

control systems in which it is important to represent a zero value accurately, and audio coding schemes in which we need to represent silence periods. Notice that the midtread quantizer has only seven intervals or levels. That means that if we were using a fixed-length 3-bit code, we would have one codeword left over.

Usually, we use a midrise quantizer if the number of levels is even and a midtread quantizer if the number of levels is odd. For the remainder of this chapter, unless we specifically mention otherwise, we will assume that we are dealing with midrise quantizers. We will also generally assume that the input distribution is symmetric around the origin and the quantizer is also symmetric. (The optimal minimum mean squared error quantizer for a symmetric distribution need not be symmetric [106].) Given all these assumptions, the design of a uniform quantizer consists of finding the step size Δ that minimizes the distortion for a given input process and number of decision levels.

Uniform Quantization of a Uniformly Distributed Source

We start our study of quantizer design with the simplest of all cases: design of a uniform quantizer for a uniformly distributed source. Suppose we want to design an M -level uniform quantizer for an input that is uniformly distributed in the interval $[-X_{\max}, X_{\max}]$. This means

we need to divide the $[-X_{\max}, X_{\max}]$ interval into M equally sized intervals. In this case, the step size Δ is given by

$$\Delta = \frac{2X_{\max}}{M}. \quad (9.9)$$

The distortion in this case becomes

$$\sigma_q^2 = 2 \sum_{i=1}^{\frac{M}{2}} \int_{(i-1)\Delta}^{i\Delta} \left(x - \frac{2i-1}{2}\Delta \right)^2 \frac{1}{2X_{\max}} dx. \quad (9.10)$$

If we evaluate this integral (after some suffering), we find that the msqe is $\Delta^2/12$.

The same result can be more easily obtained if we examine the behavior of the quantization error q given by

$$q = x - Q(x). \quad (9.11)$$

In Figure 9.6 we plot the quantization error versus the input signal for an eight-level uniform quantizer, with an input that lies in the interval $[-X_{\max}, X_{\max}]$. Notice that the quantization error lies in the interval $[-\frac{\Delta}{2}, \frac{\Delta}{2}]$. As the input is uniform, it is not difficult to establish that the quantization error is also uniform over this interval. Thus, the mean squared quantization error is the second moment of a random variable uniformly distributed in the interval $[-\frac{\Delta}{2}, \frac{\Delta}{2}]$:

$$\sigma_q^2 = \frac{1}{\Delta} \int_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} q^2 dq \quad (9.12)$$

$$= \frac{\Delta^2}{12}. \quad (9.13)$$

Let us also calculate the signal-to-noise ratio for this case. The signal variance σ_x^2 for a uniform random variable, which takes on values in the interval $[-X_{\max}, X_{\max}]$, is $\frac{(2X_{\max})^2}{12}$.

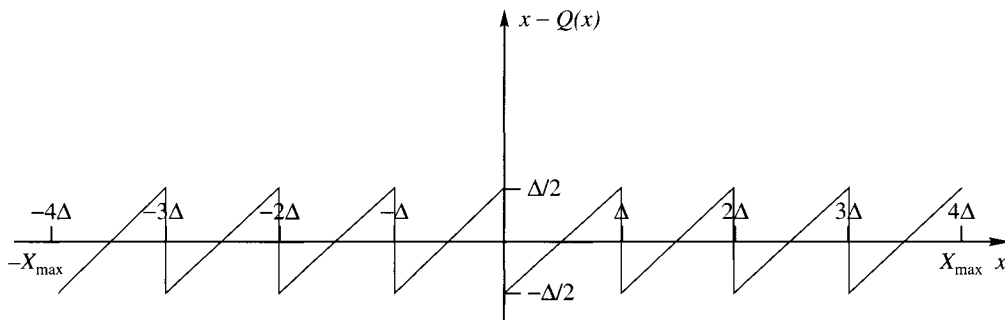


FIGURE 9.6 Quantization error for a uniform midrise quantizer with a uniformly distributed input.

The value of the step size Δ is related to X_{\max} and the number of levels M by

$$\Delta = \frac{2X_{\max}}{M}.$$

For the case where we use a fixed-length code, with each codeword being made up of n bits, the number of codewords or the number of reconstruction levels M is 2^n . Combining all this, we have

$$\text{SNR(dB)} = 10 \log_{10} \left(\frac{\sigma_s^2}{\sigma_q^2} \right) \quad (9.14)$$

$$= 10 \log_{10} \left(\frac{(2X_{\max})^2}{12} \cdot \frac{12}{\Delta^2} \right) \quad (9.15)$$

$$= 10 \log_{10} \left(\frac{(2X_{\max})^2}{12} \frac{12}{\left(\frac{2X_{\max}}{M}\right)^2} \right) \quad (9.16)$$

$$= 10 \log_{10}(M^2)$$

$$= 20 \log_{10}(2^n)$$

$$= 6.02n \text{ dB}. \quad (9.17)$$

This equation says that for every additional bit in the quantizer, we get an increase in the signal-to-noise ratio of 6.02 dB. This is a well-known result and is often used to get an indication of the maximum gain available if we increase the rate. However, remember that we obtained this result under some assumptions about the input. If the assumptions are not true, this result will not hold true either.

Example 9.4.1: Image compression

A probability model for the variations of pixels in an image is almost impossible to obtain because of the great variety of images available. A common approach is to declare the pixel values to be uniformly distributed between 0 and $2^b - 1$, where b is the number of bits per pixel. For most of the images we deal with, the number of bits per pixel is 8; therefore, the pixel values would be assumed to vary uniformly between 0 and 255. Let us quantize our test image Sena using a uniform quantizer.

If we wanted to use only 1 bit per pixel, we would divide the range $[0, 255]$ into two intervals, $[0, 127]$ and $[128, 255]$. The first interval would be represented by the value 64, the midpoint of the first interval; the pixels in the second interval would be represented by the pixel value 196, the midpoint of the second interval. In other words, the boundary values are $\{0, 128, 255\}$, while the reconstruction values are $\{64, 196\}$. The quantized image is shown in Figure 9.7. As expected, almost all the details in the image have disappeared. If we were to use a 2-bit quantizer, with boundary values $\{0, 64, 128, 196, 255\}$ and reconstruction levels $\{32, 96, 160, 224\}$, we get considerably more detail. The level of detail increases as the use of bits increases until at 6 bits per pixel, the reconstructed image is indistinguishable from the original, at least to a casual observer. The 1-, 2-, and 3-bit images are shown in Figure 9.7.

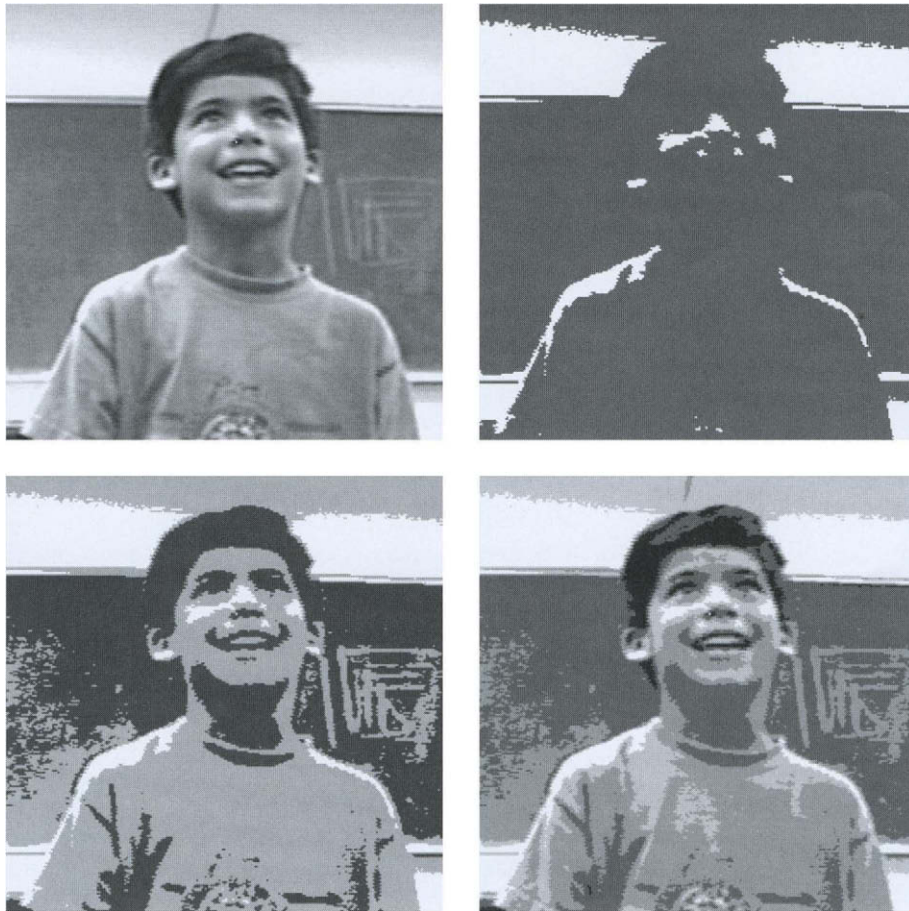


FIGURE 9.7 Top left: original Sena image; top right: 1 bit/pixel image; bottom left: 2 bits/pixel; bottom right: 3 bits/pixel.

Looking at the lower-rate images, we notice a couple of things. First, the lower-rate images are darker than the original, and the lowest-rate reconstructions are the darkest. The reason for this is that the quantization process usually results in scaling down of the dynamic range of the input. For example, in the 1-bit-per-pixel reproduction, the highest pixel value is 196, as opposed to 255 for the original image. As higher gray values represent lighter shades, there is a corresponding darkening of the reconstruction. The other thing to notice in the low-rate reconstruction is that wherever there were smooth changes in gray values there are now abrupt transitions. This is especially evident in the face and neck area, where gradual shading has been transformed to blotchy regions of constant values. This is because a range of values is being mapped to the same value, as was the case for the first two samples of the sinusoid in Example 9.3.1. For obvious reasons, this effect is called *contouring*. The perceptual effect of contouring can be reduced by a procedure called *dithering* [107]. ♦

Uniform Quantization of Nonuniform Sources

Quite often the sources we deal with do not have a uniform distribution; however, we still want the simplicity of a uniform quantizer. In these cases, even if the sources are bounded, simply dividing the range of the input by the number of quantization levels does not produce a very good design.

Example 9.4.2:

Suppose our input fell within the interval $[-1, 1]$ with probability 0.95, and fell in the intervals $[-100, 1)$, $(1, 100]$ with probability 0.05. Suppose we wanted to design an eight-level uniform quantizer. If we followed the procedure of the previous section, the step size would be 25. This means that inputs in the $[-1, 0)$ interval would be represented by the value -12.5 , and inputs in the interval $[0, 1)$ would be represented by the value 12.5. The maximum quantization error that can be incurred is 12.5. However, at least 95% of the time, the *minimum* error that will be incurred is 11.5. Obviously, this is not a very good design. A much better approach would be to use a smaller step size, which would result in better representation of the values in the $[-1, 1]$ interval, even if it meant a larger maximum error. Suppose we pick a step size of 0.3. In this case, the maximum quantization error goes from 12.5 to 98.95. However, 95% of the time the quantization error will be less than 0.15. Therefore, the average distortion, or msqe, for this quantizer would be substantially less than the msqe for the first quantizer. ♦

We can see that when the distribution is no longer uniform, it is not a good idea to obtain the step size by simply dividing the range of the input by the number of levels. This approach becomes totally impractical when we model our sources with distributions that are unbounded, such as the Gaussian distribution. Therefore, we include the *pdf* of the source in the design process.

Our objective is to find the step size that, for a given value of M , will minimize the distortion. The simplest way to do this is to write the distortion as a function of the step size, and then minimize this function. An expression for the distortion, or msqe, for an M -level uniform quantizer as a function of the step size can be found by replacing the b_i s and y_i s in Equation (9.3) with functions of Δ . As we are dealing with a symmetric condition, we need only compute the distortion for positive values of x ; the distortion for negative values of x will be the same.

From Figure 9.8, we see that the decision boundaries are integral multiples of Δ , and the representation level for the interval $[(k-1)\Delta, k\Delta]$ is simply $\frac{2k-1}{2}\Delta$. Therefore, the expression for msqe becomes

$$\begin{aligned}\sigma_q^2 &= 2 \sum_{i=1}^{\frac{M}{2}-1} \int_{(i-1)\Delta}^{i\Delta} \left(x - \frac{2i-1}{2}\Delta\right)^2 f_X(x) dx \\ &\quad + 2 \int_{(\frac{M}{2}-1)\Delta}^{\infty} \left(x - \frac{M-1}{2}\Delta\right)^2 f_X(x) dx.\end{aligned}\tag{9.18}$$

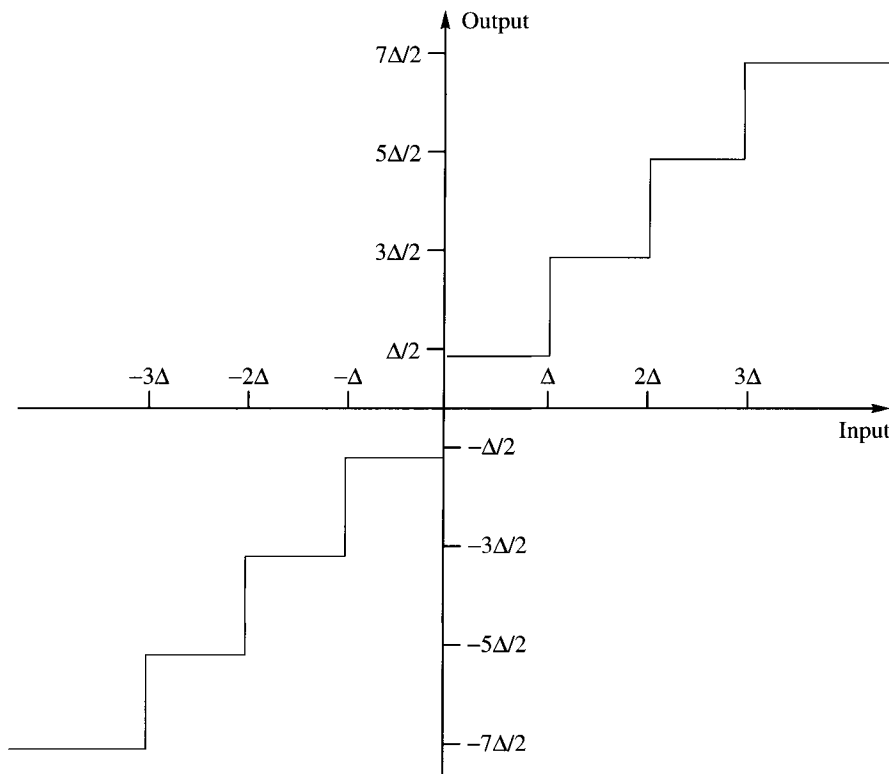


FIGURE 9.8 A uniform midrise quantizer.

To find the optimal value of Δ , we simply take a derivative of this equation and set it equal to zero [108] (see Problem 1).

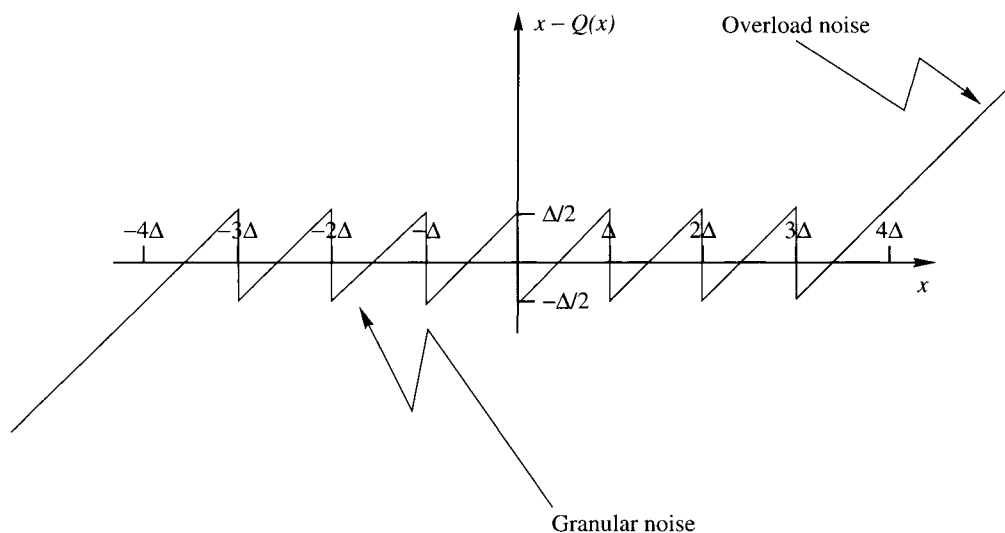
$$\begin{aligned} \frac{\delta \sigma_q^2}{\delta \Delta} &= - \sum_{i=1}^{\frac{M}{2}-1} (2i-1) \int_{(i-1)\Delta}^{i\Delta} \left(x - \frac{2i-1}{2}\Delta\right) f_X(x) dx \\ &\quad - (M-1) \int_{(\frac{M}{2}-1)\Delta}^{\infty} \left(x - \frac{M-1}{2}\Delta\right) f_X(x) dx = 0. \end{aligned} \quad (9.19)$$

This is a rather messy-looking expression, but given the *pdf* $f_X(x)$, it is easy to solve using any one of a number of numerical techniques (see Problem 2). In Table 9.3, we list step sizes found by solving (9.19) for nine different alphabet sizes and three different distributions.

Before we discuss the results in Table 9.3, let's take a look at the quantization noise for the case of nonuniform sources. Nonuniform sources are often modeled by *pdfs* with unbounded support. That is, there is a nonzero probability of getting an unbounded input. In practical situations, we are not going to get inputs that are unbounded, but often it is very convenient to model the source process with an unbounded distribution. The classic example of this is measurement error, which is often modeled as having a Gaussian distribution,

TABLE 9.3 Optimum step size and SNR for uniform quantizers for different distributions and alphabet sizes [108, 109].

Alphabet Size	Uniform		Gaussian		Laplacian	
	Step Size	SNR	Step Size	SNR	Step Size	SNR
2	1.732	6.02	1.596	4.40	1.414	3.00
4	0.866	12.04	0.9957	9.24	1.0873	7.05
6	0.577	15.58	0.7334	12.18	0.8707	9.56
8	0.433	18.06	0.5860	14.27	0.7309	11.39
10	0.346	20.02	0.4908	15.90	0.6334	12.81
12	0.289	21.60	0.4238	17.25	0.5613	13.98
14	0.247	22.94	0.3739	18.37	0.5055	14.98
16	0.217	24.08	0.3352	19.36	0.4609	15.84
32	0.108	30.10	0.1881	24.56	0.2799	20.46

**FIGURE 9.9** Quantization error for a uniform midrise quantizer.

even when the measurement error is known to be bounded. If the input is unbounded, the quantization error is no longer bounded either. The quantization error as a function of input is shown in Figure 9.9. We can see that in the inner intervals the error is still bounded by $\frac{\Delta}{2}$; however, the quantization error in the outer intervals is unbounded. These two types of quantization errors are given different names. The bounded error is called *granular error* or *granular noise*, while the unbounded error is called *overload error* or *overload noise*. In the expression for the msqe in Equation (9.18), the first term represents the granular noise, while the second term represents the overload noise. The probability that the input will fall into the overload region is called the *overload probability* (Figure 9.10).

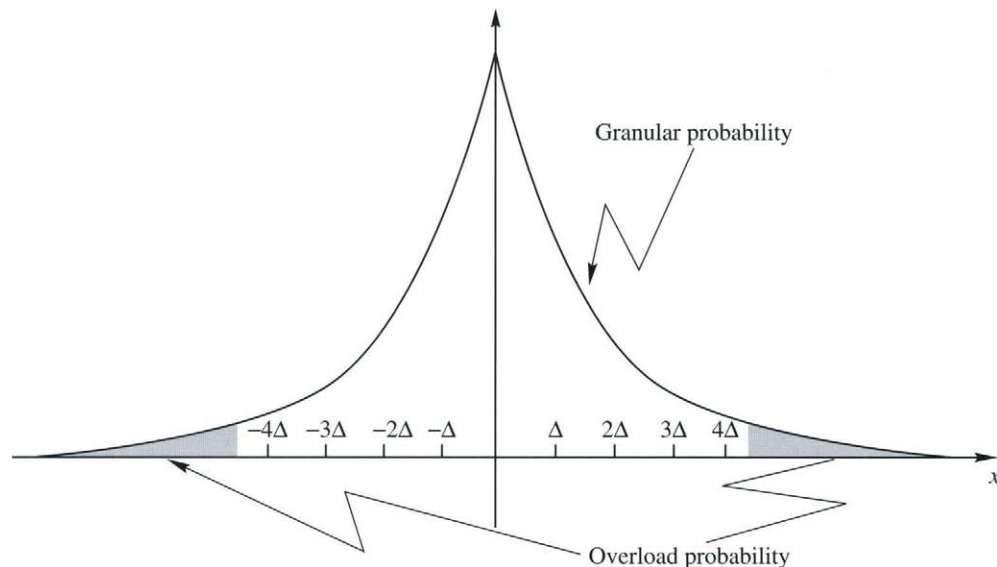


FIGURE 9.10 Overload and granular regions for a 3-bit uniform quantizer.

The nonuniform sources we deal with have probability density functions that are generally peaked at zero and decay as we move away from the origin. Therefore, the overload probability is generally much smaller than the probability of the input falling in the granular region. As we see from Equation (9.19), an increase in the size of the step size Δ will result in an increase in the value of $(\frac{M}{2} - 1)\Delta$, which in turn will result in a decrease in the overload probability and the second term in Equation (9.19). However, an increase in the step size Δ will also increase the granular noise, which is the first term in Equation (9.19). The design process for the uniform quantizer is a balancing of these two effects. An important parameter that describes this trade-off is the loading factor f_l , defined as the ratio of the maximum value the input can take in the granular region to the standard deviation. A common value of the loading factor is 4. This is also referred to as *4 σ loading*.

Recall that when quantizing an input with a uniform distribution, the SNR and bit rate are related by Equation (9.17), which says that for each bit increase in the rate there is an increase of 6.02 dB in the SNR. In Table 9.3, along with the step sizes, we have also listed the SNR obtained when a million input values with the appropriate *pdf* are quantized using the indicated quantizer.

From this table, we can see that, although the SNR for the uniform distribution follows the rule of a 6.02 dB increase in the signal-to-noise ratio for each additional bit, this is not true for the other distributions. Remember that we made some assumptions when we obtained the 6.02 n rule that are only valid for the uniform distribution. Notice that the more peaked a distribution is (that is, the further away from uniform it is), the more it seems to vary from the 6.02 dB rule.

We also said that the selection of Δ is a balance between the overload and granular errors. The Laplacian distribution has more of its probability mass away from the origin in

its tails than the Gaussian distribution. This means that for the same step size and number of levels there is a higher probability of being in the overload region if the input has a Laplacian distribution than if the input has a Gaussian distribution. The uniform distribution is the extreme case, where the overload probability is zero. For the same number of levels, if we increase the step size, the size of the overload region (and hence the overload probability) is reduced at the expense of granular noise. Therefore, for a given number of levels, if we were picking the step size to balance the effects of the granular and overload noise, distributions that have heavier tails will tend to have larger step sizes. This effect can be seen in Table 9.3. For example, for eight levels the step size for the uniform quantizer is 0.433. The step size for the Gaussian quantizer is larger (0.586), while the step size for the Laplacian quantizer is larger still (0.7309).

Mismatch Effects

We have seen that for a result to hold, the assumptions we used to obtain the result have to hold. When we obtain the optimum step size for a particular uniform quantizer using Equation (9.19), we make some assumptions about the statistics of the source. We assume a certain distribution and certain parameters of the distribution. What happens when our assumptions do not hold? Let's try to answer this question empirically.

We will look at two types of mismatches. The first is when the assumed distribution type matches the actual distribution type, but the variance of the input is different from the assumed variance. The second mismatch is when the actual distribution type is different from the distribution type assumed when obtaining the value of the step size. Throughout our discussion, we will assume that the mean of the input distribution is zero.

In Figure 9.11, we have plotted the signal-to-noise ratio as a function of the ratio of the actual to assumed variance of a 4-bit Gaussian uniform quantizer, with a Gaussian

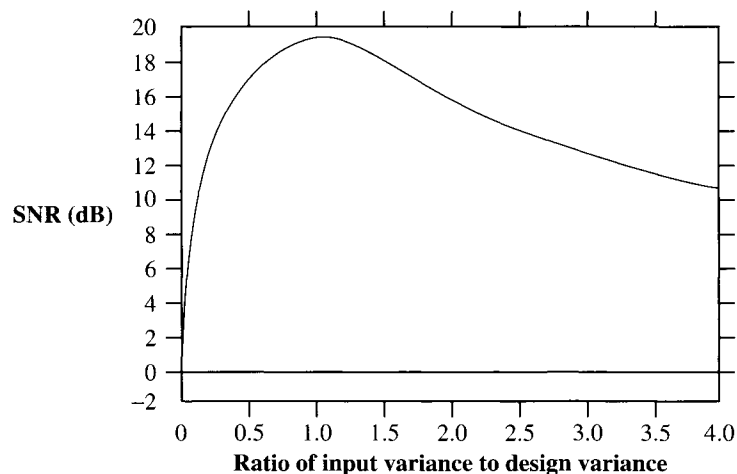


FIGURE 9.11 Effect of variance mismatch on the performance of a 4-bit uniform quantizer.

input. (To see the effect under different conditions, see Problem 5.) Remember that for a distribution with zero mean, the variance is given by $\sigma_x^2 = E[X^2]$, which is also a measure of the power in the signal X . As we can see from the figure, the signal-to-noise ratio is maximum when the input signal variance matches the variance assumed when designing the quantizer. From the plot we also see that there is an asymmetry; the SNR is considerably worse when the input variance is lower than the assumed variance. This is because the SNR is a ratio of the input variance and the mean squared quantization error. When the input variance is smaller than the assumed variance, the mean squared quantization error actually drops because there is less overload noise. However, because the input variance is low, the ratio is small. When the input variance is higher than the assumed variance, the msqe increases substantially, but because the input power is also increasing, the ratio does not decrease as dramatically. To see this more clearly, we have plotted the mean squared error versus the signal variance separately in Figure 9.12. We can see from these figures that the decrease in signal-to-noise ratio does not always correlate directly with an increase in msqe.

The second kind of mismatch is where the input distribution does not match the distribution assumed when designing the quantizer. In Table 9.4 we have listed the SNR when inputs with different distributions are quantized using several different eight-level quantizers. The quantizers were designed assuming a particular input distribution.

Notice that as we go from left to right in the table, the designed step size becomes progressively larger than the “correct” step size. This is similar to the situation where the input variance is smaller than the assumed variance. As we can see when we have a mismatch that results in a smaller step size relative to the optimum step size, there is a greater drop in performance than when the quantizer step size is larger than its optimum value.

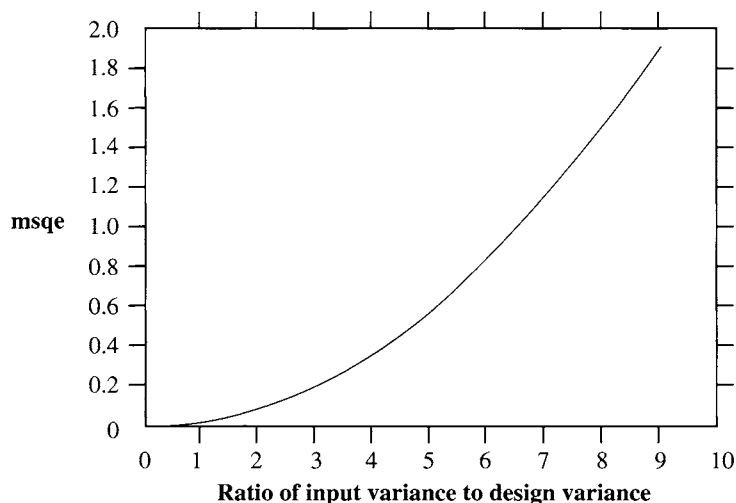


FIGURE 9.12 The msqe as a function of variance mismatch with a 4-bit uniform quantizer.

TABLE 9.4 Demonstration of the effect of mismatch using eight-level quantizers (dB).

Input Distribution	Uniform Quantizer	Gaussian Quantizer	Laplacian Quantizer	Gamma Quantizer
Uniform	18.06	15.56	13.29	12.41
Gaussian	12.40	14.27	13.37	12.73
Laplacian	8.80	10.79	11.39	11.28
Gamma	6.98	8.06	8.64	8.76

9.5 Adaptive Quantization

One way to deal with the mismatch problem is to adapt the quantizer to the statistics of the input. Several things might change in the input relative to the assumed statistics, including the mean, the variance, and the *pdf*. The strategy for handling each of these variations can be different, though certainly not exclusive. If more than one aspect of the input statistics changes, it is possible to combine the strategies for handling each case separately. If the mean of the input is changing with time, the best strategy is to use some form of differential encoding (discussed in some detail in Chapter 11). For changes in the other statistics, the common approach is to adapt the quantizer parameters to the input statistics.

There are two main approaches to adapting the quantizer parameters: an *off-line* or *forward adaptive* approach, and an *on-line* or *backward adaptive* approach. In forward adaptive quantization, the source output is divided into blocks of data. Each block is analyzed before quantization, and the quantizer parameters are set accordingly. The settings of the quantizer are then transmitted to the receiver as *side information*. In backward adaptive quantization, the adaptation is performed based on the quantizer output. As this is available to both transmitter and receiver, there is no need for side information.

9.5.1 Forward Adaptive Quantization

Let us first look at approaches for adapting to changes in input variance using the forward adaptive approach. This approach necessitates a delay of at least the amount of time required to process a block of data. The insertion of side information in the transmitted data stream may also require the resolution of some synchronization problems. The size of the block of data processed also affects a number of other things. If the size of the block is too large, then the adaptation process may not capture the changes taking place in the input statistics. Furthermore, large block sizes mean more delay, which may not be tolerable in certain applications. On the other hand, small block sizes mean that the side information has to be transmitted more often, which in turn means the amount of overhead per sample increases. The selection of the block size is a trade-off between the increase in side information necessitated by small block sizes and the loss of fidelity due to large block sizes (see Problem 7).

The variance estimation procedure is rather simple. At time n we use a block of N future samples to compute an estimate of the variance

$$\hat{\sigma}_q^2 = \frac{1}{N} \sum_{i=0}^{N-1} x_{n+i}^2. \quad (9.20)$$

Note that we are assuming that our input has a mean of zero. The variance information also needs to be quantized so that it can be transmitted to the receiver. Usually, the number of bits used to quantize the value of the variance is significantly larger than the number of bits used to quantize the sample values.

Example 9.5.1:

In Figure 9.13 we show a segment of speech quantized using a fixed 3-bit quantizer. The step size of the quantizer was adjusted based on the statistics of the entire sequence. The sequence was the `testm.raw` sequence from the sample data sets, consisting of about 4000 samples of a male speaker saying the word “test.” The speech signal was sampled at 8000 samples per second and digitized using a 16-bit A/D.

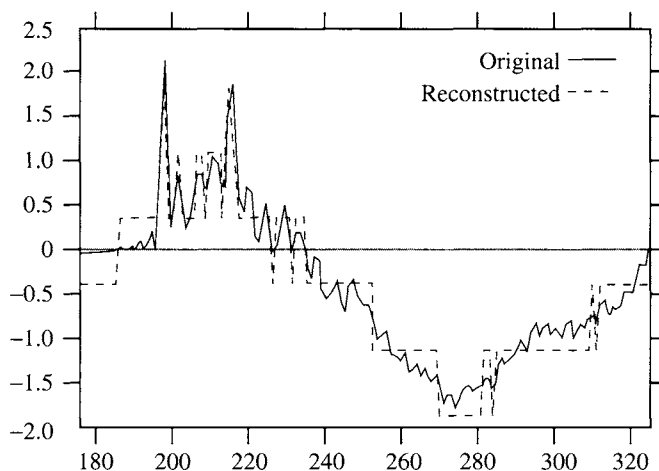


FIGURE 9.13 Original 16-bit speech and compressed 3-bit speech sequences.

We can see from the figure that, as in the case of the example of the sinusoid earlier in this chapter, there is a considerable loss in amplitude resolution. Sample values that are close together have been quantized to the same value.

The same sequence quantized with a forward adaptive quantizer is shown in Figure 9.14. For this example, we divided the input into blocks of 128 samples. Before quantizing the samples in a block, the standard deviation for the samples in the block was obtained. This value was quantized using an 8-bit quantizer and sent to both the transmitter and receiver.

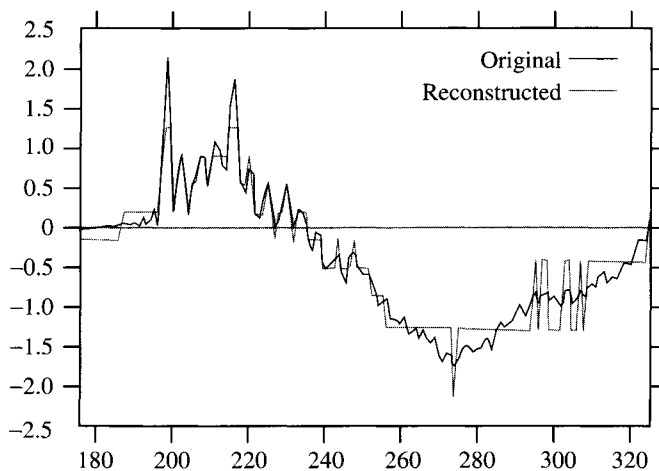


FIGURE 9.14 Original 16-bit speech sequence and sequence obtained using an eight-level forward adaptive quantizer.

The samples in the block were then normalized using this value of the standard deviation. Notice that the reconstruction follows the input much more closely, though there seems to be room for improvement, especially in the latter half of the displayed samples. ♦

Example 9.5.2:

In Example 9.4.1, we used a uniform quantizer with the assumption that the input is uniformly distributed. Let us refine this source model a bit and say that while the source is uniformly distributed over different regions, the range of the input changes. In a forward adaptive quantization scheme, we would obtain the minimum and maximum values for each block of data, which would be transmitted as side information. In Figure 9.15, we see the Sena image quantized with a block size of 8×8 using 3-bit forward adaptive uniform quantization. The side information consists of the minimum and maximum values in each block, which require 8 bits each. Therefore, the overhead in this case is $\frac{16}{8 \times 8}$ or 0.25 bits per pixel, which is quite small compared to the number of bits per sample used by the quantizer.

The resulting image is hardly distinguishable from the original. Certainly at higher rates, forward adaptive quantization seems to be a very good alternative. ♦

9.5.2 Backward Adaptive Quantization

In backward adaptive quantization, only the past quantized samples are available for use in adapting the quantizer. The values of the input are only known to the encoder; therefore, this information cannot be used to adapt the quantizer. How can we get information about mismatch simply by examining the output of the quantizer without knowing what the input was? If we studied the output of the quantizer for a long period of time, we could get some idea about mismatch from the distribution of output values. If the quantizer step size Δ is

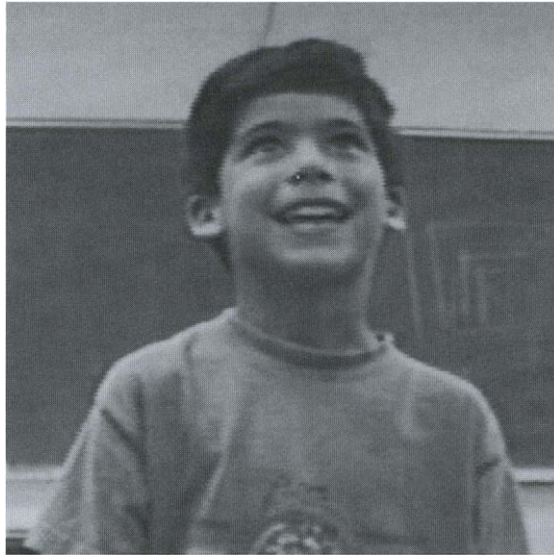


FIGURE 9.15 Sena image quantized to 3.25 bits per pixel using forward adaptive quantization.

well matched to the input, the probability that an input to the quantizer would land in a particular interval would be consistent with the *pdf* assumed for the input. However, if the actual *pdf* differs from the assumed *pdf*, the number of times the input falls in the different quantization intervals will be inconsistent with the assumed *pdf*. If Δ is smaller than what it should be, the input will fall in the outer levels of the quantizer an excessive number of times. On the other hand, if Δ is larger than it should be for a particular source, the input will fall in the inner levels an excessive number of times. Therefore, it seems that we should observe the output of the quantizer for a long period of time, then expand the quantizer step size if the input falls in the outer levels an excessive number of times, and contract the step size if the input falls in the inner levels an excessive number of times.

Nuggehally S. Jayant at Bell Labs showed that we did not need to observe the quantizer output over a long period of time [110]. In fact, we could adjust the quantizer step size after observing a single output. Jayant named this quantization approach “quantization with one word memory.” The quantizer is better known as the *Jayant quantizer*. The idea behind the Jayant quantizer is very simple. If the input falls in the outer levels, the step size needs to be expanded, and if the input falls in the inner quantizer levels, the step size needs to be reduced. The expansions and contractions should be done in such a way that once the quantizer is matched to the input, the product of the expansions and contractions is unity.

The expansion and contraction of the step size is accomplished in the Jayant quantizer by assigning a *multiplier* M_k to each interval. If the $(n - 1)$ th input falls in the k th interval, the step size to be used for the n th input is obtained by multiplying the step size used for the $(n - 1)$ th input with M_k . The multiplier values for the inner levels in the quantizer are less than one, and the multiplier values for the outer levels of the quantizer are greater than one.

Therefore, if an input falls into the inner levels, the quantizer used to quantize the next input will have a smaller step size. Similarly, if an input falls into the outer levels, the step size will be multiplied with a value greater than one, and the next input will be quantized using a larger step size. Notice that the step size for the current input is modified based on the previous quantizer output. The previous quantizer output is available to both the transmitter and receiver, so there is no need to send any additional information to inform the receiver about the adaptation. Mathematically, the adaptation process can be represented as

$$\Delta_n = M_{l(n-1)} \Delta_{n-1} \quad (9.21)$$

where $l(n-1)$ is the quantization interval at time $n-1$.

In Figure 9.16 we show a 3-bit uniform quantizer. We have eight intervals represented by the different quantizer outputs. However, the multipliers for symmetric intervals are identical because of symmetry:

$$M_0 = M_4 \quad M_1 = M_5 \quad M_2 = M_6 \quad M_3 = M_7$$

Therefore, we only need four multipliers. To see how the adaptation proceeds, let us work through a simple example using this quantizer.

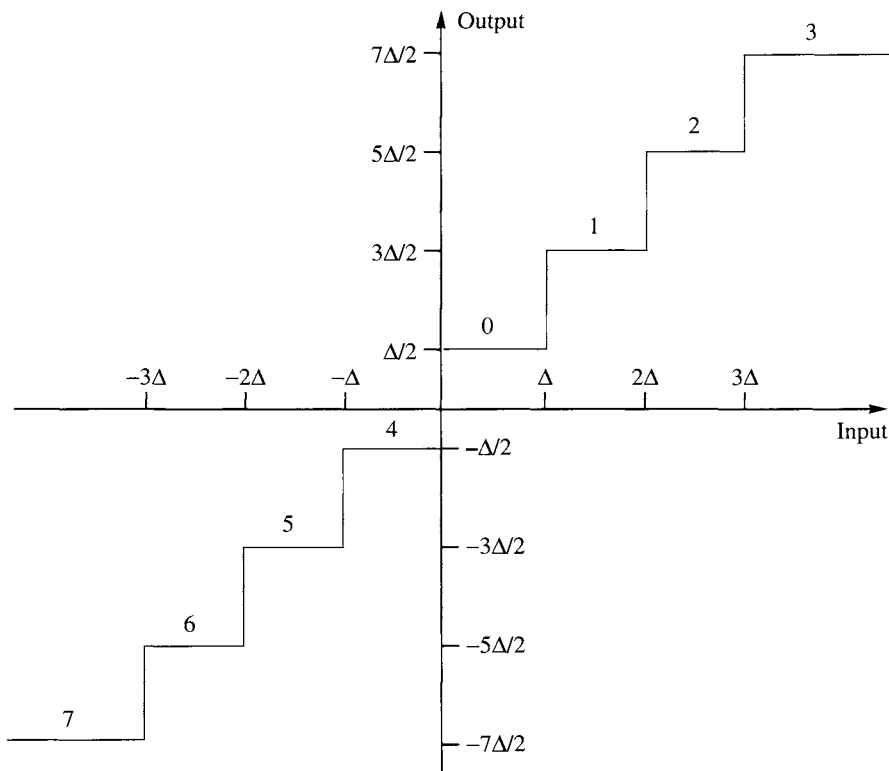


FIGURE 9.16 Output levels for the Jayant quantizer.

Example 9.5.3: Jayant quantizer

For the quantizer in Figure 9.16, suppose the multiplier values are $M_0 = M_4 = 0.8$, $M_1 = M_5 = 0.9$, $M_2 = M_6 = 1$, $M_3 = M_7 = 1.2$; the initial value of the step size, Δ_0 , is 0.5; and the sequence to be quantized is 0.1, -0.2 , 0.2, 0.1, -0.3 , 0.1, 0.2, 0.5, 0.9, 1.5, \dots . When the first input is received, the quantizer step size is 0.5. Therefore, the input falls into level 0, and the output value is 0.25, resulting in an error of 0.15. As this input fell into the quantizer level 0, the new step size Δ_1 is $M_0 \times \Delta_0 = 0.8 \times 0.5 = 0.4$. The next input is -0.2 , which falls into level 4. As the step size at this time is 0.4, the output is -0.2 . To update, we multiply the current step size with M_4 . Continuing in this fashion, we get the sequence of step sizes and outputs shown in Table 9.5.

TABLE 9.5 Operation of a Jayant quantizer.

n	Δ_n	Input	Output Level	Output	Error	Update Equation
0	0.5	0.1	0	0.25	0.15	$\Delta_1 = M_0 \times \Delta_0$
1	0.4	-0.2	4	-0.2	0.0	$\Delta_2 = M_4 \times \Delta_1$
2	0.32	0.2	0	0.16	0.04	$\Delta_3 = M_0 \times \Delta_2$
3	0.256	0.1	0	0.128	0.028	$\Delta_4 = M_0 \times \Delta_3$
4	0.2048	-0.3	5	-0.3072	-0.0072	$\Delta_5 = M_5 \times \Delta_4$
5	0.1843	0.1	0	0.0922	-0.0078	$\Delta_6 = M_0 \times \Delta_5$
6	0.1475	0.2	1	0.2212	0.0212	$\Delta_7 = M_1 \times \Delta_6$
7	0.1328	0.5	3	0.4646	-0.0354	$\Delta_8 = M_3 \times \Delta_7$
8	0.1594	0.9	3	0.5578	-0.3422	$\Delta_9 = M_3 \times \Delta_8$
9	0.1913	1.5	3	0.6696	-0.8304	$\Delta_{10} = M_3 \times \Delta_9$
10	0.2296	1.0	3	0.8036	0.1964	$\Delta_{11} = M_3 \times \Delta_{10}$
11	0.2755	0.9	3	0.9643	0.0643	$\Delta_{12} = M_3 \times \Delta_{11}$

Notice how the quantizer adapts to the input. In the beginning of the sequence, the input values are mostly small, and the quantizer step size becomes progressively smaller, providing better and better estimates of the input. At the end of the sample sequence, the input values are large and the step size becomes progressively bigger. However, the size of the error is quite large during the transition. This means that if the input was changing rapidly, which would happen if we had a high-frequency input, such transition situations would be much more likely to occur, and the quantizer would not function very well. However, in cases where the statistics of the input change slowly, the quantizer could adapt to the input. As most natural sources such as speech and images tend to be correlated, their values do not change drastically from sample to sample. Even when some of this structure is removed through some transformation, the residual structure is generally enough for the Jayant quantizer (or some variation of it) to function quite effectively. ♦

The step size in the initial part of the sequence in this example is progressively getting smaller. We can easily conceive of situations where the input values would be small for a long period. Such a situation could occur during a silence period in speech-encoding systems,

or while encoding a dark background in image-encoding systems. If the step size continues to shrink for an extended period of time, in a finite precision system it would result in a value of zero. This would be catastrophic, effectively replacing the quantizer with a zero output device. Usually, a minimum value Δ_{\min} is defined, and the step size is not allowed to go below this value to prevent this from happening. Similarly, if we get a sequence of large values, the step size could increase to a point that, when we started getting smaller values, the quantizer would not be able to adapt fast enough. To prevent this from happening, a maximum value Δ_{\max} is defined, and the step size is not allowed to increase beyond this value.

The adaptivity of the Jayant quantizer depends on the values of the multipliers. The further the multiplier values are from unity, the more adaptive the quantizer. However, if the adaptation algorithm reacts too fast, this could lead to instability. So how do we go about selecting the multipliers?

First of all, we know that the multipliers corresponding to the inner levels are less than one, and the multipliers for the outer levels are greater than one. If the input process is stationary and P_k represents the probability of being in quantizer interval k (generally estimated by using a fixed quantizer for the input data), then we can impose a stability criterion for the Jayant quantizer based on our requirement that once the quantizer is matched to the input, the product of the expansions and contractions are equal to unity. That is, if n_k is the number of times the input falls in the k th interval,

$$\prod_{k=0}^M M_k^{n_k} = 1. \quad (9.22)$$

Taking the N th root of both sides (where N is the total number of inputs) we obtain

$$\prod_{k=0}^M M_k^{\frac{n_k}{N}} = 1,$$

or

$$\prod_{k=0}^M M_k^{P_k} = 1 \quad (9.23)$$

where we have assumed that $P_k = n_k/N$.

There are an infinite number of multiplier values that would satisfy Equation (9.23). One way to restrict this number is to impose some structure on the multipliers by requiring them to be of the form

$$M_k = \gamma^{l_k} \quad (9.24)$$

where γ is a number greater than one and l_k takes on only integer values [111, 112]. If we substitute this expression for M_k into Equation (9.23), we get

$$\prod_{k=0}^M \gamma^{l_k P_k} = 1, \quad (9.25)$$

which implies that

$$\sum_{k=0}^M l_k P_k = 0. \quad (9.26)$$

The final step is the selection of γ , which involves a significant amount of creativity. The value we pick for γ determines how fast the quantizer will respond to changing statistics. A large value of γ will result in faster adaptation, while a smaller value of γ will result in greater stability.

Example 9.5.4:

Suppose we have to obtain the multiplier functions for a 2-bit quantizer with input probabilities $P_0 = 0.8$, $P_1 = 0.2$. First, note that the multiplier value for the inner level has to be less than 1. Therefore, l_0 is less than 0. If we pick $l_0 = -1$ and $l_1 = 4$, this would satisfy Equation (9.26), while making M_0 less than 1 and M_1 greater than 1. Finally, we need to pick a value for γ .

In Figure 9.17 we see the effect of using different values of γ in a rather extreme example. The input is a square wave that switches between 0 and 1 every 30 samples. The input is quantized using a 2-bit Jayant quantizer. We have used $l_0 = -1$ and $l_1 = 2$. Notice what happens when the input switches from 0 to 1. At first the input falls in the outer level of the quantizer, and the step size increases. This process continues until Δ is just greater than 1. If γ is close to 1, Δ has been increasing quite slowly and should have a value close to 1 right before its value increases to greater than 1. Therefore, the output at this point is close to 1.5. When Δ becomes greater than 1, the input falls in the inner level, and if γ is close to 1, the output suddenly drops to about 0.5. The step size now decreases until it is just

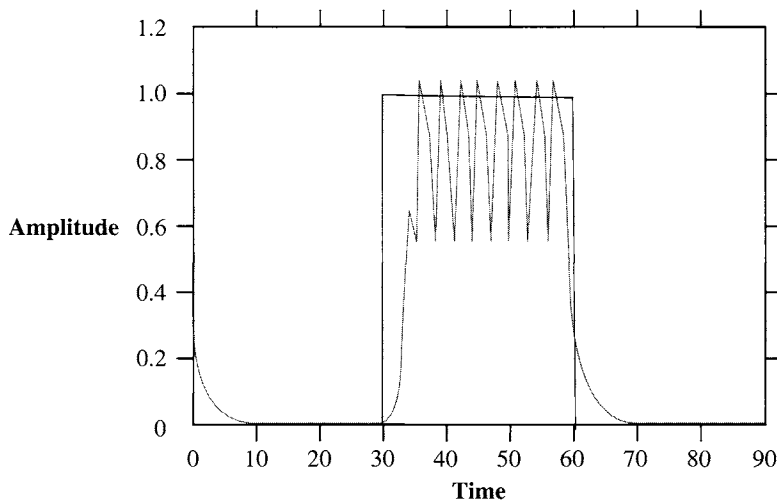


FIGURE 9.17 Effect of γ on the performance of the Jayant quantizer.

below 1, and the process repeats, causing the “ringing” seen in Figure 9.17. As γ increases, the quantizer adapts more rapidly, and the magnitude of the ringing effect decreases. The reason for the decrease is that right before the value of Δ increases above 1, its value is much smaller than 1, and subsequently the output value is much smaller than 1.5. When Δ increases beyond 1, it may increase by a significant amount, so the inner level may be much greater than 0.5. These two effects together compress the ringing phenomenon. Looking at this phenomenon, we can see that it may have been better to have two adaptive strategies, one for when the input is changing rapidly, as in the case of the transitions between 0 and 1, and one for when the input is constant, or nearly so. We will explore this approach further when we describe the quantizer used in the CCITT standard G.726. ♦

When selecting multipliers for a Jayant quantizer, the best quantizers expand more rapidly than they contract. This makes sense when we consider that, when the input falls into the outer levels of the quantizer, it is incurring overload error, which is essentially unbounded. This situation needs to be mitigated with dispatch. On the other hand, when the input falls in the inner levels, the noise incurred is granular noise, which is bounded and therefore may be more tolerable. Finally, the discussion of the Jayant quantizer was motivated by the need for robustness in the face of changing input statistics. Let us repeat the earlier experiment with changing input variance and distributions and see the performance of the Jayant quantizer compared to the *pdf*-optimized quantizer. The results for these experiments are presented in Figure 9.18.

Notice how flat the performance curve is. While the performance of the Jayant quantizer is much better than the nonadaptive uniform quantizer over a wide range of input variances, at the point where the input variance and design variance agree, the performance of the nonadaptive quantizer is significantly better than the performance of the Jayant quantizer.

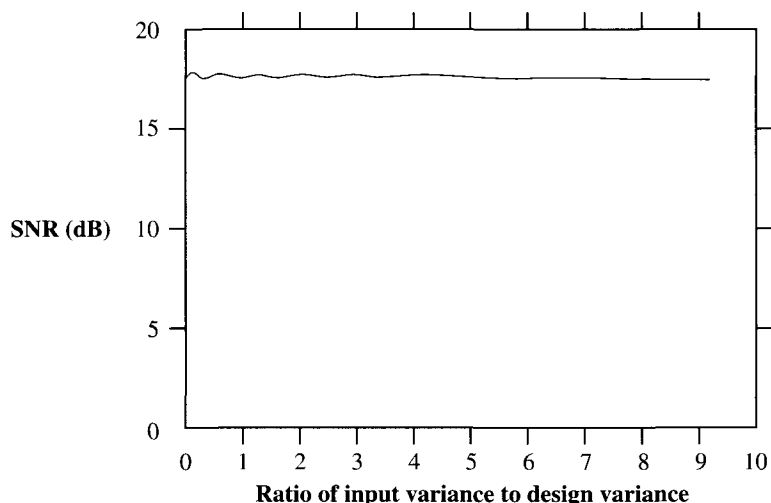


FIGURE 9. 18 Performance of the Jayant quantizer for different input variances.

This means that if we know the input statistics and we are reasonably certain that the input statistics will not change over time, it is better to design for those statistics than to design an adaptive system.

9.6 Nonuniform Quantization

As we can see from Figure 9.10, if the input distribution has more mass near the origin, the input is more likely to fall in the inner levels of the quantizer. Recall that in lossless compression, in order to minimize the *average* number of bits per input symbol, we assigned shorter codewords to symbols that occurred with higher probability and longer codewords to symbols that occurred with lower probability. In an analogous fashion, in order to decrease the average distortion, we can try to approximate the input better in regions of high probability, perhaps at the cost of worse approximations in regions of lower probability. We can do this by making the quantization intervals smaller in those regions that have more probability mass. If the source distribution is like the distribution shown in Figure 9.10, we would have smaller intervals near the origin. If we wanted to keep the number of intervals constant, this would mean we would have larger intervals away from the origin. A quantizer that has nonuniform intervals is called a *nonuniform quantizer*. An example of a nonuniform quantizer is shown in Figure 9.19.

Notice that the intervals closer to zero are smaller. Hence the maximum value that the quantizer error can take on is also smaller, resulting in a better approximation. We pay for this improvement in accuracy at lower input levels by incurring larger errors when the input falls in the outer intervals. However, as the probability of getting smaller input values is much higher than getting larger signal values, on the average the distortion will be lower than if we had a uniform quantizer. While a nonuniform quantizer provides lower average distortion, the design of nonuniform quantizers is also somewhat more complex. However, the basic idea is quite straightforward: find the decision boundaries and reconstruction levels that minimize the mean squared quantization error. We look at the design of nonuniform quantizers in more detail in the following sections.

9.6.1 pdf-Optimized Quantization

A direct approach for locating the best nonuniform quantizer, if we have a probability model for the source, is to find the $\{b_i\}$ and $\{y_i\}$ that minimize Equation (9.3). Setting the derivative of Equation (9.3) with respect to y_j to zero, and solving for y_j , we get

$$y_j = \frac{\int_{b_{j-1}}^{b_j} x f_X(x) dx}{\int_{b_{j-1}}^{b_j} f_X(x) dx}. \quad (9.27)$$

The output point for each quantization interval is the centroid of the probability mass in that interval. Taking the derivative with respect to b_j and setting it equal to zero, we get an expression for b_j as

$$b_j = \frac{y_{j+1} + y_j}{2}. \quad (9.28)$$

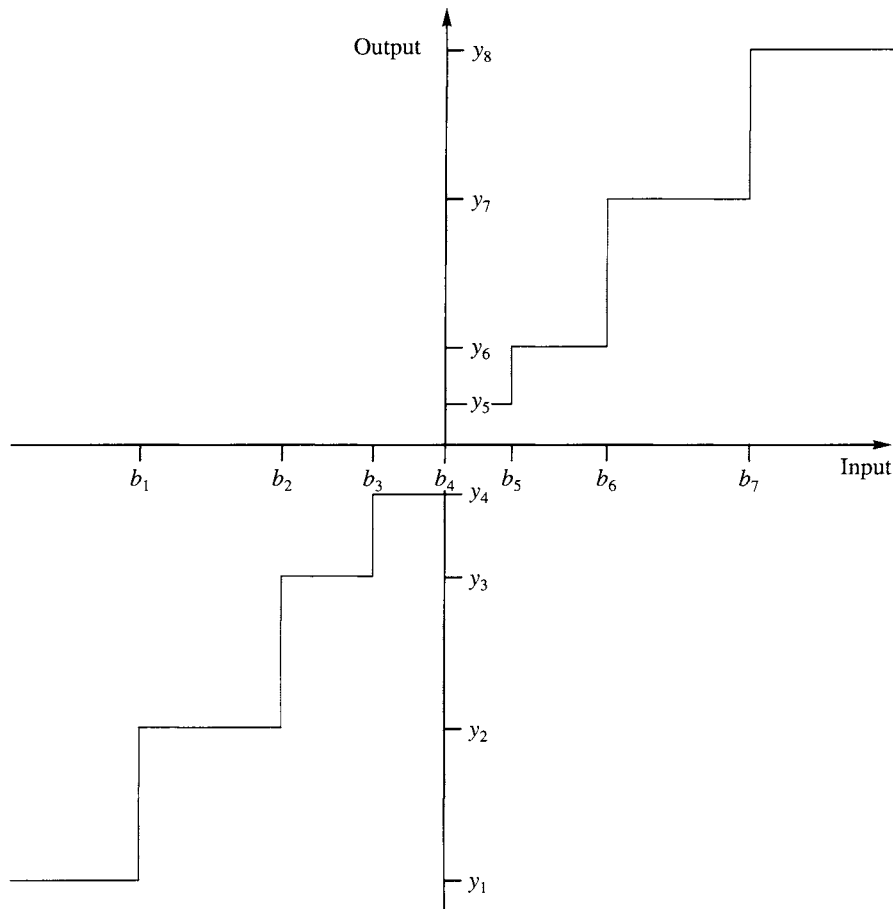


FIGURE 9.19 A nonuniform midrise quantizer.

The decision boundary is simply the midpoint of the two neighboring reconstruction levels. Solving these two equations will give us the values for the reconstruction levels and decision boundaries that minimize the mean squared quantization error. Unfortunately, to solve for y_j , we need the values of b_j and b_{j-1} , and to solve for b_j , we need the values of y_{j+1} and y_j . In a 1960 paper, Joel Max [108] showed how to solve the two equations iteratively. The same approach was described by Stuart P. Lloyd in a 1957 internal Bell Labs memorandum. Generally, credit goes to whomever publishes first, but in this case, because much of the early work in quantization was done at Bell Labs, Lloyd's work was given due credit and the algorithm became known as the Lloyd-Max algorithm. However, the story does not end (begin?) there. Allen Gersho [113] points out that the same algorithm was published by Lukaszewicz and Steinhaus in a Polish journal in 1955 [114]! Lloyd's paper remained unpublished until 1982, when it was finally published in a special issue of the *IEEE Transactions on Information Theory* devoted to quantization [115].

To see how this algorithm works, let us apply it to a specific situation. Suppose we want to design an M -level symmetric midrise quantizer. To define our symbols, we will use Figure 9.20. From the figure, we see that in order to design this quantizer, we need to obtain the reconstruction levels $\{y_1, y_2, \dots, y_{\frac{M}{2}}\}$ and the decision boundaries $\{b_1, b_2, \dots, b_{\frac{M}{2}-1}\}$. The reconstruction levels $\{y_{-1}, y_{-2}, \dots, y_{-\frac{M}{2}}\}$ and the decision boundaries $\{b_{-1}, b_{-2}, \dots, b_{-(\frac{M}{2}-1)}\}$ can be obtained through symmetry, the decision boundary b_0 is zero, and the decision boundary $b_{\frac{M}{2}}$ is simply the largest value the input can take on (for unbounded inputs this would be ∞).

Let us set j equal to 1 in Equation (9.27):

$$y_1 = \frac{\int_{b_0}^{b_1} x f_X(x) dx}{\int_{b_0}^{b_1} f_X(x) dx}. \quad (9.29)$$

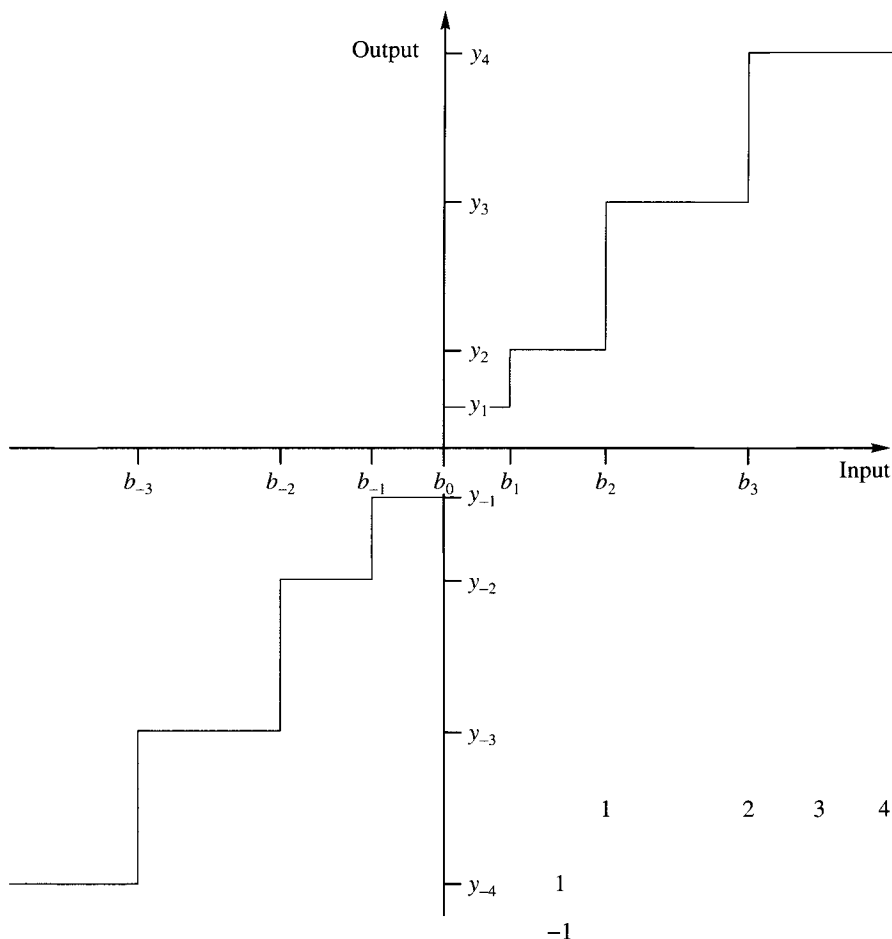


FIGURE 9.20 A nonuniform midrise quantizer.

As b_0 is known to be 0, we have two unknowns in this equation, b_1 and y_1 . We make a guess at y_1 and later we will try to refine this guess. Using this guess in Equation (9.29), we numerically find the value of b_1 that satisfies Equation (9.29). Setting j equal to 1 in Equation (9.28), and rearranging things slightly, we get

$$y_2 = 2b_1 + y_1 \quad (9.30)$$

from which we can compute y_2 . This value of y_2 can then be used in Equation (9.27) with $j = 2$ to find b_2 , which in turn can be used to find y_3 . We continue this process, until we obtain a value for $\{y_1, y_2, \dots, y_{M/2}\}$ and $\{b_1, b_2, \dots, b_{M/2-1}\}$. Note that the accuracy of all the values obtained to this point depends on the quality of our initial estimate of y_1 . We can check this by noting that $y_{M/2}$ is the centroid of the probability mass of the interval $[b_{M/2-1}, b_{M/2}]$. We know $b_{M/2}$ from our knowledge of the data. Therefore, we can compute the integral

$$y_{M/2} = \frac{\int_{b_{M/2-1}}^{b_{M/2}} x f_X(x) dx}{\int_{b_{M/2-1}}^{b_{M/2}} f_X(x) dx} \quad (9.31)$$

and compare it with the previously computed value of $y_{M/2}$. If the difference is less than some tolerance threshold, we can stop. Otherwise, we adjust the estimate of y_1 in the direction indicated by the sign of the difference and repeat the procedure.

Decision boundaries and reconstruction levels for various distributions and number of levels generated using this procedure are shown in Table 9.6. Notice that the distributions that have heavier tails also have larger outer step sizes. However, these same quantizers have smaller inner step sizes because they are more heavily peaked. The SNR for these quantizers is also listed in the table. Comparing these values with those for the *pdf*-optimized uniform quantizers, we can see a significant improvement, especially for distributions further away from the uniform distribution. Both uniform and nonuniform *pdf*-optimized, or Lloyd-Max,

TABLE 9.6 Quantizer boundary and reconstruction levels for nonuniform Gaussian and Laplacian quantizers.

Levels	b_i	Gaussian y_i	SNR	b_i	Laplacian y_i	SNR
4	0.0	0.4528	9.3 dB	0.0	0.4196	7.54 dB
	0.9816	1.510		1.1269	1.8340	
6	0.0	0.3177	12.41 dB	0.0	0.2998	10.51 dB
	0.6589	1.0		0.7195	1.1393	
	1.447	1.894		1.8464	2.5535	
8	0.0	0.2451	14.62 dB	0.0	0.2334	12.64 dB
	0.7560	0.6812		0.5332	0.8330	
	1.050	1.3440		1.2527	1.6725	
	1.748	2.1520		2.3796	3.0867	

quantizers have a number of interesting properties. We list these properties here (their proofs can be found in [116, 117, 118]):

- **Property 1:** The mean values of the input and output of a Lloyd-Max quantizer are equal.
- **Property 2:** For a given Lloyd-Max quantizer, the variance of the output is always less than or equal to the variance of the input.
- **Property 3:** The mean squared quantization error for a Lloyd-Max quantizer is given by

$$\sigma_q^2 = \sigma_x^2 - \sum_{j=1}^M y_j^2 P[b_{j-1} \leq X < b_j] \quad (9.32)$$

where σ_x^2 is the variance of the quantizer input, and the second term on the right-hand side is the second moment of the output (or variance if the input is zero mean).

- **Property 4:** Let N be the random variable corresponding to the quantization error. Then for a given Lloyd-Max quantizer,

$$E[XN] = -\sigma_q^2. \quad (9.33)$$

- **Property 5:** For a given Lloyd-Max quantizer, the quantizer output and the quantization noise are orthogonal:

$$E[Q(X)N \mid b_0, b_1, \dots, b_M] = 0. \quad (9.34)$$

Mismatch Effects

As in the case of uniform quantizers, the *pdf*-optimized nonuniform quantizers also have problems when the assumptions underlying their design are violated. In Figure 9.21 we show the effects of variance mismatch on a 4-bit Laplacian nonuniform quantizer.

This mismatch effect is a serious problem because in most communication systems the input variance can change considerably over time. A common example of this is the telephone system. Different people speak with differing amounts of loudness into the telephone. The quantizer used in the telephone system needs to be quite robust to the wide range of input variances in order to provide satisfactory service.

One solution to this problem is the use of adaptive quantization to match the quantizer to the changing input characteristics. We have already looked at adaptive quantization for the uniform quantizer. Generalizing the uniform adaptive quantizer to the nonuniform case is relatively straightforward, and we leave that as a practice exercise (see Problem 8). A somewhat different approach is to use a nonlinear mapping to flatten the performance curve shown in Figure 9.21. In order to study this approach, we need to view the nonuniform quantizer in a slightly different manner.

9.6.2 Companded Quantization

Instead of making the step size small, we could make the interval in which the input lies with high probability large—that is, expand the region in which the input lands with high

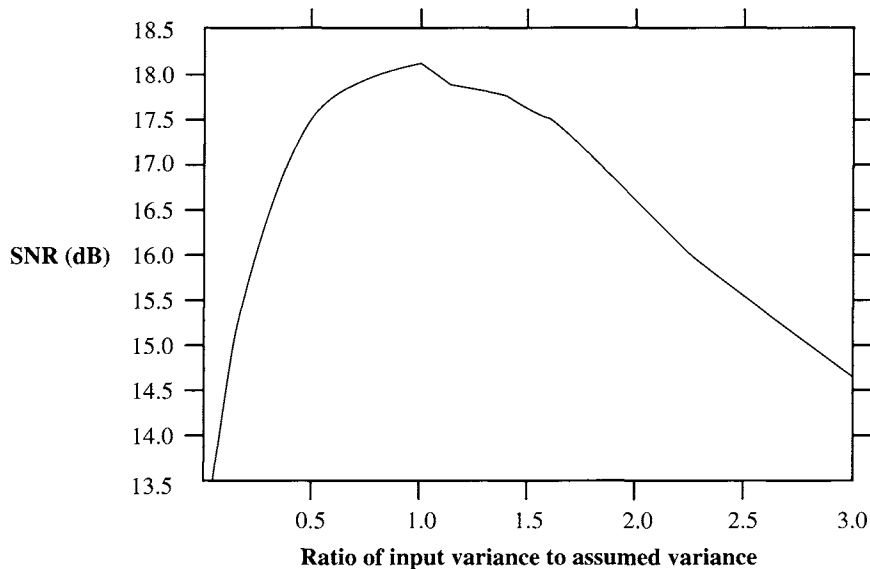


FIGURE 9. 21 Effect of mismatch on nonuniform quantization.

probability in proportion to the probability with which the input lands in this region. This is the idea behind companded quantization. This quantization approach can be represented by the block diagram shown in Figure 9.22. The input is first mapped through a *compressor* function. This function “stretches” the high-probability regions close to the origin, and correspondingly “compresses” the low-probability regions away from the origin. Thus, regions close to the origin in the input to the compressor occupy a greater fraction of the total region covered by the compressor. If the output of the compressor function is quantized using a uniform quantizer, and the quantized value transformed via an *expander* function,

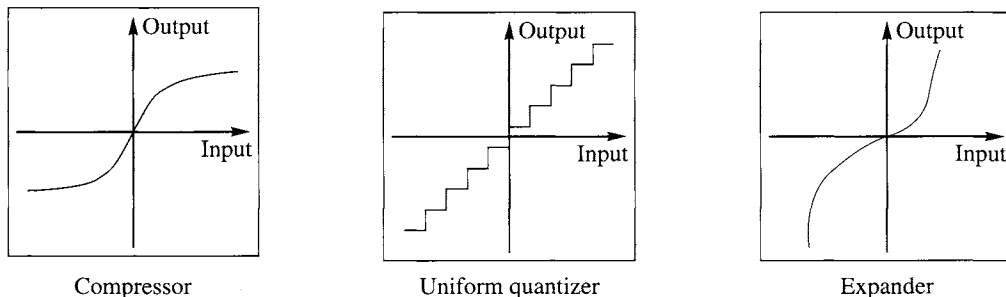


FIGURE 9. 22 Block diagram for log companded quantization.

the overall effect is the same as using a nonuniform quantizer. To see this, we devise a simple compander and see how the process functions.

Example 9.6.1:

Suppose we have a source that can be modeled as a random variable taking values in the interval $[-4, 4]$ with more probability mass near the origin than away from it. We want to quantize this using the quantizer of Figure 9.3. Let us try to flatten out this distribution using the following compander, and then compare the companded quantization with straightforward uniform quantization. The compressor characteristic we will use is given by the following equation:

$$c(x) = \begin{cases} 2x & \text{if } -1 \leq x \leq 1 \\ \frac{2x}{3} + \frac{4}{3} & x > 1 \\ \frac{2x}{3} - \frac{4}{3} & x < -1. \end{cases} \quad (9.35)$$

The mapping is shown graphically in Figure 9.23. The inverse mapping is given by

$$c^{-1}(x) = \begin{cases} \frac{x}{2} & \text{if } -2 \leq x \leq 2 \\ \frac{3x}{2} - 2 & x > 2 \\ \frac{3x}{2} + 2 & x < -2. \end{cases} \quad (9.36)$$

The inverse mapping is shown graphically in Figure 9.24.

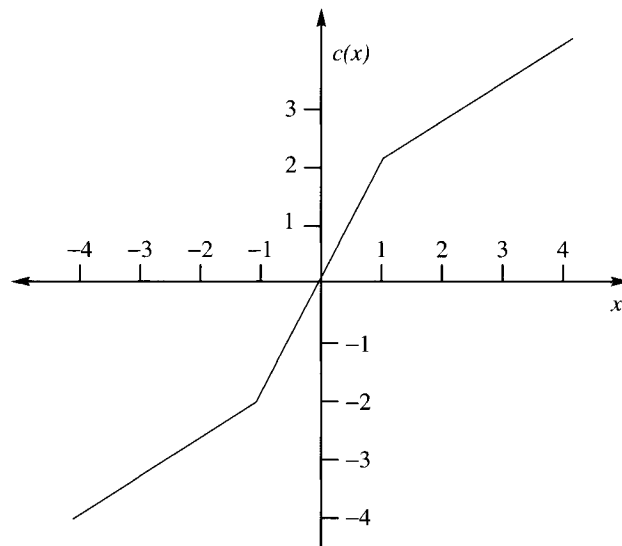


FIGURE 9.23 Compressor mapping.

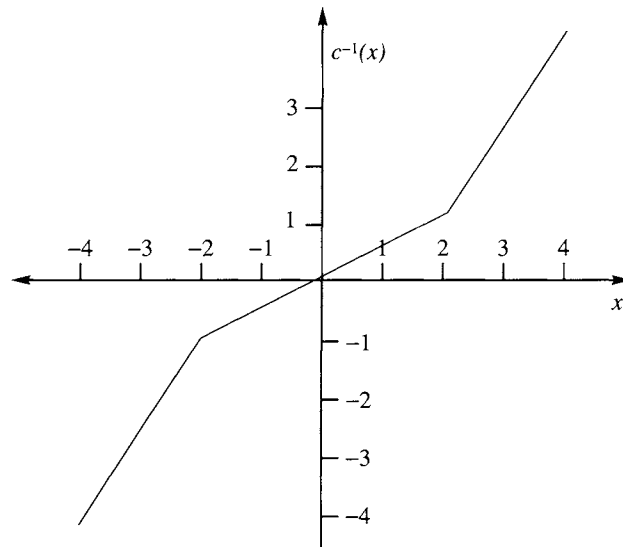


FIGURE 9. 24 Expander mapping.

Let's see how using these mappings affects the quantization error both near and far from the origin. Suppose we had an input of 0.9. If we quantize directly with the uniform quantizer, we get an output of 0.5, resulting in a quantization error of 0.4. If we use the companded quantizer, we first use the compressor mapping, mapping the input value of 0.9 to 1.8. Quantizing this with the same uniform quantizer results in an output of 1.5, with an apparent error of 0.3. The expander then maps this to the final reconstruction value of 0.75, which is 0.15 away from the input. Comparing 0.15 with 0.4, we can see that relative to the input we get a substantial reduction in the quantization error. In fact, for all values in the interval $[-1, 1]$, we will not get any increase in the quantization error, and for most values we will get a decrease in the quantization error (see Problem 6 at the end of this chapter). Of course, this will not be true for the values outside the $[-1, 1]$ interval. Suppose we have an input of 2.7. If we quantized this directly with the uniform quantizer, we would get an output of 2.5, with a corresponding error of 0.2. Applying the compressor mapping, the value of 2.7 would be mapped to 3.13, resulting in a quantized value of 3.5. Mapping this back through the expander, we get a reconstructed value of 3.25, which differs from the input by 0.55.

As we can see, the companded quantizer effectively works like a nonuniform quantizer with smaller quantization intervals in the interval $[-1, 1]$ and larger quantization intervals outside this interval. What is the effective input-output map of this quantizer? Notice that all inputs in the interval $[0, 0.5]$ get mapped into the interval $[0, 1]$, for which the quantizer output is 0.5, which in turn corresponds to the reconstruction value of 0.25. Essentially, all values in the interval $[0, 0.5]$ are represented by the value 0.25. Similarly, all values in

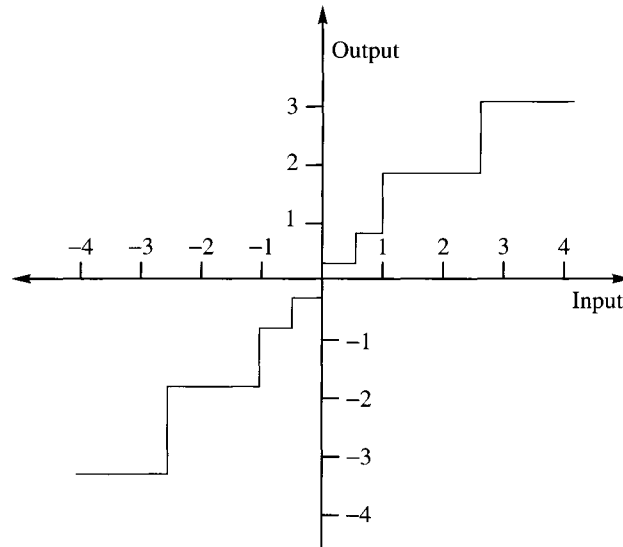


FIGURE 9. 25 Nonuniform companded quantizer.

the interval $[0.5, 1]$ are represented by the value 0.75, and so on. The effective quantizer input-output map is shown in Figure 9.25. ♦

If we bound the source output by some value x_{\max} , any nonuniform quantizer can always be represented as a companding quantizer. Let us see how we can use this fact to come up with quantizers that are robust to mismatch. First we need to look at some of the properties of high-rate quantizers, or quantizers with a large number of levels.

Define

$$\Delta_k = b_k - b_{k-1}. \quad (9.37)$$

If the number of levels is high, then the size of each quantization interval will be small, and we can assume that the *pdf* of the input $f_X(x)$ is essentially constant in each quantization interval. Then

$$f_X(x) = f_X(y_k) \quad \text{if } b_{k-1} \leq x < b_k. \quad (9.38)$$

Using this we can rewrite Equation (9.3) as

$$\sigma_q^2 = \sum_{i=1}^M f_X(y_i) \int_{b_{i-1}}^{b_i} (x - y_i)^2 dx \quad (9.39)$$

$$= \frac{1}{12} \sum_{i=1}^M f_X(y_i) \Delta_i^3. \quad (9.40)$$

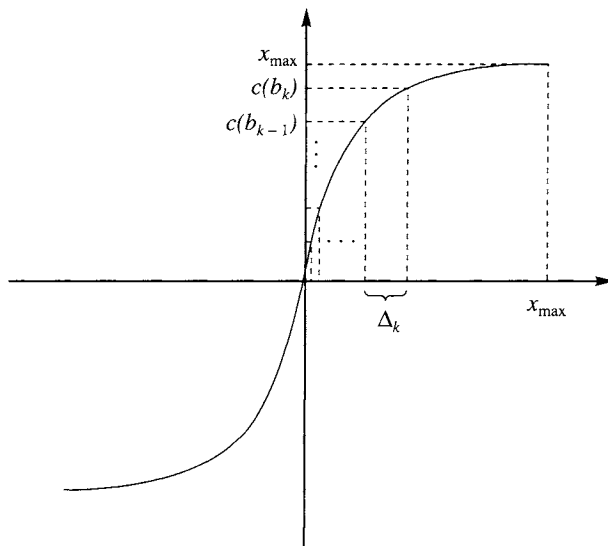


FIGURE 9.26 A compressor function.

Armed with this result, let us return to companded quantization. Let $c(x)$ be a companding characteristic for a symmetric quantizer, and let $c'(x)$ be the derivative of the compressor characteristic with respect to x . If the rate of the quantizer is high, that is, if there are a large number of levels, then within the k th interval, the compressor characteristic can be approximated by a straight line segment (see Figure 9.26), and we can write

$$c'(y_k) = \frac{c(b_k) - c(b_{k-1})}{\Delta_k}. \quad (9.41)$$

From Figure 9.26 we can also see that $c(b_k) - c(b_{k-1})$ is the step size of a uniform M -level quantizer. Therefore,

$$c(b_k) - c(b_{k-1}) = \frac{2x_{\max}}{M}. \quad (9.42)$$

Substituting this into Equation (9.41) and solving for Δ_k , we get

$$\Delta_k = \frac{2x_{\max}}{Mc'(y_k)}. \quad (9.43)$$

Finally, substituting this expression for Δ_k into Equation (9.40), we get the following relationship between the quantizer distortion, the *pdf* of the input, and the compressor characteristic:

$$\begin{aligned}\sigma_q^2 &= \frac{1}{12} \sum_{i=1}^M f_X(y_i) \left(\frac{2x_{\max}}{Mc'(y_i)} \right)^3 \\ &= \frac{x_{\max}^2}{3M^2} \sum_{i=1}^M \frac{f_X(y_i)}{c'^2(y_i)} \cdot \frac{2x_{\max}}{Mc'(y_i)} \\ &= \frac{x_{\max}^2}{3M^2} \sum_{i=1}^M \frac{f_X(y_i)}{c'^2(y_i)} \Delta_i\end{aligned}\quad (9.44)$$

which for small Δ_i can be written as

$$\sigma_q^2 = \frac{x_{\max}^2}{3M^2} \int_{-x_{\max}}^{x_{\max}} \frac{f_X(x)}{(c'(x))^2} dx. \quad (9.45)$$

This is a famous result, known as the Bennett integral after its discoverer, W.R. Bennett [119], and it has been widely used to analyze quantizers. We can see from this integral that the quantizer distortion is dependent on the *pdf* of the source sequence. However, it also tells us how to get rid of this dependence. Define

$$c'(x) = \frac{x_{\max}}{\alpha |x|}, \quad (9.46)$$

where α is a constant. From the Bennett integral we get

$$\sigma_q^2 = \frac{x_{\max}^2}{3M^2} \frac{\alpha^2}{x_{\max}^2} \int_{-x_{\max}}^{x_{\max}} x^2 f_X(x) dx \quad (9.47)$$

$$= \frac{\alpha^2}{3M^2} \sigma_x^2 \quad (9.48)$$

where

$$\sigma_x^2 = \int_{-x_{\max}}^{x_{\max}} x^2 f_X(x) dx. \quad (9.49)$$

Substituting the expression for σ_q^2 into the expression for SNR, we get

$$\text{SNR} = 10 \log_{10} \frac{\sigma_x^2}{\sigma_q^2} \quad (9.50)$$

$$= 10 \log_{10}(3M^2) - 20 \log_{10} \alpha \quad (9.51)$$

which is independent of the input *pdf*. This means that if we use a compressor characteristic whose derivative satisfies Equation (9.46), then regardless of the input variance, the signal-to-noise ratio will remain constant. This is an impressive result. However, we do need some caveats.

Notice that we are not saying that the mean squared quantization error is independent of the quantizer input. It is not, as is clear from Equation (9.48). Remember also that this

result is valid as long as the underlying assumptions are valid. When the input variance is very small, our assumption about the *pdf* being constant over the quantization interval is no longer valid, and when the variance of the input is very large, our assumption about the input being bounded by x_{\max} may no longer hold.

With fair warning, let us look at the resulting compressor characteristic. We can obtain the compressor characteristic by integrating Equation (9.46):

$$c(x) = x_{\max} + \beta \log \frac{|x|}{x_{\max}} \quad (9.52)$$

where β is a constant. The only problem with this compressor characteristic is that it becomes very large for small x . Therefore, in practice we approximate this characteristic with a function that is linear around the origin and logarithmic away from it.

Two companding characteristics that are widely used today are μ -law companding and A-law companding. The μ -law compressor function is given by

$$c(x) = x_{\max} \frac{\ln \left(1 + \mu \frac{|x|}{x_{\max}} \right)}{\ln(1 + \mu)} \operatorname{sgn}(x). \quad (9.53)$$

The expander function is given by

$$c^{-1}(x) = \frac{x_{\max}}{\mu} \left[\left(1 + \mu \frac{|x|}{x_{\max}} \right) - 1 \right] \operatorname{sgn}(x). \quad (9.54)$$

This companding characteristic with $\mu = 255$ is used in the telephone systems in North America and Japan. The rest of the world uses the A-law characteristic, which is given by

$$c(x) = \begin{cases} \frac{A|x|}{1 + \ln A} \operatorname{sgn}(x) & 0 \leq \frac{|x|}{x_{\max}} \leq \frac{1}{A} \\ x_{\max} \frac{1 + \ln \frac{A|x|}{x_{\max}}}{1 + \ln A} \operatorname{sgn}(x) & \frac{1}{A} \leq \frac{|x|}{x_{\max}} \leq 1 \end{cases} \quad (9.55)$$

and

$$c^{-1}(x) = \begin{cases} \frac{|x|}{A} (1 + \ln A) & 0 \leq \frac{|x|}{x_{\max}} \leq \frac{1}{1 + \ln A} \\ \frac{x_{\max}}{A} \exp \left[\frac{|x|}{x_{\max}} (1 + \ln A) - 1 \right] & \frac{1}{1 + \ln A} \leq \frac{|x|}{x_{\max}} \leq 1. \end{cases} \quad (9.56)$$

9.7 Entropy-Coded Quantization

In Section 9.3 we mentioned three tasks: selection of boundary values, selection of reconstruction levels, and selection of codewords. Up to this point we have talked about accomplishment of the first two tasks, with the performance measure being the mean squared quantization error. In this section we will look at accomplishing the third task, assigning codewords to the quantization interval. Recall that this becomes an issue when we use variable-length codes. In this section we will be looking at the latter situation, with the rate being the performance measure.

We can take two approaches to the variable-length coding of quantizer outputs. We can redesign the quantizer by taking into account the fact that the selection of the decision boundaries will affect the rate, or we can keep the design of the quantizer the same

(i.e., Lloyd-Max quantization) and simply entropy-code the quantizer output. Since the latter approach is by far the simpler one, let's look at it first.

9.7.1 Entropy Coding of Lloyd-Max Quantizer Outputs

The process of trying to find the optimum quantizer for a given number of levels and rate is a rather difficult task. An easier approach to incorporating entropy coding is to design a quantizer that minimizes the msqe, that is, a Lloyd-Max quantizer, then entropy-code its output.

In Table 9.7 we list the output entropies of uniform and nonuniform Lloyd-Max quantizers. Notice that while the difference in rate for lower levels is relatively small, for a larger number of levels, there can be a substantial difference between the fixed-rate and entropy-coded cases. For example, for 32 levels a fixed-rate quantizer would require 5 bits per sample. However, the entropy of a 32-level uniform quantizer for the Laplacian case is 3.779 bits per sample, which is more than 1 bit less. Notice that the difference between the fixed rate and the uniform quantizer entropy is generally greater than the difference between the fixed rate and the entropy of the output of the nonuniform quantizer. This is because the nonuniform quantizers have smaller step sizes in high-probability regions and larger step sizes in low-probability regions. This brings the probability of an input falling into a low-probability region and the probability of an input falling in a high-probability region closer together. This, in turn, raises the output entropy of the nonuniform quantizer with respect to the uniform quantizer. Finally, the closer the distribution is to being uniform, the less difference in the rates. Thus, the difference in rates is much less for the quantizer for the Gaussian source than the quantizer for the Laplacian source.

9.7.2 Entropy-Constrained Quantization ★

Although entropy coding the Lloyd-Max quantizer output is certainly simple, it is easy to see that we could probably do better if we take a fresh look at the problem of quantizer

TABLE 9.7 Output entropies in bits per sample for minimum mean squared error quantizers.

Number of Levels	Gaussian		Laplacian	
	Uniform	Nonuniform	Uniform	Nonuniform
4	1.904	1.911	1.751	1.728
6	2.409	2.442	2.127	2.207
8	2.759	2.824	2.394	2.479
16	3.602	3.765	3.063	3.473
32	4.449	4.730	3.779	4.427

design, this time with the entropy as a measure of rate rather than the alphabet size. The entropy of the quantizer output is given by

$$H(Q) = - \sum_{i=1}^M P_i \log_2 P_i \quad (9.57)$$

where P_i is the probability of the input to the quantizer falling in the i th quantization interval and is given by

$$P_i = \int_{b_{i-1}}^{b_i} f_X(x) dx. \quad (9.58)$$

Notice that the selection of the representation values $\{y_j\}$ has no effect on the rate. This means that we can select the representation values solely to minimize the distortion. However, the selection of the boundary values affects both the rate and the distortion. Initially, we found the reconstruction levels and decision boundaries that minimized the distortion, while keeping the rate fixed by fixing the quantizer alphabet size and assuming fixed-rate coding. In an analogous fashion, we can now keep the entropy fixed and try to minimize the distortion. Or, more formally:

For a given R_o , find the decision boundaries $\{b_j\}$ that minimize σ_q^2 given by Equation (9.3), subject to $H(Q) \leq R_o$.

The solution to this problem involves the solution of the following $M - 1$ nonlinear equations [120]:

$$\ln \frac{P_{l+1}}{P_l} = \lambda (y_{k+1} - y_k)(y_{k+1} + y_k - 2b_k) \quad (9.59)$$

where λ is adjusted to obtain the desired rate, and the reconstruction levels are obtained using Equation (9.27). A generalization of the method used to obtain the minimum mean squared error quantizers can be used to obtain solutions for this equation [121]. The process of finding optimum entropy-constrained quantizers looks complex. Fortunately, at higher rates we can show that the optimal quantizer is a uniform quantizer, simplifying the problem. Furthermore, while these results are derived for the high-rate case, it has been shown that the results also hold for lower rates [121].

9.7.3 High-Rate Optimum Quantization ★

At high rates, the design of optimum quantizers becomes simple, at least in theory. Gish and Pierce's work [122] says that at high rates the optimum entropy-coded quantizer is a uniform quantizer. Recall that any nonuniform quantizer can be represented by a compander and a uniform quantizer. Let us try to find the optimum compressor function at high rates that minimizes the entropy for a given distortion. Using the calculus of variations approach, we will construct the functional

$$J = H(Q) + \lambda \sigma_q^2, \quad (9.60)$$

then find the compressor characteristic to minimize it.

For the distortion σ_q^2 , we will use the Bennett integral shown in Equation (9.45). The quantizer entropy is given by Equation (9.57). For high rates, we can assume (as we did before) that the *pdf* $f_X(x)$ is constant over each quantization interval Δ_i , and we can replace Equation (9.58) by

$$P_i = f_X(y_i)\Delta_i. \quad (9.61)$$

Substituting this into Equation (9.57), we get

$$H(Q) = -\sum f_X(y_i)\Delta_i \log[f_X(y_i)\Delta_i] \quad (9.62)$$

$$= -\sum f_X(y_i) \log[f_X(y_i)]\Delta_i - \sum f_X(y_i) \log[\Delta_i]\Delta_i \quad (9.63)$$

$$= -\sum f_X(y_i) \log[f_X(y_i)]\Delta_i - \sum f_X(y_i) \log \frac{2x_{\max}/M}{c'(y_i)} \Delta_i \quad (9.64)$$

where we have used Equation (9.43) for Δ_i . For small Δ_i we can write this as

$$H(Q) = -\int f_X(x) \log f_X(x) dx - \int f_X(x) \log \frac{2x_{\max}/M}{c'(x)} dx \quad (9.65)$$

$$= -\int f_X(x) \log f_X(x) dx - \log \frac{2x_{\max}}{M} + \int f_X(x) \log c'(x) dx \quad (9.66)$$

where the first term is the differential entropy of the source $h(X)$. Let's define $g = c'(x)$. Then substituting the value of $H(Q)$ into Equation (9.60) and differentiating with respect to g , we get

$$\int f_X(x) [g^{-1} - 2\lambda \frac{x_{\max}^2}{3M^2} g^{-3}] dx = 0. \quad (9.67)$$

This equation is satisfied if the integrand is zero, which gives us

$$g = \sqrt{\frac{2\lambda}{3}} \frac{x_{\max}}{M} = K(\text{constant}). \quad (9.68)$$

Therefore,

$$c'(x) = K \quad (9.69)$$

and

$$c(x) = Kx + \alpha. \quad (9.70)$$

If we now use the boundary conditions $c(0) = 0$ and $c(x_{\max}) = x_{\max}$, we get $c(x) = x$, which is the compressor characteristic for a uniform quantizer. Thus, at high rates the optimum quantizer is a uniform quantizer.

Substituting this expression for the optimum compressor function in the Bennett integral, we get an expression for the distortion for the optimum quantizer:

$$\sigma_q^2 = \frac{x_{\max}^2}{3M^2}. \quad (9.71)$$

Substituting the expression for $c(x)$ in Equation (9.66), we get the expression for the entropy of the optimum quantizer:

$$H(Q) = h(X) - \log \frac{2x_{\max}}{M}. \quad (9.72)$$

Note that while this result provides us with an easy method for designing optimum quantizers, our derivation is only valid if the source *pdf* is entirely contained in the interval $[-x_{\max}, x_{\max}]$, and if the step size is small enough that we can reasonably assume the *pdf* to be constant over a quantization interval. Generally, these conditions can only be satisfied if we have an extremely large number of quantization intervals. While theoretically this is not much of a problem, most of these reconstruction levels will be rarely used. In practice, as mentioned in Chapter 3, entropy coding a source with a large output alphabet is very problematic. One way we can get around this is through the use of a technique called *recursive indexing*.

Recursive indexing is a mapping of a countable set to a collection of sequences of symbols from another set with finite size [76]. Given a countable set $A = \{a_0, a_1, \dots\}$ and a finite set $B = \{b_0, b_1, \dots, b_M\}$ of size $M + 1$, we can represent any element in A by a sequence of elements in B in the following manner:

1. Take the index i of element a_i of A .
2. Find the quotient m and remainder r of the index i such that

$$i = mM + r.$$

3. Generate the sequence: $\underbrace{b_M b_M \cdots b_M}_{m \text{ times}} b_r$.

B is called the representation set. We can see that given any element in A we will have a unique sequence from B representing it. Furthermore, no representative sequence is a prefix of any other sequence. Therefore, recursive indexing can be viewed as a trivial, uniquely decodable prefix code. The inverse mapping is given by

$$\underbrace{b_M b_M \cdots b_M}_{m \text{ times}} b_r \mapsto a_{mM+r}.$$

Since it is one-to-one, if it is used at the output of the quantizer to convert the index sequence of the quantizer output into the sequence of the recursive indices, the former can be recovered without error from the latter. Furthermore, when the size $M + 1$ of the representation set B is chosen appropriately, in effect we can achieve the reduction in the size of the output alphabets that are used for entropy coding.

Example 9.7.1:

Suppose we want to represent the set of nonnegative integers $A = \{0, 1, 2, \dots\}$ with the representation set $B = \{0, 1, 2, 3, 4, 5\}$. Then the value 12 would be represented by the sequence 5, 5, 2, and the value 16 would be represented by the sequence 5, 5, 5, 1. Whenever the

decoder sees the value 5, it simply adds on the next value until the next value is smaller than 5. For example, the sequence 3, 5, 1, 2, 5, 5, 1, 5, 0 would be decoded as 3, 6, 2, 11, 5. ♦

Recursive indexing is applicable to any representation of a large set by a small set. One way of applying recursive indexing to the problem of quantization is as follows: For a given step size $\Delta > 0$ and a positive integer K , define x_l and x_h as follows:

$$x_l = - \left\lfloor \frac{K-1}{2} \right\rfloor \Delta$$

$$x_h = x_l + (K-1)\Delta$$

where $\lfloor x \rfloor$ is the largest integer not exceeding x . We define a recursively indexed quantizer of size K to be a uniform quantizer with step size Δ and with x_l and x_h being its smallest and largest output levels. (Q defined this way also has 0 as its output level.) The quantization rule Q , for a given input value x , is as follows:

1. If x falls in the interval $(x_l + \frac{\Delta}{2}, x_h - \frac{\Delta}{2})$, then $Q(x)$ is the nearest output level.
2. If x is greater than $x_h - \frac{\Delta}{2}$, see if $x_1 \triangleq x - x_h \in (x_l + \frac{\Delta}{2}, x_h - \frac{\Delta}{2})$. If so, $Q(x) = (x_h, Q(x_1))$. If not, form $x_2 = x - 2x_h$ and do the same as for x_1 . This process continues until for some m , $x_m = x - mx_h$ falls in $(x_l + \frac{\Delta}{2}, x_h - \frac{\Delta}{2})$, which will be quantized into

$$Q(x) = (\underbrace{x_h, x_h, \dots, x_h}_{m \text{ times}}, Q(x_m)). \quad (9.73)$$

3. If x is smaller than $x_l + \frac{\Delta}{2}$, a similar procedure to the above is used; that is, form $x_m = x + mx_l$ so that it falls in $(x_l + \frac{\Delta}{2}, x_h - \frac{\Delta}{2})$, and quantize it to $(x_l, x_l, \dots, x_l, Q(x_m))$.

In summary, the quantizer operates in two modes: one when the input falls in the range (x_l, x_h) , the other when it falls outside of the specified range. The recursive nature in the second mode gives it the name.

We pay for the advantage of encoding a larger set by a smaller set in several ways. If we get a large input to our quantizer, the representation sequence may end up being intolerably large. We also get an increase in the rate. If $H(Q)$ is the entropy of the quantizer output, and γ is the average number of representation symbols per input symbol, then the minimum rate for the recursively indexed quantizer is $\gamma H(Q)$.

In practice, neither cost is too large. We can avoid the problem of intolerably large sequences by adopting some simple strategies for representing these sequences, and the value of γ is quite close to one for reasonable values of M . For Laplacian and Gaussian quantizers, a typical value for M would be 15 [76].

9.8 Summary

The area of quantization is a well-researched area and much is known about the subject. In this chapter, we looked at the design and performance of uniform and nonuniform quantizers for a variety of sources, and how the performance is affected when the assumptions used

in the design process are not correct. When the source statistics are not well known or change with time, we can use an adaptive strategy. One of the more popular approaches to adaptive quantization is the Jayant quantizer. We also looked at the issues involved with entropy-coded quantization.

Further Reading

With an area as broad as quantization, we had to keep some of the coverage rather cursory. However, there is a wealth of information on quantization available in the published literature. The following sources are especially useful for a general understanding of the area:

1. A very thorough coverage of quantization can be found in *Digital Coding of Waveforms*, by N.S. Jayant and P. Noll [123].
2. The paper “Quantization,” by A. Gersho, in *IEEE Communication Magazine*, September 1977 [113], provides an excellent tutorial coverage of many of the topics listed here.
3. The original paper by J. Max, “Quantization for Minimum Distortion,” *IRE Transactions on Information Theory* [108], contains a very accessible description of the design of *pdf*-optimized quantizers.
4. A thorough study of the effects of mismatch is provided by W. Mauersberger in [124].

9.9 Projects and Problems

1. Show that the derivative of the distortion expression in Equation (9.18) results in the expression in Equation (9.19). You will have to use a result called Leibnitz’s rule and the idea of a telescoping series. Leibnitz’s rule states that if $a(t)$ and $b(t)$ are monotonic, then

$$\frac{\delta}{\delta t} \int_{a(t)}^{b(t)} f(x, t) dx = \int_{a(t)}^{b(t)} \frac{\delta f(x, t)}{\delta t} dx + f(b(t), t) \frac{\delta b(t)}{\delta t} - f(a(t), t) \frac{\delta a(t)}{\delta t}. \quad (9.74)$$

2. Use the program `falspos` to solve Equation (9.19) numerically for the Gaussian and Laplacian distributions. You may have to modify the function `func` in order to do this.
3. Design a 3-bit uniform quantizer (specify the decision boundaries and representation levels) for a source with a Laplacian *pdf*, with a mean of 3 and a variance of 4.
4. The pixel values in the Sena image are not really distributed uniformly. Obtain a histogram of the image (you can use the `hist_image` routine), and using the fact that the quantized image should be as good an approximation as possible for the original, design 1-, 2-, and 3-bit quantizers for this image. Compare these with the results displayed in Figure 9.7. (For better comparison, you can reproduce the results in the book using the program `uquan_img`.)

5. Use the program `misuquan` to study the effect of mismatch between the input and assumed variances. How do these effects change with the quantizer alphabet size and the distribution type?
6. For the companding quantizer of Example 9.6.1, what are the outputs for the following inputs: $-0.8, 1.2, 0.5, 0.6, 3.2, -0.3$? Compare your results with the case when the input is directly quantized with a uniform quantizer with the same number of levels. Comment on your results.
7. Use the test images *Sena* and *Bookshelf1* to study the trade-offs involved in the selection of block sizes in the forward adaptive quantization scheme described in Example 9.5.2. Compare this with a more traditional forward adaptive scheme in which the variance is estimated and transmitted. The variance information should be transmitted using a uniform quantizer with differing number of bits.
8. Generalize the Jayant quantizer to the nonuniform case. Assume that the input is from a known distribution with unknown variance. Simulate the performance of this quantizer over the same range of ratio of variances as we have done for the uniform case. Compare your results to the fixed nonuniform quantizer and the adaptive uniform quantizer. To get a start on your program, you may wish to use `misnuq.c` and `juquan.c`.
9. Let's look at the rate distortion performance of the various quantizers.
 - (a) Plot the rate-distortion function $R(D)$ for a Gaussian source with mean zero and variance $\sigma_X^2 = 2$.
 - (b) Assuming fixed length codewords, compute the rate and distortion for 1, 2, and 3 bit pdf-optimized nonuniform quantizers. Also, assume that X is a Gaussian random variable with mean zero and $\sigma_X^2 = 2$. Plot these values on the same graph with \mathbf{x} 's.
 - (c) For the 2 and 3 bit quantizers, compute the rate and distortion assuming that the quantizer outputs are entropy coded. Plot these on the graph with \mathbf{o} 's.