Chapter 5
Quantisation and PCM

In Chapter Five

- ✓ Quantisation: An in-depth study of scalar quantisation involving an introduction to the different types of quantisation, detailed discussion of uniform quantisation, its performance measure, design parameters, trade-offs and drawbacks, and a comprehensive specification of logarithmic non-uniform quantisation that delivers a constant signal-to-quantisation noise ratio (SQNR) across all input signal levels. The benefit and price of non-uniform quantisation are also quantified in terms of companding gain and companding penalty.
- ✓ PCM: A comprehensive treatment of A-law and μ-law PCM source coding standards, including derivation of the standards from the ideal logarithmic compression curve through piece-wise linear approximations, detailed SQNR analysis, and performance comparisons with linear ADC.
- ✓ Lossy data compression: A brief overview highlighting various categories of low bit rate speech coding, namely waveform coders, vocoders and hybrid coders. The discussion particularly focuses on speech quality measures and trade-offs in low bit rate speech coding, and on differential quantisation, linear prediction and various special cases of bit rate reduction, including ADPCM, ADM and LPC-10.

5.1 Introduction

This chapter completes the treatment of the digitisation or, more appropriately, source coding of analogue signals started in the previous chapter. It deals in detail with the processes involved in converting a sampled analogue signal into a sequence of binary numbers (called *bit stream*) as illustrated in Fig. 1.6. We discuss the quantisation process with an emphasis on the performance metric of signal-to-quantisation noise ratio (SQNR), and the measures and trade-offs involved in maximising SQNR as well as ensuring its consistency across the full range of input signal amplitudes. The two standardised methods for binary representation of non-uniformly quantised signals, namely A-law and μ-law pulse code modulation (PCM), are then presented and their bandwidth savings and SQNR improvement for weak signals (called *companding gain*) when compared to linear ADC are evaluated. As always there is a price to pay because this improvement afforded to weak signals is achieved at the expense of a coarser quantisation of stronger signals and hence a reduction in their SQNR, which we quantify as *companding penalty*.

First we discuss the process of uniform quantisation, explain the subtle differences between the mid-rise and mid-tread types, calculate quantisation noise and explore various measures for reducing this noise power and the costs involved.

It emerges from our evaluation that uniform quantisation is unsuitable especially for voice communication due to its excessive bandwidth requirement and the inconsistency of signal quality when input signal amplitudes are spread over a wide range as is typically the case for speech signals. We then set about developing a method of non-uniform quantisation that fully remedies these deficiencies and find that this can be achieved through a coarser quantisation of large input signals and finer quantisation of small inputs using signal compression prior to uniform quantisation. The transfer characteristic of the compressor device is derived and is shown to obey a logarithmic function. A piecewise linear approximation of this theoretical curve to overcome practical implementation difficulties is discussed and expressions are derived to assess the impact of the approximation on SQNR which is no longer strictly constant but varies by about 2.5 dB over a 36 dB input amplitude range.

A 64 kb/s PCM is the toll quality benchmark for telephone speech communication, but this bit rate is excessive in most storage and wireless communication applications. We therefore include a brief discussion of bit rate reduction measures for telephone speech coding, known generally as *lossy data compression* or more specifically as *low bit rate speech coding*. There are a number of important worked examples that serve to further clarify and extend the concepts discussed.

5.2 Concept and classes of quantisation

The sampled signal $g(nT_s)$, obtained by sampling an analogue signal $g(t)$ at regular intervals T_s as discussed in the previous chapter, is still an analogue signal since it can take on any value in a continuum from the smallest to the largest value of $g(t)$. To transform $g(nT_s)$ into a digital signal, which by definition has discrete values taken from a finite set of numbers, we must replace (i.e. approximate) each value of $g(nT_s)$ by its nearest neighbour in the finite set of allowed values or levels. This is the process of quantisation. Unlike sampling, quantisation is an irreversible process that introduces irrecoverable errors (called *quantisation noise*) because once a sample value has been approximated by its nearest allowed level we lose the detail and information that would allow us to recover (i.e. return to) the original value. However, human audio-visual perception is limited in its sensitivity, so the quantised signal will be subjectively identical to the original signal if the approximations (or loss of detail or added quantisation noise) are kept small. Although we can make this noise as small as we wish by setting a large number of closely spaced allowed levels, the design goal is always to achieve a *subjectively acceptable quality* through a measure known as the *signal-to-quantisation noise ratio* (SQNR), rather than to aim for an *objectively perfect quality* at prohibitive costs.

Quantisation can be classified as *uniform* or *nonuniform*, and *mid-tread* or *mid-rise*. In uniform quantisation the quantisation levels to which sample values are approximated are equally spaced, whereas in non-uniform quantisation the spacing is unequal, being smaller near zero and progressively larger away from the origin. Mid-tread (also known as mid-step) quantisation has the value zero as a quantisation level, whereas in mid-rise quantisation the value zero is at the boundary between two quantisation intervals. The description 'mid' is a reference to the fact that zero is the middle value of the quantiser input range from say $-C$ to $+C$, and it is one of the levels or steps or treads in mid-tread, whereas in mid-rise quantisers zero is a point of transition where you 'rise' from one quantisation interval to another.

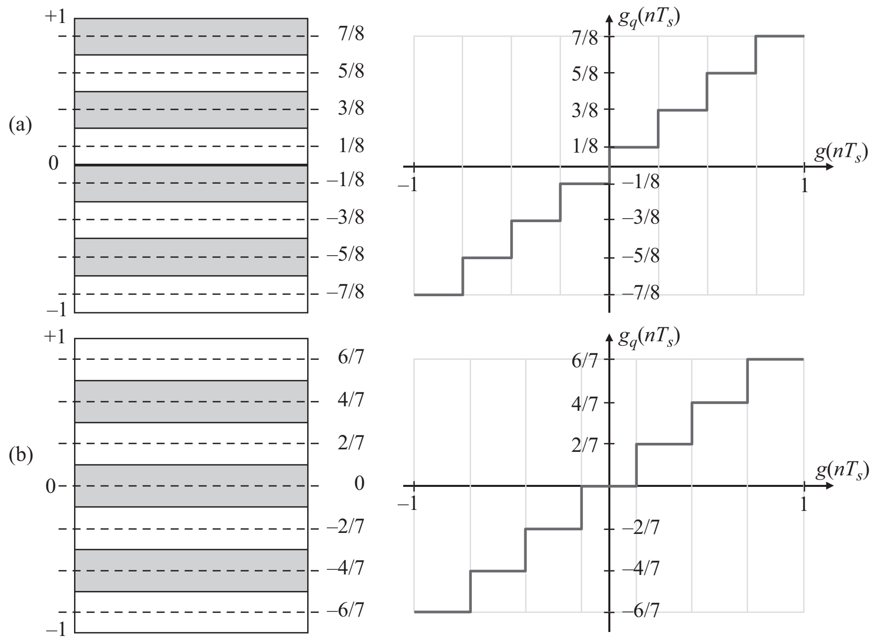


Fig. 5.1 Quantisation types: (a) uniform mid-rise; (b) uniform mid-tread

Fig. 5.1 illustrates uniform quantisation, based on $N = 8$ quantisation intervals and a normalised input range (–1, +1), of the mid-rise type in (a) and the mid-tread type in (b). Two formats have been adopted for this illustration. The RHS shows the *transfer characteristic* of each quantiser, i.e. graph of quantised output $g_q(nT_s)$ versus sampled input $g(nT_s)$, whereas the LHS features alternately shaded quantisation intervals covering the (normalised) quantiser range -1 to $+1$. The dashed line bisecting each interval is the quantiser output for all inputs in that interval, i.e. it is the quantisation level to which all sample values falling in the interval are approximated. Both types of uniform quantisers have *odd symmetry*, but there are significant differences.

First of all, on an idle line with $g(nT_s) = 0$ V, thermal noise would cause the output of a mid-rise quantiser to jump randomly between two steps (in this case between $-1/8$ V and $+1/8$ V) resulting in *noise chatter* on idle lines. On the other hand, in a mid-tread quantiser this idle condition produces a steady 0 V quantised output thus eliminating noise chatter. Second, mid-rise quantisers have $N/2$ quantisation intervals above zero and $N/2$ intervals below, making a total of N intervals; whereas mid-tread quantisers have $N/2 - 1$ intervals above and below the one interval containing zero, making a total of $N - 1$ intervals. This means that to cover the same quantiser range, say $(-C, +C)$, a mid-tread quantiser has to use a larger step size Δ than a mid-rise quantiser, and hence incurs larger approximation errors or quantisation noise. The difference in step size is $2C/(N-1) - 2C/N \approx 2C/N^2$, which is quite small for large N. Mid-tread quantisation is used in the μ-law PCM standard, whereas A-law PCM employs mid-rise quantisation. In both standards the impact of the drawbacks highlighted here is minimised in that $N = 256$, which is large enough to make the difference in step size negligible, and non-uniform quantisation is used with a very small starting step size Δ_0, which makes idle line noise chatter $\pm\Delta_0/2$ insignificant.

In what follows, we discuss uniform quantisation based on the mid-rise structure and derive the design parameters and performance measures of quantisation in general. We then outline the drawbacks of uniform quantisation and find that a satisfactory solution lies in non-uniform quantisation designed with step sizes that increase exponentially away from the origin.

5.3 Uniform quantisation

Consider a quantiser of range $(-C, +C)$ divided into N equal quantisation intervals as shown in Fig. 5.2. The quantised value Q_j, $j = 0, 1, 2, 3, \ldots, N-1$, of each interval is the *mean* of all input samples that fall in that interval. On the assumption that the samples in each interval are *equally likely to be located anywhere within the interval*, in other words that they have a *uniform distribution*, the quantised value Q_j is the midpoint of the j^{th} interval. Since these intervals, or equivalently quantised values Q_j, will be numbered using k binary digits (bits) and each output identified by this number, the number of intervals N has to be a power of 2 to ensure that all 2^k combinations of the k bits are utilised. That is

$$N = 2^k, \quad \Rightarrow \quad k = \log_2 N \tag{5.1}$$

The size of each interval, called *quantiser step size*, is

$$\Delta = \frac{2C}{N} = \frac{C}{2^{k-1}} \tag{5.2}$$

A quantiser is characterised by the number N of its quantisation levels or the number k of bits per sample, which is the number of bits needed to uniquely identify each of the N levels. We may also characterise a quantiser by specifying its

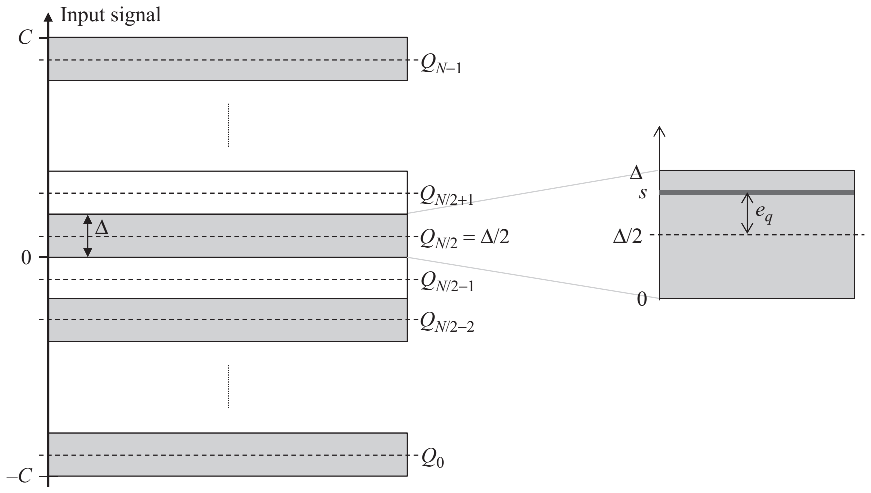


Fig. 5.2 Analysis of uniform quantisation. The interval just above zero is blown up on the RHS and shows the quantisation error e_q incurred when input sample s in this interval is quantised

dynamic range, which is the ratio between the largest amplitude A_2 of a sinusoidal signal that avoids clipping by the quantiser and the largest amplitude A_1 of a sinusoid whose variations are confined entirely to one interval of the quantiser and hence go undetected. Noting that $A_2 = C$ and $A_1 = \Delta/2$, we use Eq. (5.2) to obtain

$$\begin{aligned}\text{Dynamic range} &= \frac{C}{\Delta/2} = 2^k \\ &= 6k \text{ (dB)}\end{aligned} \tag{5.3}$$

Thus the dynamic range of a quantiser depends only on the number of bits per sample and increases by 6 dB per extra bit used to represent each sample.

The error incurred when an input sample s that falls in the j^{th} interval is represented by the quantised value Q_j of that interval is

$$e_q = s - Q_j \tag{5.4}$$

Ignoring the sign, e_q reaches a maximum $e_{q\,\max}$ when s is just below the upper limit of the interval (i.e. $s = Q_j + \Delta/2$) or s is just above the lower limit of the interval (i.e. $s = Q_j - \Delta/2$). Thus maximum quantisation error is half the quantiser step size:

$$e_{q\,\max} = \frac{\Delta}{2} = \frac{C}{2^k} \tag{5.5}$$

With the sample values in each interval equally likely to be above or below the interval's quantised level, e_q has a mean of zero and therefore its variance and mean square value (which defines normalised power) are identical. If the input signal lies entirely within the quantiser range, i.e. it does not overload the quantiser, the error statistics are the same across all the uniform quantiser intervals spanned by the input. Therefore to determine *mean square quantisation error* (MSQE), which gives quantisation noise power, we focus on the interval just above zero (shown blown out on the RHS of Fig. 5.2) and sum over the entire interval ($0 \to \Delta$) the product of $e_q^2 = (s - \Delta/2)^2$ and the probability ds/Δ that the sample s lies in an infinitesimal interval ds around s. This gives

$$\begin{aligned}\text{MSQE} &= \int_0^\Delta e_q^2 ds/\Delta = \frac{1}{\Delta}\int_0^\Delta (s - \Delta/2)^2 ds \\ &= \frac{1}{\Delta}\left[\left(\frac{s^3}{3} - \frac{s^2\Delta}{2} + \frac{s\Delta^2}{4}\right)\Bigg|_0^\Delta\right] \\ &= \Delta^2/12\end{aligned} \tag{5.6}$$

It can be seen that quantisation error depends only on step size. Assuming small Δ this error will have the same effect as thermal noise on the quantised signal and is therefore referred to as *quantisation noise* with rms value:

$$e_{\text{qrms}} = \sqrt{\text{MSQE}} = \frac{\Delta}{2\sqrt{3}} \tag{5.7}$$

We can make this noise due to quantisation as small as desired by sufficiently reducing Δ, which, from Eq. (5.2), is achieved by increasing k, the number of bits per sample. Clearly, this increases the bit rate $R_b = kf_s$ (where f_s is sampling rate) of the resulting digital signal and hence the transmission bandwidth required.

The quality of the quantised signal is measured by the *signal-to-quantisation noise ratio* (SQNR), which is the ratio between signal power and quantisation noise power:

$$\text{SQNR} = \frac{\text{Signal power}}{\text{Quantisation noise power}} = \frac{P_s}{\text{MSQE}} \tag{5.8}$$

Let the input signal has peak value A_p, rms value A_{rms}, and peak-to-rms ratio $R = A_p/A_{\text{rms}}$. It follows that $P_s = A_{\text{rms}}^2 = (A_p/R)^2$, and hence, using Eqs. (5.6) and (5.2), that

$$\begin{aligned}\text{SQNR} &= \frac{A_p{}^2/R^2}{\Delta^2/12} = \frac{12A_p{}^2 \times (2^{k-1})^2}{R^2C^2} \\ &= \frac{3A_p{}^2}{R^2C^2}2^{2k}\end{aligned} \tag{5.9}$$

Expressing this in dB yields

$$\text{SQNR} = 4.77 + 6.02k - 20\log(R) + 20\log(A_p/C)\ \text{dB} \tag{5.10}$$

Recall that it is assumed and in fact required that the quantiser is never overloaded so that the peak value of the input signal $A_p \le C$, which means that SQNR is reduced by the last term in the above equation. SQNR therefore reduces at the rate of 20 dB per decade reduction in input signal amplitude. The highest SQNR is experienced by a signal that fully loads the quantiser so that $A_p = C$ and $20\log(A_p/C) = 0$, which yields

$$\text{SQNR}_{\text{max}} = 4.77 + 6.02k - 20\log(R)\ \text{dB} \tag{5.11}$$

The highest SQNR achievable for different signals through uniform quantisation can be determined if we insert the value of R for that signal into Eq. (5.11). For example, sinusoidal signals have $R = \sqrt{2}$; speech signals have $R \approx 9$ dB; and a signal with uniform probability density function (pdf), i.e. the signal is equally likely to have any value in the range $(-C, C)$, has $R = \sqrt{3}$; so Eq. (5.11) gives

$$\text{SQNR}_{\text{max}} = \begin{cases} 6.02k + 1.76\ \text{dB}, & \text{Sinusoidal signal} \\ 6.02k - 4.23\ \text{dB}, & \text{Speech signal} \\ 6.02k\ \text{dB}, & \text{Signal with uniform pdf} \end{cases} \tag{5.12}$$

5.3.1 Quantisation design parameters and trade-offs

It is worth taking a moment to examine the parameters and trade-offs involved in achieving the usual design goal of maximising SQNR. These are encapsulated by Eqs. (5.9) and (5.10).

- SQNR increases exponentially with k, which, other factors being equal, is directly proportional to the transmission bandwidth required by the resulting digital signal. More specifically SQNR increases by 6 dB per extra bit used to represent each quantised sample. If information about the quantised levels is subsequently conveyed to the receiver without error then no further degradation will be introduced into the signal and this SQNR will be the signal to noise ratio (SNR) of the analogue signal recovered at the receiver, assuming that aperture distortion due to flat-top pulses at the reconstruction filter is avoided as discussed in the previous chapter. Therefore quantisation (and ultimately PCM) provides an effective mechanism for trading bandwidth for improvement in SNR. If the signal-to-noise ratio is SNR_1 at bandwidth B_1 and we increase the number of bits per sample by a factor $n > 1$, which means bandwidth also increases by the same factor to nB_1 (other factors being equal), then SNR increases exponentially to $(\mathrm{SNR}_1)^n$, assuming the factor $3A_p^2/R^2C^2$ in Eq. (5.9) equals unity, which is the case if the signal fully loads the quantiser and $R = \sqrt{3}$.

- SQNR is inversely proportional to the square of the quantiser range $2C$ needed to fully accommodate the input signal without clipping. Thus reducing C by a factor $n > 1$ increases SQNR by a factor n^2 for the same number of bits per sample. Alternatively, reducing C by a factor $n > 1$ allows us to achieve the same SQNR using $\log_2 n$ fewer bits per sample. For example, reducing C by a factor of 16 allows the same SQNR to be achieved using $k = 4$ bits per sample as is realised with $k = 8$. You are right to wonder how, given an input signal with amplitude $A_p = C$, we may reduce C without causing severe clipping distortion. The trick is to recognise that it is possible to avoid direct quantisation of the input signal samples by exploiting the correlation that exists between neighbouring samples of information-bearing signals. Basically, we use the past p samples $(p \geq 1)$ to obtain a prediction $\hat{g}(nT_s)$ of the current sample $g(nT_s)$ and obtain the prediction error $\varepsilon(nT_s) = g(nT_s) - \hat{g}(nT_s)$. We then quantise and transmit $\varepsilon(nT_s)$ rather than $g(nT_s)$. The receiver makes its own local prediction (based on the same algorithm used by the transmitter) and then simply adds the received error to obtain the original sample value. If the sampling rate is sufficiently high and the prediction algorithm is well designed, the errors are very small so that the range of $\varepsilon(nT_s)$ is much smaller than the range of $g(nT_s)$, and this allows us to reduce quantiser range accordingly and hence to realise the benefits discussed above. This is the principle behind various types of *differential quantisers* (e.g. differential pulse code modulation, DPCM).
- SQNR increases by 20 dB per decade increase in the ratio r $(= A_p/C)$ between the peak-to-peak amplitude of the input signal and the quantiser input range, up to a maximum at $r = 1$. This maximum SQNR is given by Eq. (5.11) for general signals and by Eq. (5.12) for three special cases. So SQNR is maximised by ensuring that the input signal fully loads the quantiser. This is okay for signals such as a sinusoid that have a low peak-to-rms ratio R. However, for signals such as speech that have higher R in addition to dynamic variations featuring a mixture of high-amplitude and low-amplitude sub-intervals, SQNR during weak passages will be significantly lower than the maximum SQNR attained in strong passages. Note that this problem cannot be tackled by scaling the weak signals to fully load the quantiser since this would cause clipping distortion during the strong passages that intermittently follow.

5.3.2 Shortcomings of uniform quantisation

Uniform quantisation has two significant drawbacks that make it unsuitable for digital transmission systems although it is widely used for linear analogue to digital conversion (ADC) in digital signal processing (DSP) systems.

- First of all, in order to faithfully represent crucial but small variations in the sampled signal, such as consonants in speech which carry intelligibility but are usually of weak amplitudes, the quantiser step size Δ must be sufficiently small, and this requires a large k (bits/sample) and hence a large transmission bandwidth and storage capacity for the resulting digital signal. Take the example of speech with a dynamic range of ~60 dB, which gives the ratio between highest and lowest sample magnitude as $10^{60/20} = 1000$. What this means is that if the strongest sub-intervals within the speech signal have peak values as high as 1 V then the weakest significant sub-intervals (not silence periods) may have peak values as low as 1 mV. If we are to faithfully quantise these small samples then the step size must be no more than 1 mV, which leads to 1000 quantisation intervals in the positive range and a further 1000 in the negative range, making a total of 2048 intervals (to the next power of 2). This requires $k = \log_2(2048) = 11$ bits/sample, which combined with a sampling rate of 8000 samples per second (i.e. 8 kHz) produces a bit rate of 88 kb/s and imposes an unacceptably high transmission bandwidth requirement.
- Second, uniform quantisation produces an SQNR that varies across the input signal range. Near the origin the maximum quantisation error Δ/2 is comparable in magnitude to the small sample values A_L in this region and therefore the distortion is perceptible, whereas near the top end of the quantiser range this error is negligible compared to the large sample values A_H. So $A_L^2/\Delta^2 \ll A_H^2/\Delta^2$ and SQNR at the bottom end is much lower than SQNR at the top end. Another way of looking at this problem is that the small samples A_L underload the quantiser when compared to the large samples A_H and therefore their SQNR is $20\log_{10}(A_H/A_L)$ dB lower.

It is desirable, especially in telephony, to maintain a high and consistent SQNR and service quality across all regimes of significant signal amplitudes while making judicious use of bandwidth. An excellent solution is provided through non-uniform quantisation designed with step sizes that increase exponentially away from the origin so that large input samples are coarsely quantised using larger step sizes whereas smaller input samples are more finely quantised using smaller step sizes. This will be dealt with in the next section after the following worked examples.

Worked Example 5.1: Quantiser Parameters

Given a 4-bit uniform quantiser (also called linear ADC) that covers the input range –16 V to 16 V, we wish to determine the following: (a) quantiser step size Δ; (b) quantisation noise power; (c) rms quantisation noise voltage; (d) signal-to-quantisation noise ratio (SQNR) when the quantiser is used to digitise a sinusoidal signal of amplitude $A_m = 16$ V; (e) SQNR when the input is a sinusoid of amplitude 1 V.

(a) Quantiser peak $C = 16$ V and range $= 2C = 32$ V. The quantiser uses $k = 4$ bits per sample. Eq. (5.2) yields step size

$$\Delta = \frac{C}{2^{k-1}} = \frac{16}{2^3} = 2 \text{ V}$$

(b) Quantisation noise power is given by MSQE in Eq. (5.6) as

$$\text{MSQE} = \frac{\Delta^2}{12} = \frac{2^2}{12} = \frac{1}{3}\ \text{W}$$

(c) rms quantisation noise voltage is of course the square root of noise power:

$$e_{\text{qrms}} = \sqrt{\text{MSQE}} = \sqrt{1/3} = 0.5774\ \text{V}$$

(d) Signal power $P_s = A_m^2/2 = 16^2/2 = 128\ \text{W}$. Thus

$$\text{SQNR} = \frac{P_s}{\text{MSQE}} = \frac{128}{1/3} = 384 = 25.84\ \text{dB}$$

Note that the above result can be obtained using Eq. (5.12) for a sinusoid that fully loads a quantiser (as is the case here): SQNR = $6.02 \times 4 + 1.76 = 25.84\ \text{dB}$.

(e) Signal power $P_s = A_m^2/2 = 1^2/2 = 1/2\ \text{W}$. Thus

$$\text{SQNR} = \frac{P_s}{\text{MSQE}} = \frac{1/2}{1/3} = 1.5 = 1.76\ \text{dB}$$

Again note that this could have been solved using Eq. (5.10), with $k = 4$, $A_p = 1$ V, $C = 16$ V, $R = \sqrt{2}$ (for a sinusoid) to obtain

$$\begin{aligned}\text{SQNR} &= 4.77 + 6.02k - 20\log(R) + 20\log(A_p/C)\ \text{dB}\\ &= 4.77 + 6.02 \times 4 - 20\log(\sqrt{2}) + 20\log(1/16)\ \text{dB}\\ &= 25.84 - 24.08\ \text{dB}\\ &= 1.76\ \text{dB}\end{aligned}$$

The difference in SQNR between (d) and (e) serves to highlight the problem with uniform quantisation. A strong input signal in (d) enjoys a reasonable SQNR, whereas a weak input signal in (e) with amplitude 24 dB below that of signal (d) experiences a poor SQNR that is lower than the SQNR in (d) by the same dB.

Worked Example 5.2: Dynamic Range

An analogue signal of range −8 V to +8 V and rms value 6 V fully loads the quantiser of a linear ADC that has a dynamic range of 42 dB. We wish to determine the signal-to-quantisation noise ratio (SQNR) in dB of the ADC output.

From Eq. (5.3) the number of bits (per sample) of the ADC is

$$k = \frac{\text{Dynamic range}}{6} = \frac{42}{6} = 7$$

The peak-to-rms ratio of the input signal is $R = 8/6 = 4/3$.

Substituting these values into Eq. (5.11), since the quantiser is fully loaded, yields

$$\begin{aligned}\text{SQNR} &= 4.77 + 6.02k - 20\log(R)\ \text{dB}\\ &= 4.77 + 6.02 \times 7 - 20\log(4/3)\ \text{dB}\\ &= 44.4\ \text{dB}\end{aligned}$$

5.4 Non-uniform quantisation

The main goal of non-uniform quantisation is to achieve a constant SQNR across all levels of input signal amplitude. This requires coarse quantisation of large input samples using large step sizes and finer quantisation of small input samples using smaller step sizes. In other words, a wider range of large input samples than that of small input samples is mapped to one quantised output level, meaning that the larger samples are effectively compressed. And since these output levels are then sequentially (i.e. uniformly) numbered irrespective of the size of the gap between adjacent levels, it follows that non-uniform quantisation is equivalent to a two-stage process of *compression* followed by uniform quantisation (or linear ADC) as illustrated in Fig. 5.3 for $N = 8$ quantisation intervals. This equivalent view involves the following steps at the transmitter and receiver.

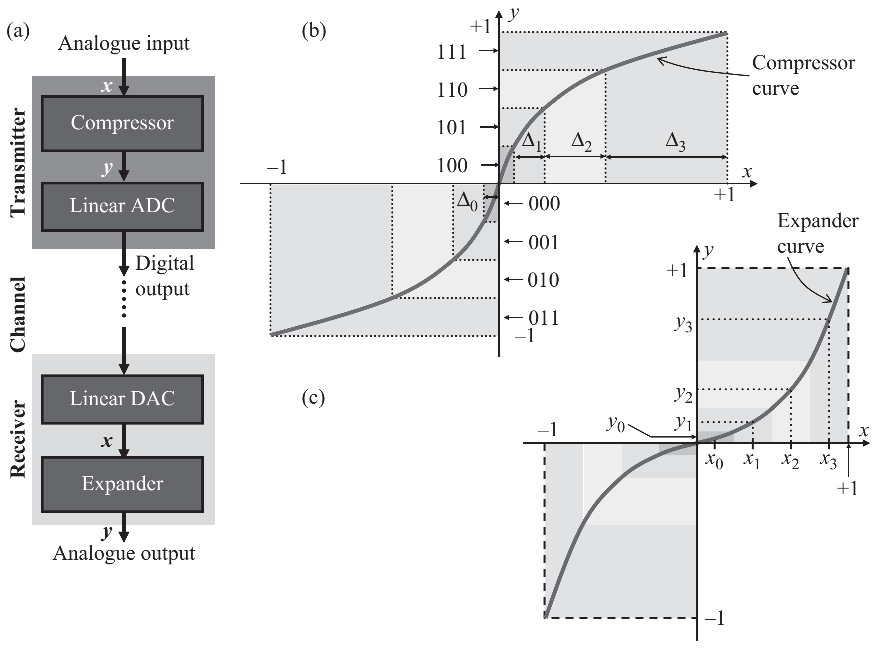


Fig. 5.3 Non-uniform quantisation: (a) equivalent steps; (b) compressor; (c) expander

- At the transmitter, first use a compressor to compress the analogue input samples in order to reduce high sample values relative to low sample values. Next, quantise the compressed signal using a uniform quantiser (i.e. linear ADC). Notice that Fig. 5.3b embodies these two steps. The input x is first converted into output y through the compressor curve and then the output y is divided into numbered uniform intervals. The end result is that we have achieved non-uniform quantisation of the input x by using progressively larger step sizes $\Delta_0, \Delta_1, \ldots$, as shown along the x-axis of the graph.
- At the receiver, recover the signal using a linear DAC (digital to analogue converter). Then expand the recovered signal using an expander having a transfer characteristic that is the exact inverse of the compressor. This perfectly removes the distortion due to compression at the transmitter. Notice in Fig. 5.3c how the expander curve maps equal intervals along the x-axis (expander input) to unequal intervals along the y-axis (expander output) that are progressively larger the further you are from the origin, just like the intervals along the x-axis of Fig. 5.3b.

This combined process of signal compression at the transmitter and expansion at the receiver is known as *companding* and does deliver tangible benefits the most important of which is the so-called *companding gain* defined in Section 5.4.5.

5.4.1 Ideal log-companding

To determine the function $y = f(x)$ that gives the required compressor characteristic, consider Fig. 5.4 which shows a compressor curve that maps an input interval of size Δ_x centred at x along the horizontal axis into its corresponding output interval along the vertical axis. The input and output are both normalised to the range

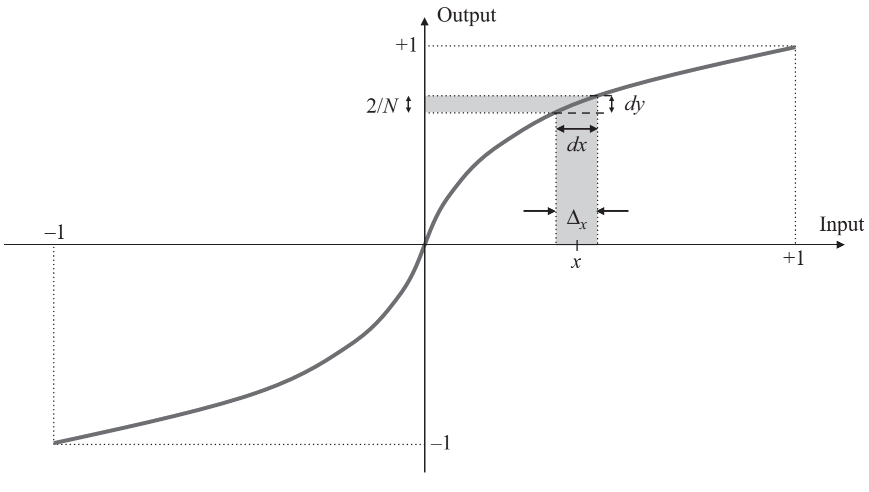


Fig. 5.4 Compressor characteristic, y = f(x). The range $-1 \to +1$ *along the y-axis is divided into N equal quantisation intervals, so each interval is of size 2/N as shown. The range* $-1 \to +1$ *along the x-axis is divided into N unequal quantisation intervals. One interval of size* Δ_x *centred at x is shown and is mapped by the compressor curve into an interval of size 2/N along the y-axis*

$(-1, +1)$, and the quantiser has N intervals, which means that the step size along the vertical axis is $2/N$. For large N, the intervals $2/N$ and Δ_x represent small changes denoted by dy and dx, respectively and their ratio gives the slope of the compressor curve at x. That is

$$\frac{dy}{dx} = \frac{2/N}{\Delta_x} \tag{5.13}$$

Or

$$\Delta_x = \frac{2/N}{\text{Slope of compressor curve}} = \frac{2}{N}\frac{dx}{dy} \tag{5.14}$$

If the input signal has probability density function $p(x)$, then the probability that its sample falls in an interval of size dx centred at x is $p(x)dx$, and its mean square value or signal power is obtained by summing the product of this probability and x^2 over the entire input range (–1, 1):

$$P_s = \int_{-1}^{1} x^2 p(x)dx \tag{5.15}$$

Eq. (5.6) gives the quantisation noise power in a uniform quantiser where step sizes are equal. To apply this to a non-uniform quantiser we multiply $\Delta_x^2/12$ by the probability $p(x)dx$ that the input sample falls in the interval x of step size Δ_x and sum this product over the entire input range to obtain

$$\begin{aligned}\text{MSQE} &= \int_{-1}^{1} \frac{\Delta_x^2}{12} p(x)dx \\ &= \frac{1}{3N^2}\int_{-1}^{1}\left(\frac{dx}{dy}\right)^2 p(x)dx \end{aligned} \tag{5.16}$$

where we made use of Eq. (5.14) for Δ_x. The ratio between Eqs. (5.15) and (5.16) yields

$$\text{SQNR} = \frac{P_s}{\text{MSQE}} = 3N^2 \frac{\int_{-1}^{1} x^2 p(x)dx}{\int_{-1}^{1}\left(\frac{dx}{dy}\right)^2 p(x)dx} \tag{5.17}$$

Since our goal is to achieve an SQNR that is independent of input level x, the RHS of the above equation must be independent of x, and this can only be the case if in the denominator

$$\left(\frac{dx}{dy}\right)^2 = K^2 x^2 \tag{5.18}$$

where K is some constant, so that the integrals cancel out leaving

$$\text{SQNR} = \frac{3N^2}{K^2} \tag{5.19}$$

which is independent of input signal level x as desired. Thus the compressor characteristic must be a function $y = f(x)$ that satisfies Eq. (5.18). Taking the square root of both sides of this equation, and making dy the subject of the equation and integrating yields

$$y = \frac{1}{K}\ln(x) + Y_1$$

where Y_1 is a constant that we choose in order to make $(x, y) = (1, 1)$ a point on the curve, as is obvious in Fig. 5.4, since the normalised maximum input is compressed to the normalised maximum output. Thus $Y_1 = 1$, and the desired compressor characteristic is

$$y = \frac{1}{K}\ln(x) + 1 \tag{5.20}$$

Taking the derivative with respect to x gives the slope of the compressor curve as

$$\frac{dy}{dx} = \frac{1}{Kx} \equiv \frac{\text{Constant interval (along } y\text{-axis)}}{\text{Quantiser step size (along } x\text{-axis)}} \tag{5.21}$$

Eq. (5.20) gives a complete specification of a compressor curve that can be used to compress the input signal x to give an output y, which when uniformly quantised achieves the desired fine quantisation of small input values and coarse quantisation of larger input values, and delivers an SQNR that is exactly the same across all input sample amplitudes, no matter how small. If this sounds too good to be true, it really is, because there is a practical problem with implementing this compression function. Its slope is infinite at $x = 0$, so the required quantiser step sizes become vanishingly small near the origin (as $x \to 0$), which means that you need an infinite number of intervals to be able to implement the curve down to $x = 0$. To circumvent this problem, we are in practice forced to abandon the logarithmic function in the region $x \to 0$ and to adopt a linear function in this region, say $|x| \leq 1/A$, where A is some constant ≥ 1. The effect of this pragmatic solution is that SQNR is constant in the region $1/A \leq |x| \leq 1$ where the logarithmic compression curve is followed, whereas in the region $-1/A \leq x \leq 1/A$ where uniform quantisation is followed, SQNR decreases with input amplitude as discussed in Section 5.3. Notice therefore that the constant A sets the boundaries of the logarithmic and linear compression regions. For example, if $A = 1$ the entire curve is linear and there is no compression (which corresponds to uniform quantisation) and if $A = \infty$ the compression curve is entirely logarithmic down to $x = 0$ (which is not feasible). Because the compression curve is a logarithmic function, digitisation based on non-uniform quantisation is sometimes referred to as *log-PCM*.

Worked Example 5.3: Logarithmic Versus Linear Compression

Given a quantiser, of normalised range (–1, +1), that has $N = 256$ quantisation intervals, we wish to determine SQNR in the following cases.

(a) The intervals are uniform (which corresponds to a linear compressor) and the input is a sinusoid of amplitude 1 V.

(b) The quantiser incorporates a logarithmic compressor having $K = 1 + \ln(A)$, where $A = 87.6$, so that the intervals are non-uniform, and the input is a sinusoid of amplitude 1 V.

(c) The quantiser is linear as in (a) and the input is a sinusoid of amplitude 10 mV.
(d) The quantiser is logarithmic as in (b) and the input is a sinusoid of amplitude 10 mV.

(a) The sinusoid fully loads this linear quantiser that has $N = 256$ intervals which corresponds to $k = \log_2 N = 8$ bits/sample, so SQNR is as given by Eq. (5.12):

$$\text{SQNR} = 6.02k + 1.76 = 6.02 \times 8 + 1.76 = 49.92 \text{ dB}$$

(b) The SQNR of this logarithmic quantiser is given by Eq. (5.19) irrespective of input signal amplitude. Thus

$$\text{SQNR} = \frac{3N^2}{K^2} = \frac{3 \times 256^2}{(1 + \ln 87.6)^2} = 6564.2 = 38.2 \text{ dB}$$

(c) We apply Eq. (5.10) with $A_p = 10$ mV, $C = 1$ V and $R = \sqrt{2}$ (for a sinusoid) to obtain

$$\begin{aligned}\text{SQNR} &= 4.77 + 6.02k - 20\log(R) + 20\log(A_p/C) \text{ dB} \\ &= 4.77 + 6.02 \times 8 - 10\log(2) + 20\log(10/1000) \text{ dB} \\ &= 49.92 - 40 = 9.92 \text{ dB}\end{aligned}$$

Notice that this 10 mV sinusoid underloads the 1 V quantiser by 40 dB. Hence its SQNR is down by 40 dB from the value in (a) for an input that fully loads the quantiser.

(d) As noted in (b) the SQNR of the logarithmic quantiser is independent of input signal level, so SQNR for this 10 mV sinusoidal input is 38.2 dB, exactly the same as for the 1 V sinusoid in (b).

It can be seen that SQNR in (a) is higher than in (b). In general, for input signals at the top end, i.e. for inputs that approximately fully load the quantiser, a linear quantiser will always deliver a higher SQNR than a log-quantiser of the same number of bits per sample. However, the SQNR of log-quantisers might be modest but they are consistent for all input amplitudes, as can be seen by comparing the results in (c) and (d) where the linear quantiser has a very poor SQNR of 9.92 dB for the 10 mV sinusoid whereas the SQNR of the log-quantiser is still 38.2 dB.

The ITU-T has standardised two practicable logarithmic compressor characteristics, the A-law in Europe, and the μ-law in North America and Japan.

5.4.2 A-law companding

The A-law compressor characteristic is based on Eq. (5.20) with the following settings and modifications:

- The constant $K = 1 + \ln(A)$, with $A = 87.6$, so that the logarithmic compressor function takes the form

$$y_{\log} = \frac{1 + \ln(Ax)}{1 + \ln(A)} \tag{5.22}$$

- The logarithmic function $y_{\log}$ is followed down to $|x| = 1/A$ at which point it is abandoned and a linear characteristic $y_{\text{lin}} = mx + c$ is followed down to $x = 0$. The linear characteristic y_{lin} must satisfy two conditions. First, it must pass through the origin in order that the compressor characteristic retains its odd symmetry. This means that $c = 0$. Second, y_{lin} must be equal to $y_{\log}$ at $x = 1/A$ in order to maintain continuity, and this means that

$$m = \frac{A}{1 + \ln(A)}$$

which gives

$$y_{\text{lin}} = \frac{Ax}{1 + \ln(A)} \tag{5.23}$$

The A-law compressor characteristic may therefore be expressed as follows:

$$y = \begin{cases} \dfrac{1 + \ln(A|x|)}{1 + \ln(A)} \operatorname{sgn}(x), & \dfrac{1}{A} \le |x| \le 1 \\ \dfrac{Ax}{1 + \ln(A)}, & -\dfrac{1}{A} \le x \le \dfrac{1}{A} \end{cases} \tag{5.24}$$

The signum function expresses the odd symmetry feature of the compressor. To determine compressor output, apply the above function to the magnitude (i.e. absolute value) of the input, and then assign to the output the sign of the input. Subsequently we will restrict our analysis to the positive input range of a logarithmic compressor knowing that a negative input is processed exactly as if it were positive to obtain an output that is then negated.

5.4.3 *μ-Law companding*

The μ-law compressor is also based on Eq. (5.20) with the following settings and modifications:

- The constant K is set to $\ln(1 + \mu)$, where μ is a positive constant usually 255. The logarithmic compressor function therefore takes the form

$$y = \frac{\ln(x)}{\ln(1 + \mu)} + 1 = \frac{\ln(x) + \ln(1 + \mu)}{\ln(1 + \mu)} = \frac{\ln(x + \mu x)}{\ln(1 + \mu)}$$

- Next, the function is modified by replacing $\ln(x + \mu x)$ in the numerator by $\ln(1 + \mu x)$. This gives the μ-law compressor, which we express with the aid of the signum function (see the comments following Eq. (5.24)) as

$$y = \frac{\ln(1 + \mu|x|)}{\ln(1 + \mu)} \operatorname{sgn}(x), \quad -1 \le x \le 1 \tag{5.25}$$

The modification carried out in the second step above is crucial in producing a practicable logarithmic compressor. To see this, note that in the region $|x| \to 1$, the term $1 + \mu|x| \approx \mu|x|$, since $\mu \gg 1$ so that the compressor function is logarithmic

$$y_{\log} = \frac{\ln(\mu|x|)}{\ln(1 + \mu)} \operatorname{sgn}(x), \quad |x| \to 1 \tag{5.26}$$

And in the region $|x| \to 0$, the term $\ln(1 + \mu|x|) \approx \mu|x|$ so that the compressor function is linear

$$y_{\text{lin}} = \frac{\mu x}{\ln(1+\mu)}, \qquad |x| \to 0 \tag{5.27}$$

Therefore the μ-law compressor of Eq. (5.25) exhibits a gradual transition from a logarithmic function $y_{\log}$ in Eq. (5.26) in the region $|x| \to 1$ to a linear function y_{lin} in Eq. (5.27) in the region $|x| \to 0$. The A-law compressor on the other hand has a sudden transition from logarithmic to linear at the point $|x| = 1/A$. Fig. 5.5 shows the A-law and μ-law compressor characteristics for various values of A and μ. At $A = 1$ and $\mu \to 0$, both curves are linear and pass through the origin (0, 0) as well as the saturation points $(-1, -1)$ and (1, 1), so that input x equals output y at all points and there is no compression, which means that quantisation is uniform or linear. As the positive constants A and μ increase, compression increases until there is 'infinite compression' at A, $\mu = \infty$, whereby all non-zero input values are compressed into one output interval. Compression that is suitable for speech digitisation lies between these two extremes of zero compression at one end and infinite compression at the other.

5.4.4 Specification of companding

It can be seen from Fig. 5.5 that the A-law and μ-law compressor curves are almost identical at their standardised values of $A = 87.6$ and $\mu = 255$. With the aid of the horizontal and vertical graph lines, you should be able to read these two standard curves in Fig. 5.5 to observe the following approximate mappings: The top half ($0.5 \to 1$) of input is compressed into the top 1/8th of output ($0.875 \to 1$); the next ¼ of input ($0.25 \to 0.5$) is compressed into the next 1/8th of output ($0.75 \to 0.875$); the next 1/8th of input ($0.125 \to 0.25$) is mapped into the next 1/8th of output ($0.625 \to 0.75$); the next 1/16th of input ($0.0625 \to 0.125$) is expanded into the next 1/8th of

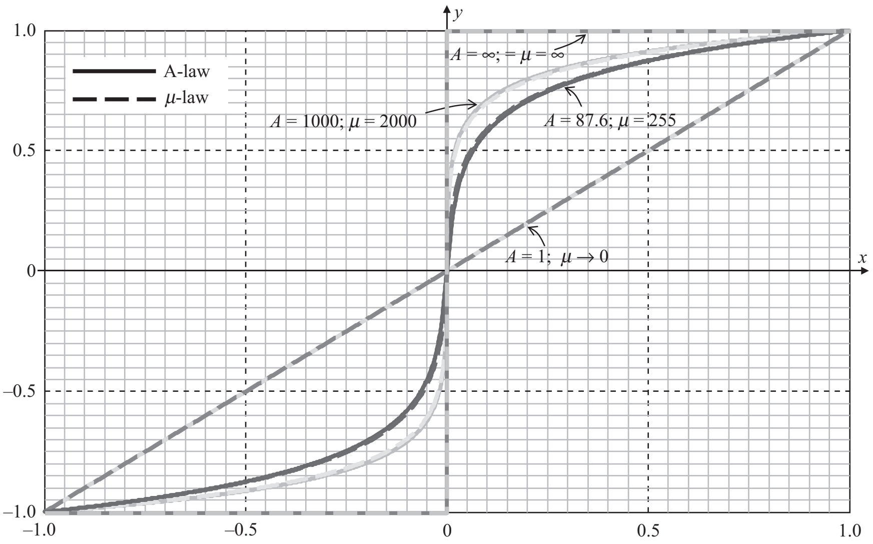


Fig. 5.5 A-law and μ-law compression curves at various values of A and μ

output (0.5 → 0.625); and the bottom 1/16th of input (0 → 0.0625) is expanded into the bottom half of output (0 → 0.5). These curves therefore map input intervals whose sizes reduce progressively by a factor of 2 (starting at the top end of input) into equal intervals at the output, and this achieves the desired compression of large input values and expansion of the smaller ones.

The amount of expansion of small input values and compression of large input values in each law is controlled by the constants A and μ in the respective laws. These constants can be set to achieve any specified compression as follows. If you want to

> *compress the top half of input* $0.5 \le x \le 1$ *into the top* $(1/2m)^{\text{th}}$ *of output* $(2m-1)/2m \le y \le 1$ *while expanding the bottom* $(1/2^m)^{\text{th}}$ *of input* $0 \le x \le 1/2^m$ *into the bottom half of output* $0 \le y \le 0.5$

then the value of A to use in the A-law expression of Eq. (5.24) is

$$A = 2^{2m}/e; \quad m \ge \log_2 e, \quad e = 2.718281828459\cdots \tag{5.28}$$

and the value of μ to use in Eq. (5.25) is

$$\mu = 2^{2m} - 2^{m+1}; \qquad m > 1 \tag{5.29}$$

Eq. (5.28) is derived in the next worked example and gives an exact match of the compression and expansion stated above, whereas Eq. (5.29) only gives an approximate match that improves with m, the maximum error being less than 0.25% for $m \ge 4$ and no more than 3.75% at any $m > 1$.

An indication of the extent of compression is given by the ratio between the maximum step size $\Delta_{\max}$, which occurs at $x = 1$, and the minimum step size $\Delta_{\min}$ that occurs at $x = 0$. Using Eq. (5.14) we obtain

$$\frac{\Delta_{\max}}{\Delta_{\min}} = \left(\left.\frac{dy}{dx}\right|_{x=0}\right) \Big/ \left(\left.\frac{dy}{dx}\right|_{x=1}\right) \tag{5.30}$$

For the A-law, the applicable curves are Eq. (5.23) at $x = 0$ and Eq. (5.22) at $x = 1$. Taking their derivatives we obtain

$$\left.\frac{dy}{dx}\right|_{x=0} = \frac{A}{1+\ln A}; \quad \left.\frac{dy}{dx}\right|_{x=1} = \left.\frac{1/x}{1+\ln A}\right|_{x=1} = \frac{1}{1+\ln A} \tag{5.31}$$

Similarly, taking the derivative of the μ-law function in Eq. (5.25) written simply as

$$y = \frac{\ln(1+\mu x)}{\ln(1+\mu)}$$

we obtain

$$\frac{dy}{dx} = \frac{\mu}{1+\mu x} \cdot \frac{1}{\ln(1+\mu)}$$

and hence

$$\left.\frac{dy}{dx}\right|_{x=0} = \frac{\mu}{\ln(1+\mu)}; \quad \left.\frac{dy}{dx}\right|_{x=1} = \frac{\mu}{(1+\mu)\ln(1+\mu)} \tag{5.32}$$

Therefore the ratio between maximum and minimum step sizes is

$$\frac{\Delta_{\max}}{\Delta_{\min}} = \begin{cases} A, & \text{A-law} \\ 1+\mu, & \mu\text{-law} \end{cases} \tag{5.33}$$

If $\Delta_{\max} = \Delta_{\min}$, it means that step sizes are equal across the entire input range, which is the special case of uniform quantisation. We see from Eq. (5.33) that this happens in A-law when $A = 1$, and in μ-law when $\mu = 0$. Clearly, the constant A cannot be less than +1 and μ cannot be less than 0, otherwise $\Delta_{\max}$ would be less than $\Delta_{\min}$, which is contradictory.

Worked Example 5.4: A-Law Specification

We wish to derive Eq. (5.28), giving an expression for the constant A in terms of m so that the A-law compressor will compress the top half of input into the top $(1/2m)^{\text{th}}$ of output while expanding the bottom $(1/2^m)^{\text{th}}$ of input to fill the bottom half of output.

The top $(1/2m)^{\text{th}}$ of output starts at $y = 1 - 1/2m = (2m - 1)/2m$, so the compressor specification given above corresponds to $y = (2m - 1)/2m$ at $x = 1/2$, and $y = 1/2$ at $x = 1/2^m$. Let us set the A-law function (logarithmic portion) to the second condition and solve for A:

$$\frac{1}{2} = \frac{1 + \ln(A/2^m)}{1 + \ln(A)}$$

$$1 + \ln(A) = 2 + 2\ln(A) - \ln(2^{2m})$$

$$\ln A = \ln(2^{2m}) - 1 = \ln(2^{2m}) - \ln e = \ln(2^{2m}/e)$$

$$A = 2^{2m}/e$$

Next, we verify that this value for A satisfies the first condition. That is, we verify that if A is set to $2^{2m}/e$ in the A-law function (logarithmic portion) then y will be $(2m - 1)/2m$ at $x = 1/2$. Substituting $A = 2^{2m}/e$ and $x = 1/2$ into the A-law yields

$$\begin{aligned} y &= \frac{1 + \ln(0.5 \times 2^{2m}/e)}{1 + \ln(2^{2m}/e)} = \frac{1 + \ln(2^{2m-1}/e)}{1 + \ln(2^{2m}/e)} \\ &= \frac{1 + (2m-1)\ln 2 - \ln e}{1 + 2m\ln 2 - \ln e} = \frac{1 + (2m-1)\ln 2 - 1}{1 + 2m\ln 2 - 1} \\ &= \frac{(2m-1)\ln 2}{2m\ln 2} = \frac{2m-1}{2m} \end{aligned}$$

Therefore the value $A = 2^{2m}/e$ ensures that the A-law function compresses the (normalised) input range $1/2 \le x \le 1$ into the output range $(2m - 1)/2m \le y \le 1$ while expanding the bottom end input range $0 \le x \le 1/2^m$ into the output range $0 \le y \le 1/2$. Since the above derivation assumes that x lies in the logarithmic portion of the A-law function, it is essential that the top of this bottom input range, i.e. $x = 1/2^m$, lies within this logarithmic portion that only extends down to $x = 1/A$. This means that we must have

$$\frac{1}{2^m} \ge \frac{1}{A} = \frac{e}{2^{2m}}$$

which upon manipulation as follows yields the restriction on m stated in Eq. (5.28):

$$\begin{aligned}
&2^m \leq 2^{2m}/e \\
&\log_2(2^m) \leq \log_2(2^{2m}/e) = \log_2(2^{2m}) - \log_2 e \\
&m \log_2 2 \leq 2m \log_2 2 - \log_2 e \\
&m \leq 2m - \log_2 e \qquad\qquad (\text{since } \log_2 2 = 1) \\
&m \geq \log_2 e
\end{aligned}$$

5.4.5 *Companding gain and penalty*

The benefit of non-uniform quantisation is measured in terms of the improvement that it delivers to the SQNR of small samples. *Companding gain* G_c is one such measure defined as the ratio between the SQNR of small input values in a non-uniform quantiser and the SQNR of the same signal type when using a uniform quantiser of the same number of bits/sample. A non-uniform quantiser employs the smallest step size $\Delta_{\min}$ in the region around $x = 0$. The SQNR of the non-uniform quantiser in this region is therefore

$$\frac{P_s}{\text{MSQE}} = \frac{P_s}{\Delta_{\min}^2/12}$$

A uniform quantiser would employ a constant step size of $2/N$ (see Fig. 5.4), where N is the number of uniform intervals, so that the SQNR of the uniform quantiser in this small input region is

$$\frac{P_s}{(2/N)^2/12}$$

Dividing the former SQNR by the latter yields companding gain as

$$G_c = \left(\frac{2/N}{\Delta_{\min}}\right)^2 = \left(\left.\frac{dy}{dx}\right|_{x=0}\right)^2$$

where we have used Eq. (5.13) to express the gain as the square of the derivative of the compressor curve evaluated at the origin. These derivatives were computed in Eqs. (5.31) and (5.32) for the A-law and μ-law curves, so we may write

$$G_c = \begin{cases} \left(\dfrac{A}{1 + \ln A}\right)^2, & \text{A-law} \\ \left(\dfrac{\mu}{\ln(1+\mu)}\right)^2, & \mu\text{-law} \end{cases}$$

In dB these expressions become

$$G_c = \begin{cases} 20 \log_{10}\left(\dfrac{A}{1 + \ln A}\right) \text{dB}, & \text{A-law} \\ 20 \log_{10}\left(\dfrac{\mu}{\ln(1+\mu)}\right) \text{dB}, & \mu\text{-law} \end{cases} \tag{5.34}$$

Putting the standard values $A = 87.6$ and $\mu = 255$ into the above equations yields $G_c = 24.1$ dB for A-law and 33.3 dB for μ-law. Since one extra bit per sample yields a 6 dB improvement in SQNR, it follows that a 24 dB improvement due to companding is equivalent to an extra four coding bits. What this means is that an 8-bit A-law quantiser achieves the same SQNR for small inputs as a 12-bit uniform quantiser. Thus by using A-law quantisation the required number of bits per sample is reduced from 12 to 8, which represents a 33% reduction in bit rate and the same percentage reduction in bandwidth. In practice a piecewise linear approximation is adopted in the implementation of the A-law and μ-law standards, as discussed in Section 5.5.1. This leaves the A-law curve unchanged in the region around $x = 0$ where the law is already governed by a linear curve, so that practical A-law companding gain remains at 24 dB as computed above. However, the piecewise linear approximation alters the μ-law curve in the region around $x = 0$ so that practical μ-law companding gain is 30 dB, rather than the 33.3 dB obtained above, which corresponds to an extra five coding bits. That is, an 8-bit μ-law quantiser achieves the same SQNR as a 13-bit uniform quantiser in the small signal region, which represents a saving of 38.5% in bit rate and transmission bandwidth for the same quality of small input signals.

It is important to recognise that the improvement in SQNR enjoyed by the small input samples is at the expense of the SQNR of large input samples. There really is no free lunch! Let us therefore define *companding penalty* L_c as the ratio between the SQNR of large input values in a uniform quantiser and the SQNR of the same signal type when using a non-uniform quantiser of the same number of bits per sample. A non-uniform quantiser uses the largest step size $\Delta_{\max}$ at the top end of the input range, whereas a uniform quantiser uses the step size $2/N$ in this region. By a similar argument as for G_c, we obtain

$$\begin{aligned} L_c &= \left(\frac{\Delta_{\max}}{2/N}\right)^2 = \left(\frac{1}{dy/dx}\bigg|_{x=1}\right)^2 \\ &= \begin{cases} (1+\ln A)^2, & \text{A-law} \\ \left[\dfrac{(1+\mu)\ln(1+\mu)}{\mu}\right]^2, & \mu\text{-law} \end{cases} \end{aligned}$$

where we have made use of Eqs. (5.31) and (5.32) for dy/dx evaluated at $x = 1$. Expressed in dB, the companding penalty is

$$L_c\ (\text{dB}) = \begin{cases} 20\log_{10}(1+\ln A), & \text{A-law} \\ 20\log_{10}\left[\dfrac{(1+\mu)\ln(1+\mu)}{\mu}\right], & \mu\text{-law} \end{cases} \tag{5.35}$$

Again substituting $A = 87.6$ and $\mu = 255$ into the above expressions yields $L_c = 14.8$ dB for A-law and 14.9 dB for μ-law. Thus A-law and μ-law quantisers sacrifice signal quality at the top end of input by ~15 dB (and by progressively lower than ~15 dB as you go down the input range) in order to improve the quality of small input signals by 24 dB (for A-law) and by progressively lower as you go up the input range. The end result of this exchange is a lower but consistent SQNR across the entire input range that is more satisfying in communication services than the situation that obtains in uniform quantisation where SQNR is very high at the top end and poor at the bottom end. Note that a piecewise linear approximation of the two laws leads to a lower $\Delta_{\max}$ than the one used above and hence to a companding penalty of 12 dB, which is smaller than the L_c given by Eq. (5.35).

Worked Example 5.5: Quantised Outputs

A sinusoidal signal $x(t) = 5\cos(2\pi \times 250t + \pi/6)$ V, sampled at 8 kHz starting at $t = 0$, fully loads an eight-bit μ-law quantiser ($\mu = 255$). Determine the first 10 quantised output values and calculate SQNR based on the first 10 samples.

The quantiser is described as eight-bit, so it has $N = 2^8 = 256$ quantisation intervals. Sampling frequency $f_s = 8$ kHz, so sampling interval $T_s = 1/f_s = 125$ µs. The steps to follow in solving this problem are shown in Fig. 5.6 and explained below.

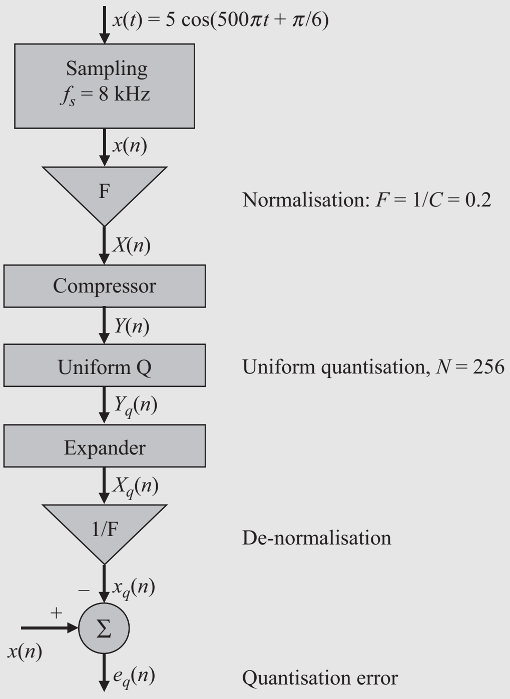


Fig. 5.6 Quantisation of a sinusoidal signal (Worked Example 5.5)

(i) Obtain the values of $x(t)$ at $t = 0, T_s, 2T_s, 3T_s, \ldots, 9T_s$. These are the first 10 samples, denoted by $x(n)$, $n = 0, 1, 2, \ldots, 9$, in the third column of Table 5.1. For example, $x(0)$ is obtained by substituting $t = 0$ into the expression for $x(t)$, so $x(0) = 5\cos(2\pi \times 250 \times 0 + \pi/6) = 5\cos(\pi/6) = 4.3301$; $x(1)$ is obtained by evaluating $x(t)$ at $t = T_s = 125$ µs, so $x(1) = 5\cos(2\pi \times 250 \times 125 \times 10^{-6} + \pi/6) = 5\cos(0.7199) = 3.7592$; $x(2)$ is obtained by evaluating $x(t)$ at $t = 2T_s = 250$ µs; and so on.

(ii) Since the given compressor function has range ±1, normalise the input signal (of amplitude $C = 5$ V) to this range by multiplying each sample $x(n)$ by the factor $F = 1/C = 0.2$. The result is the normalised input denoted by $X(n)$ in the fourth column of Table 5.1. For example, $X(0) = 0.2x(0) = 0.866$. Retain only the absolute value $|X(n)|$. The sign of $x(n)$ will be assigned to the quantised output at the end.

(iii) Determine compressed output $|Y(n)|$ by using $|X(n)|$ as input to the μ-law function of Eq. (5.25). These results are listed in column 5 of Table 5.1. For example

$$|Y(0)| = \frac{\ln(1 + 255|X(0)|)}{\ln(1 + 255)} = 0.9742$$

(iv) Uniformly quantise $|Y(n)|$ to obtain the normalised quantised output $|Y_q(n)|$ by noting that $|Y(n)|$ falls in one of the $N/2$ uniform intervals of size $\Delta = 2/N = 2/256 = 7.8125 \times 10^{-3}$ that covers the positive quantiser range (0, 1). Numbering these intervals from $j = 0$ to $N/2 - 1$, the sample $|Y(n)|$ falls in interval $j = \lfloor |Y(n)|/\Delta \rfloor$ and is quantised to the midpoint of that interval which is $j\Delta + \Delta/2$, where $\lfloor z \rfloor$ represents the largest integer less than or equal to z, subject to a maximum of $N/2 - 1$. For example, 0.9742 falls in interval $j = \lfloor 0.9742/7.8125 \times 10^{-3} \rfloor = \lfloor 124.69 \rfloor = 124$ and is quantised to $124 \times 7.8125 \times 10^{-3} + 7.8125 \times 10^{-3}/2 = 0.9727$. All the normalised quantised outputs are obtained in this way and are listed in column 6 of Table 5.1.

Table 5.1 Tabulated results (Worked Example 5.5)

n	$t = nT_s$ (μs)	$x(n)$ (V)	$\|X(n)\|$	$\|Y(n)\|$	$\|Y_q(n)\|$	$y_q(n)$ (V)	$X_q(n)$	$x_q(n)$(V)	$e_q(n)$ (mV)
0	0	4.3301	0.8660	0.9742	0.9727	4.8633	0.8588	4.2938	36.3374
1	125	3.7592	0.7518	0.9488	0.9492	4.7461	0.7536	3.7681	−8.8999
2	250	3.0438	0.6088	0.9109	0.9102	4.5508	0.6061	3.0304	13.3824
3	375	2.2114	0.4423	0.8538	0.8555	4.2773	0.4465	2.2326	−21.1326
4	500	1.2941	0.2588	0.7583	0.7617	3.8086	0.2639	1.3195	−25.4535
5	625	0.3270	0.0654	0.5180	0.5195	2.5977	0.0660	0.3300	−2.9880
6	750	−0.6526	0.1305	0.6374	0.6367	−3.1836	−0.1300	−0.6500	−2.6605
7	875	−1.6072	0.3214	0.7968	0.7930	−3.9648	−0.3146	−1.5729	−34.2706
8	1000	−2.5000	0.5000	0.8757	0.8789	−4.3945	−0.5090	−2.5452	45.1536
9	1125	−3.2967	0.6593	0.9253	0.9258	−4.6289	−0.6613	−3.3065	9.7472

(v) Obtain the actual quantised output $y_q(n)$ by (a) denormalising $|Y_q(n)|$ through division by the normalising factor F, and (b) assigning to the result the sign of the input sample $x(n)$. For example, the first sample yields output $y_q(0) = |Y_q(0)|/F = 0.9727/0.2 = 4.8633$ V, and the last sample yields output $y_q(9) = -|Y_q(9)|/F = -0.9258/0.2 = -4.6289$ V. These are the desired quantised outputs and are listed in column 7 of Table 5.1.

(vi) The normalised quantised output $Y_q(n)$ is usually digitally encoded as discussed in Section 5.5. When recovered from its digital format, $Y_q(n)$ must be expanded to remove the distortion introduced by the compressor and yield $X_q(n)$. The μ-law expander is derived by making x the subject of Eq. (5.25). After swapping variables to retain the conventional use of x for input and y for output, this gives the *μ-law expander*

$$y = \frac{1}{\mu}\left[(1+\mu)^{|x|} - 1\right]\mathrm{sgn}(x), \quad -1 \le x \le 1 \tag{5.36}$$

Now substitute $Y_q(n)$ for x in the above equation to obtain the expanded values $X_q(n)$. For example, the last sample yields

$$X_q(9) = \frac{1}{255}\left[(1+255)^{0.9258} - 1\right]\mathrm{sgn}(x(9)) = -0.6613$$

where we have used the fact that $Y_q(n)$ has the same sign as $x(n)$. The full set of values of $X_q(n)$ are in column 8, with the denormalised values $x_q(n) = X_q(n)/F$ in column 9 of Table 5.1.

(vii) Calculate the quantisation error $e_q(n) = x(n) - x_q(n)$ as listed in column 10, being careful to minimise rounding errors. Note that although Table 5.1 lists values rounded to four decimal places, the calculations were performed to greater accuracy.

(viii) The quantisation noise power MSQE, signal power P_s, and finally SQNR are computed as follows using values from relevant columns of Table 5.1:

$$\mathrm{MSQE} = \frac{1}{10}\sum_{n=0}^{9} e_q^2(n) = 5.9975 \times 10^{-4}\ \mathrm{W}$$

$$P_s = \frac{1}{10}\sum_{n=0}^{9} x^2(n) = 6.8946\ \mathrm{W}$$

$$\mathrm{SQNR} = \frac{P_s}{\mathrm{MSQE}} = 1.1496 \times 10^4 = 40.6\ \mathrm{dB}$$

Exercise: Compare the above SQNR with that which would be obtainable using A-law by solving the above problem for the case of an A-law quantiser with $A = 87.6$, given that the *A-law expander* is

$$y = \begin{cases} \dfrac{1}{A}\exp[(1+\ln A)|x| - 1]\mathrm{sgn}(x), & |x| \ge \dfrac{1}{1+\ln A} \\ \dfrac{1+\ln A}{A}|x|\mathrm{sgn}(x), & |x| \le \dfrac{1}{1+\ln A} \end{cases} \tag{5.37}$$

5.5 PCM

ITU-T [1] defines pulse code modulation (PCM) as

a process in which a signal is sampled, and each sample is quantised independently of other samples and converted by encoding into a digital signal.

The recommended nominal sampling rate is 8000 samples per second (i.e. 8 kHz) with a tolerance of 50 parts per million (ppm), each sample being then quantised using eight bits, which yields a digital signal at a nominal bit rate of 64 kb/s. We illustrated these steps of the PCM process in Fig. 1.6 and have discussed in detail the steps of filtering and sampling (Chapter 4) and quantisation (Sections 5.3 and 5.4). The stage is now set for a discussion of the final step of encoding based on standardised log-quantisers, which leads to digital signals referred to as A-law PCM and μ-law PCM. It is worth noting here again that this description of a base band digital signal as PCM is a regrettable historical misnomer for reasons given in Section 4.7 in the context of the so-called *pulse amplitude modulation*. A more appropriate terminology would be to use ADC in place of PCM so that we have A-law ADC, μ-law ADC, linear ADC (instead of linear PCM), differential ADC (instead of DPCM) and so on. However, current literature seems wedded to PCM without much hope of divorce, so the terminology will be sparingly used in this book for the sole purpose of maintaining continuity with the large body of literature on the subject. But it must be made absolutely clear that PCM is simply a source coding technique by which an analogue signal is presented in digital format as a bit stream without loss of any significant information. Contrary to what the term PCM misleadingly suggests, modulation is not at all involved in this source coding process but takes place elsewhere further down the signal processing chain of a communication system, as Fig. 1.13 makes clear.

The log-quantisation process discussed in the previous section is based on a continuous variation of quantisation step size Δx with input x, from a maximum $\Delta_{\max}$ at $x = 1$ (normalised) to a minimum $\Delta_{\min}$ at $x = 0$, each quantisation interval having a unique step size. Practical implementations however use a piecewise linear approximation of the A-law and μ-law compression curves. In what follows we introduce this approximation, discuss the encoding rule (i.e. procedure for assigning a unique set of 8 bits to each of the 256 quantised levels) and evaluate the SQNR of the digital signal.

5.5.1 A-law and μ-law PCM

The quantisation that produces A-law PCM is specified through piecewise linear approximation of the A-law characteristic of Eq. (5.24). This approximation is shown in Fig. 5.7 where the dashed-curve is Eq. (5.24) and the solid lines labelled s_0 to s_7 are the approximations used. Only the positive range of the characteristic is shown. The negative range is similar except for a negative sign, as earlier discussed. Note that the input is normalised to $\pm C$, where $C = 4096$ in order to render all step sizes as integers. The output is divided into 256 equal intervals, 128 of which are in the positive range as shown numbered along the y-axis of Fig. 5.7, with 16 intervals per segment s_0 to s_7. The piecewise linear approximation for A-law PCM actually comprises 13 line segments, namely one line passing through the origin made up of s_0 and s_1 which are collinear, and six distinct lines s_2 to s_7 above and below this one.

The linear approximation means that step sizes are held constant within each segment, and only change from segment to segment, rather than from interval to interval. The maximum step size $\Delta_{\max}$ is used within segment s_7 and is obtained by dividing the range of s_7 (2048, 4096) by the number of intervals (=16) into which s_7 is divided. Thus

$$\Delta_{\max} = \frac{4096 - 2048}{16} = 128$$

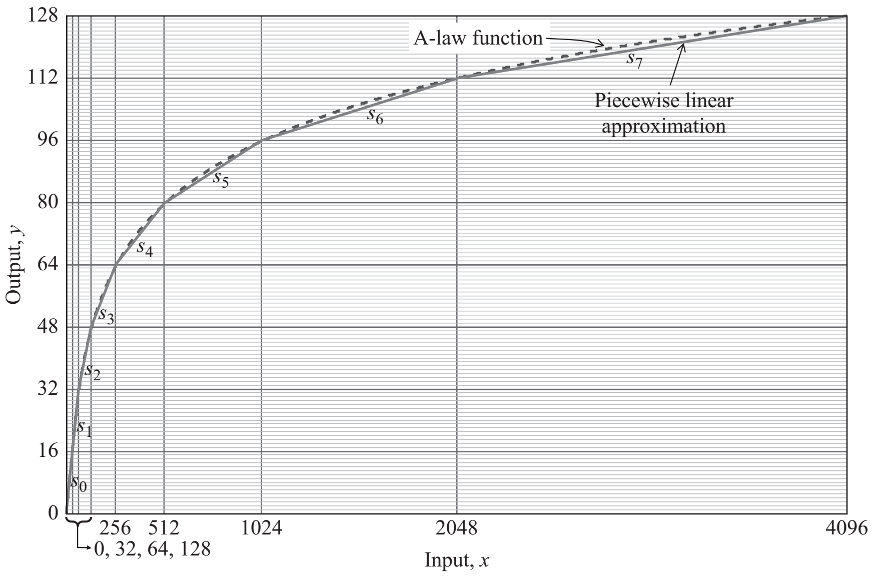


Fig. 5.7 Piecewise linear approximation of A-law compression function

The next step size Δ_6 used in segment s_6 is similarly obtained by dividing the range of s_6 (1024, 2048) by 16:

$$\Delta_6 = \frac{2048 - 1024}{16} = 64$$

and so on until the minimum step size $\Delta_{\min}$ is reached in segments s_0 and s_1, given by

$$\Delta_{\min} = \frac{64 - 32}{16} = \frac{32 - 0}{16} = 2$$

Observe that if the entire range (−4096, 4096) were uniformly quantised using step size $\Delta_{\max}$, the number of uniform intervals would be

$$N = \frac{2 \times 4096}{128} = 64$$

which requires $k = \log_2(64) = 6$ bits per sample. But if $\Delta_{\min}$ were used over the entire input range, the number of uniform intervals would be

$$N = \frac{2 \times 4096}{2} = 4096$$

which requires $k = \log_2(4096) = 12$ bits per sample. Thus in A-law PCM the largest input samples are coarsely quantised at a resolution of six bits/sample, whereas the smallest input samples are finely quantised at a resolution of 12 bits/sample. This means that, compared to eight-bit linear ADC, A-law PCM delivers a four-bit improvement at the bottom end, which, since each extra bit yields a 6 dB increase in SQNR, corresponds to a *companding gain* of 24 dB as discussed in Section 5.4.5. Notice however that this improvement is achieved at the expense of a two-bit shortage for the top end, which corresponds to a *companding penalty* of 12 dB as also earlier discussed.

Table 5.2 A-law coding and decoding specifications

Segment (s)	**Step size (Δ)**	**Segment interval (v)**	**Input range (X)**	**Quantised (compressor) output (Y_q)**	**Output code (8 bit)**	**Receiver (expander) output (X_q)**
0	2	0	0–2	0	1 000 0000	1
		↓	↓	↓	↓	↓
		15	30–32	15	1 000 1111	31
1	2	0	32–34	16	1 001 0000	33
		↓	↓	↓	↓	↓
		15	62–64	31	1 001 1111	63
2	4	0	64–68	32	1 010 0000	66
		↓	↓	↓	↓	↓
		15	124–128	47	1 010 1111	126
3	8	0	128–136	48	1 011 0000	132
		↓	↓	↓	↓	↓
		15	248–256	63	1 011 1111	252
4	16	0	256–272	64	1 100 0000	264
		↓	↓	↓	↓	↓
		15	496–512	79	1 100 1111	504
5	32	0	512–544	80	1 101 0000	528
		↓	↓	↓	↓	↓
		15	992–1024	95	1 101 1111	1008
6	64	0	1024–1088	96	1 110 0000	1056
		↓	↓	↓	↓	↓
		15	1984–2048	111	1 110 1111	2016
7	128	0	2048–2176	112	1 111 0000	2112
		↓	↓	↓	↓	↓
		15	3968–4096	127	1 111 1111	4032

Table 5.2 provides a detailed specification of A-law coding and decoding. The segments s are numbered 0 to 7 in the first column, and have step sizes Δ given in column 2. Each segment is divided into 16 intervals v numbered from 0 to 15 in column 3, with the input range X of each of these intervals given in column 4. All X values in one interval are mapped to a single (quantised) output Y_q given in column 5. For example, the first interval of segment $s = 5$ starts at $X = 512$ and ends at $X = 544$ and produces a single output $Y_q = 80$; the second interval of segment $s = 5$ starts at $X = 544$ and ends at $X = 576$ and produces a single output $Y_q = 81$; and so on as shown in Fig. 5.8, the size of each interval being the step size of the segment. Column 6 gives the encoded output, which is an eight-bit binary number $b_7b_6b_5b_4b_3b_2b_1b_0$ assigned as follows:

- The most significant bit (MSB) b_7 is set to 1 to indicate a positive input sample and to 0 for a negative input sample.
- The next 3 bits $b_6b_5b_4$ is the binary number equivalent of the segment s (=0 to 7) within which the input sample X falls.
- The last 4 bits $b_3b_2b_1b_0$ is the binary number equivalent of the interval v (=0 to 15) of segment s in which the input sample X falls.

For example, from Fig. 5.8, a positive input $X = 789$ lies in interval $v = 8$ of segment $s = 5$ and so is encoded as 11011000. Finally, at the receiver the PCM codewords of column 6 are converted into the quantised outputs of column 7, each of which is simply the middle of the corresponding input range X in column 4. This final step at the receiver corresponds to passing Y_q (column 5) through an expander to obtain X_q (column 7), as earlier illustrated in Fig. 5.6.

μ-Law PCM is based on quantisation derived from a 15 segment approximation of the μ-law characteristic of Eq. (5.25) as shown in Fig. 5.9. Note that there are eight segments s_0 to s_7 in the positive range and the same number in the negative range (not shown), but s_0 straddles the origin and continues as a single line

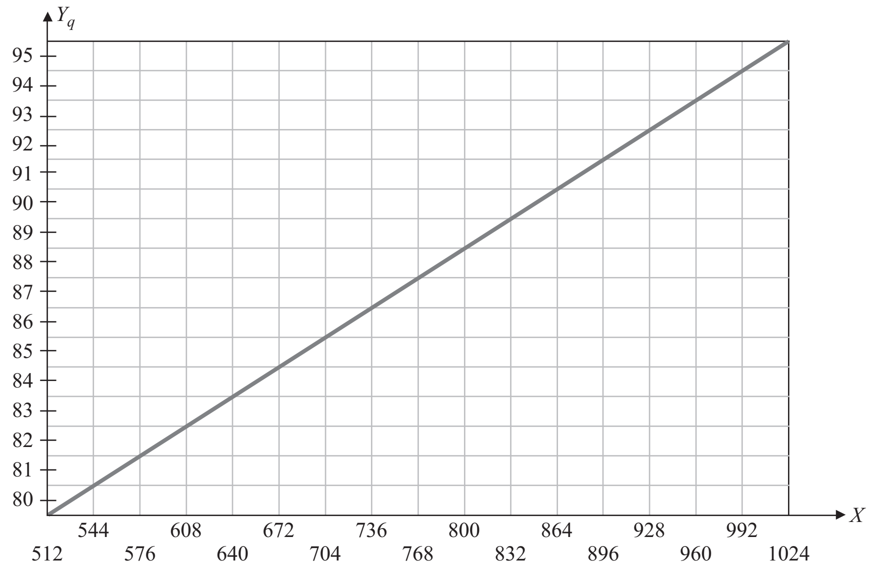


Fig. 5.8 Detailed view of A-law segment $s = 5$

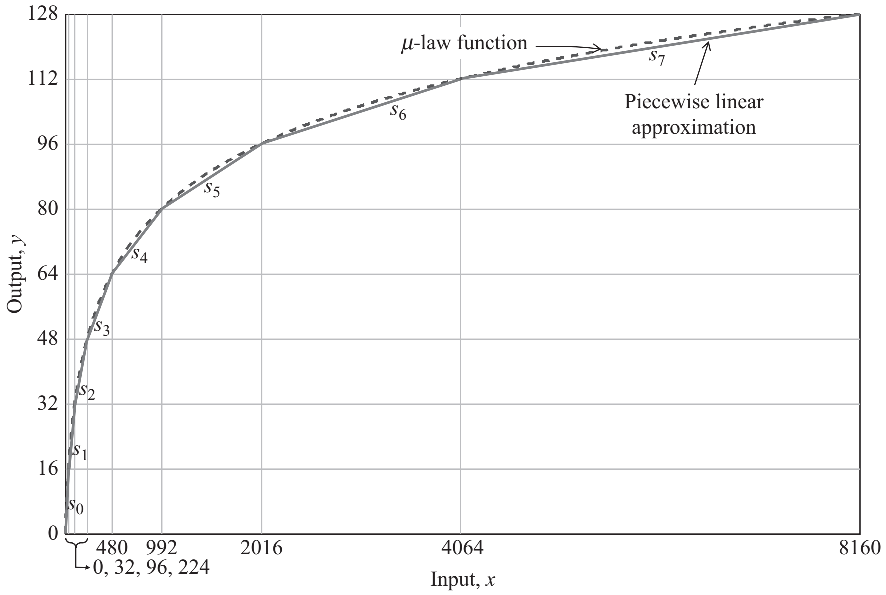


Fig. 5.9 Piecewise linear approximation of μ-law compression function

Table 5.3 μ-Law coding and decoding specifications

Segment (s)	**Step size (Δ)**	**Segment interval (v)**	**Input range (X)**	**Quantised (compressor) output (Y_q)**	**Output code (8 bit)**	**Receiver (expander) output (X_q)**
0	2	0	0–2	0	1 000 0000	1
		↓	↓	↓	↓	↓
		15	30–32	15	1 000 1111	31
1	4	0	32–36	16	1 001 0000	34
		↓	↓	↓	↓	↓
		15	92–96	31	1 001 1111	94
2	8	0	96–104	32	1 010 0000	100
		↓	↓	↓	↓	↓
		15	216–224	47	1 010 1111	220
3	16	0	224–240	48	1 011 0000	232
		↓	↓	↓	↓	↓
		15	464–480	63	1 011 1111	472
4	32	0	480–512	64	1 100 0000	496
		↓	↓	↓	↓	↓
		15	960–992	79	1 100 1111	976
5	64	0	992–1056	80	1 101 0000	1024
		↓	↓	↓	↓	↓
		15	1952–2016	95	1 101 1111	1984
6	128	0	2016–2144	96	1 110 0000	2080
		↓	↓	↓	↓	↓
		15	3936–4064	111	1 110 1111	4000
7	256	0	4064–4320	112	1 111 0000	4192
		↓	↓	↓	↓	↓
		15	7904–8160	127	1 111 1111	8032

into the negative range, giving a total of 15 distinct segments. To render all step sizes as integers, the input is normalised to $\pm C$, where $C = 8160$. Table 5.3 gives detailed specification of μ-law coding and decoding, and is interpreted in the same way as Table 5.2. We see that μ-law PCM uses step sizes that start at a minimum value $\Delta_{\min} = 2$ in the lowest segment s_0 and progressively double in each subsequent segment to a maximum $\Delta_{\max} = 256$ in the top segment s_7. This means — see explanation under A-law — that μ-law PCM is equivalent to a coarse uniform quantisation of the largest input samples using

$$k = \log_2(2C/\Delta_{\max}) = \log_2(2 \times 8160/256) = 6 \text{ bits/sample}$$

and a fine uniform quantisation of the smallest input samples using

$$k = \log_2(2C/\Delta_{\min}) = \log_2(2 \times 8160/2) = 13 \text{ bits/sample}$$

Thus, compared to eight-bit linear ADC, μ-law PCM provides a higher resolution coding of the smallest input samples by five extra bits and hence a companding gain of 30 dB, which is achieved at the expense of a lower resolution coding of the largest input samples by two fewer bits, corresponding to a companding penalty of 12 dB.

Table 5.4 gives a summary of A-law and μ-law PCM step sizes and segments, which may be applied as follows for PCM coding and decoding as well as to determine quantisation error.

Given samples $x(n)$ of an analogue input signal of peak absolute value V_p which fully loads the quantiser, the first step before using Table 5.4 is to normalise

Table 5.4 Summary of A-law and μ-law segments and step sizes

Segment (s)	**A-law**			**μ-Law**		
	Step size (Δ)	**Input range**		**Step size (Δ)**	**Input range**	
		$X_{\min}$	$X_{\max}$		$X_{\min}$	$X_{\max}$
0	2	0	32	2	0	32
1	2	32	64	4	32	96
2	4	64	128	8	96	224
3	8	128	256	16	224	480
4	16	256	512	32	480	992
5	32	512	1024	64	992	2016
6	64	1024	2048	128	2016	4064
7	128	2048	4096	256	4064	8160

these inputs to the range $\pm C$ of the table, where $C = 4096$ for A-law and $C = 8160$ for μ-law. That is, scale $x(n)$ to $X(n)$ by multiplying $|x(n)|$ by a scale factor F, where

$$X(n) = |x(n)|F$$
$$F = \begin{cases} 4096/V_p, & \text{A-law} \\ 8160/V_p, & \mu\text{-law} \end{cases} \tag{5.38}$$

To determine the PCM code to which the sample $x(n)$ is converted:

- Look at Table 5.4 (using the LHS of the table for A-law and the RHS for μ-law) and read the segment s in which $X(n)$ calculated above in Eq. (5.38) lies. For example, for A-law, $X(n) = 600$ is in segment $s = 6$, which has step size $\Delta = 32$ and lower limit $X_{\min} = 512$, and for μ-law $X(n) = 600$ is in segment $s = 4$, which has lower limit $X_{\min} = 480$.
- Next, determine interval v (numbered 0 to 15) of segment s in which $X(n)$ lies:
$$v = \left\lfloor \frac{X(n) - X_{\min}}{\Delta} \right\rfloor \tag{5.39}$$
where $\lfloor z \rfloor$ denotes the integer part of the number z, Δ is the step size and $X_{\min}$ the lower limit of segment s. In the above A-law example with $X(n) = 600$, $s = 5$, $\Delta = 32$ and $X_{\min} = 512$, we obtain $v = \lfloor 2.75 \rfloor = 2$.
- Finally, the 8-bit PCM codeword for the sample $x(n)$ is obtained by setting the MSB to signify the sign of $x(n)$ as previously discussed, and setting the next three bits to the binary equivalent of s and the last four bits to the binary equivalent of v (using leading zeroes as necessary in both cases). Thus in the above example, the A-law PCM codeword is 1 101 0010.

The sample value $x_q(n)$ to which a PCM codeword $b_7b_6b_5b_4b_3b_2b_1b_0$ is decoded is obtained through the following steps:

$$\begin{aligned} s &= 4b_6 + 2b_5 + b_4 \\ v &= 8b_3 + 4b_2 + 2b_1 + b_0 \\ X_q(n) &= X_{\min} + (v + \frac{1}{2})\Delta \\ x_q(n) &= \frac{X_q(n)}{F}(-1)^{b_7+1} \end{aligned} \tag{5.40}$$

where $X_{\min}$ and Δ are respectively the lower limit and step size (read from Table 5.4) of the segment s determined in the first line, F is the scale factor defined in Eq. (5.38) and the last line sets the sign of $x_q(n)$ based on the MSB b_7 and de-normalises to correctly place $x_q(n)$ within the analogue signal range $\pm V_p$. An alternative way to determine the decoded sample $x_q(n)$ from s and v without having to read Table 5.4 is provided by the following formula:

$$x_q(n) = \frac{(-1)^{b_7+1}}{F} \times \begin{cases} 2^{s-1}(2v+1), & \text{A-law}, \quad s = 0 \\ 2^{s-1}(2v+33), & \text{A-law}, \quad s = 1, 2, 3, \ldots, 7 \\ 2^{s}(2v+33) - 32, & \mu\text{-law} \end{cases} \tag{5.41}$$

For example, assuming $V_p = 1$, then $F = 4096$ for A-law and $F = 8160$ for μ-law. Using Eqs. (5.40) and (5.41), an A-law PCM codeword 0 001 1011 would be decoded to the value $x_q(n)$ as follows:

$$\begin{aligned} s &= 4 \times 0 + 2 \times 0 + 1 = 1 \\ v &= 8 \times 1 + 4 \times 0 + 2 \times 1 + 1 = 11 \\ x_q(n) &= \frac{(-1)^{0+1}}{4096} 2^{1-1}(2 \times 11 + 33) = -\frac{55}{4096} = -0.0134 \end{aligned}$$

And a μ-law codeword 1 110 0101 is decoded to the value $x_q(n)$ as follows:

$$\begin{aligned} s &= 4 \times 1 + 2 \times 1 + 0 = 6 \\ v &= 8 \times 0 + 4 \times 1 + 2 \times 0 + 1 = 5 \\ x_q(n) &= \frac{(-1)^{1+1}}{8160} \left[2^6(2 \times 5 + 33) - 32\right] = \frac{64 \times 43 - 32}{8160} = 0.3333 \end{aligned}$$

Finally, given N_s samples of an analogue signal $x(n)$, $n = 0, 1, \ldots, N_s - 1$, the quantised values $x_q(n)$ are obtained as discussed above, and hence the SQNR of the PCM signal is estimated as

$$\text{SQNR} = 10 \log_{10} \left(\frac{\sum_{n=0}^{N_s-1} \left[x(n) - x_q(n)\right]^2}{\sum_{n=0}^{N_s-1} x^2(n)} \right) \text{dB} \tag{5.42}$$

5.5.2 SQNR of A-law and μ-law PCM

We know from Eq. (5.19) that the SQNR of an ideal log-PCM (i.e. one that uses an entirely logarithmic compressor curve) is constant across the entire input range. For an 8-bit quantiser (for which $N = 2^8 = 256$) this constant SQNR is

$$\text{SQNR} = 10 \log_{10} \left(\frac{3N^2}{K^2} \right) = \begin{cases} 38.2 \text{ dB}, & \text{for } K = 1 + \ln(A), \quad A = 87.6 \\ 38.1 \text{ dB}, & \text{for } K = \ln(1 + \mu), \quad \mu = 255 \end{cases}$$

However, in implementing A-law and μ-law PCM, piecewise linear approximation of the logarithmic curve was employed. It is of interest to assess the impact of this approximation on SQNR. Is the SQNR of A-law and μ-law PCM still constant at ~38 dB across the entire input range? We will carry out this evaluation for an input signal of peak value V_p that has uniform distribution, which means that its sample is equally likely to be located anywhere in the interval ($-V_p$, V_p) and that its peak-to-rms ratio is $R = \sqrt{3}$. The normalised power of this signal is

$$P_s = V_{\text{rms}}^2 = (V_p/R)^2 = V_p^2/3 \tag{5.43}$$

Quantisation noise power is given by MSQE in Eq. (5.6), but here the step size varies from segment to segment, so we obtain MSQE by summing the product of $\Delta_j^2/12$ and the probability P_j that the sample of the input signal lies in the j^{th} segment pair (i.e. including both positive and negative ranges). If $X_{s\,\min} \le V_p \le X_{s\,\max}$, where $X_{s\,\min}$ and $X_{s\,\max}$ are the lower and upper limits of segment s (i.e. the peak value of the input signal lies in segment s) then $P_j = 16\Delta_j/V_p$ for segments $j = 0, 1, \ldots, s-1$, and $P_j = (V_p - X_{s\,\min})/V_p$ for $j = s$, so that

$$\begin{aligned}\text{MSQE} &= \frac{1}{12}\sum_{j=0}^{s-1}\Delta_j^2\left(\frac{16\Delta_j}{V_p}\right) + \frac{\Delta_s^2}{12}\left(\frac{V_p - X_{s\,\min}}{V_p}\right)\\ &= \frac{4}{3V_p}\sum_{j=0}^{s-1}\Delta_j^3 + \frac{\Delta_s^2}{12}\left(\frac{V_p - X_{s\,\min}}{V_p}\right)\end{aligned} \tag{5.44}$$

SQNR is the ratio between Eqs. (5.43) and (5.44) expressed in dB:

$$\text{SQNR} = 30\log_{10}(V_p) - 10\log_{10}\left[\frac{\Delta_s^2}{4}(V_p - X_{s\,\min}) + 4\sum_{j=0}^{s-1}\Delta_j^3\right]\text{ dB} \tag{5.45}$$

This is an important result that applies to both A-law and μ-law PCM and to any quantiser range $\pm C$. Note however that Tables 5.2–5.4 and Figs. 5.7–5.9 are based on $C = 4096$ for A-law and $C = 8160$ for μ-law. If a different quantiser range is to be used, say $\pm D$, then all step sizes and signal ranges in these figures and tables must be reduced by the factor C/D in order to maintain compressor shape. Let us take two examples to explain how Eq. (5.45) is applied.

Consider A-law PCM with input signal having 0 dB peak relative to quantiser input limit C. That is, the signal fully loads the quantiser and $V_p = C = 4096$. Thus, V_p lies in segment $s = 7$ with step size $\Delta_s = 128$ and lower limit $\Delta_{s\,\min} = 2048$. The SQNR of the PCM representation of this signal is obtained from Eq. (5.45) as

$$\begin{aligned}\text{SQNR} &= 30\log(4096)\\ &\quad -10\log\left[\frac{128^2(4096-2048)}{4} + 4(2^2+2^2+4^2+8^2+16^2+32^2+64^2)\right]\\ &= 108.37 - 69.82 = 38.55\text{ dB}\end{aligned}$$

Next, consider a smaller input signal having peak value 15 dB below quantiser input limit. We apply Eq. (5.45) with V_p 15 dB below 4096, which means

$$V_p = 4096/10^{15/20} = 728.38$$

From Table 5.4 (A-law) we see that V_p lies in segment $s = 5$ with step size $\Delta_s = 32$ and lower limit $X_{s\,\min} = 512$. Substituting into Eq. (5.45) yields

$$\begin{aligned}\text{SQNR} &= 30\log(728.38) - 10\log\left[\frac{32^2(728.38-512)}{4} + 4(2^2+2^2+4^2+8^2+16^2)\right]\\ &= 85.87 - 48.70 = 37.17\text{ dB}\end{aligned}$$

We can therefore see the effect of the piecewise linear approximation on the SQNR of A-law PCM, which slightly exceeds 38 dB at the top end of input signals and is a little lower for smaller input signals. Fig. 5.10 shows a plot of SQNR (dB) of A-law and μ-law PCM versus peak input level V_p in dB relative to quantiser limit C, calculated as discussed above using Eq. (5.45). A peak input level of 0 dB corresponds to an input signal that fully loads the quantiser so that its peak $V_p = C$, whereas a peak input level of −60 dB corresponds to a very small input signal having peak value $V_p = C/1000$. The SQNR of k-bit linear ADC for $k = 8$, 12, 13, calculated using Eq. (5.10), is also shown for comparison. Fig. 5.10 confirms some of the observations made earlier. For example:

- The performance, in terms of SQNR, of (eight bit) A-law PCM coincides with that of a 12-bit linear ADC for small inputs ($V_p \leq -36$ dB).
- The SQNR of μ-law PCM equals that of a 13-bit linear ADC for small inputs ($V_p \leq -48$ dB).
- The vertical gap between the SQNR curves for eight-bit linear ADC and A-law PCM in the small input region ($V_p \leq -36$ dB) is 24 dB. This is the A-law companding gain.
- The vertical gap between the SQNR curves for eight-bit linear ADC and μ-law PCM in the small input region ($V_p \leq -48$ dB) is 30 dB. To obtain this you will need to extend the SQNR curve for μ-law in the small input region (which coincides with the linear graph for 13-bit ADC) backwards until you reach $V_p = -38$ dB where the eight-bit linear graph stops. The vertical gap between the two graphs at this point is 30 dB. This is the μ-law companding gain.
- The SQNR of A-law PCM remains within the range (37, 38.6 dB) over a 30 dB range of peak input levels and degrades by only ~2 dB at peak input $V_p = -36$ dB. The peaks in the slight fluctuation in SQNR in this region coincide with segment boundaries. In comparison, the SQNR of eight-bit linear ADC is an unacceptable 12 dB at $V_p = -36$ dB. We see that due to the

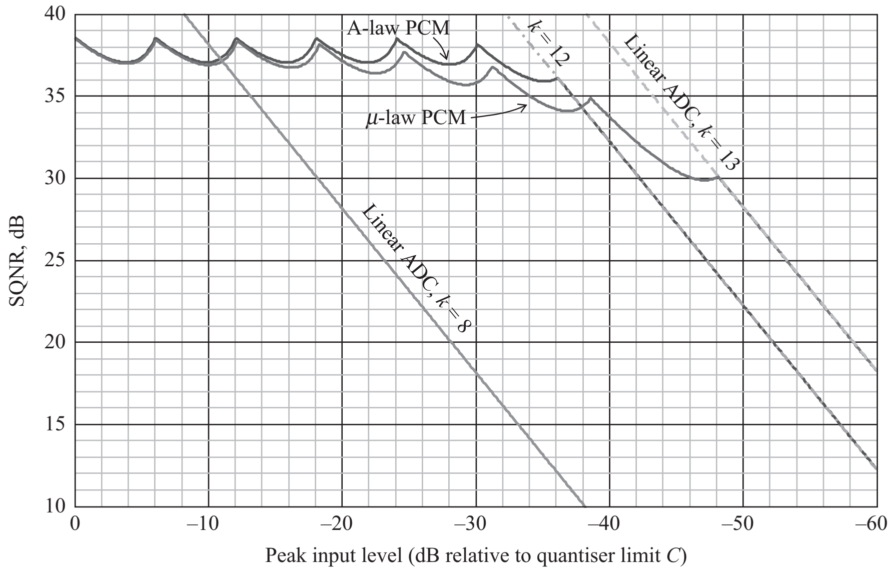


Fig. 5.10 SQNR of A-law PCM, μ-law PCM and k-bit linear ADC (for k = 8, 12, 13*)*

piecewise linear approximation employed we have not quite kept SQNR exactly constant across the entire range of input levels, but we have succeeded in keeping the variation small to within about ± 1 dB over a 36 dB range of input levels. Note that below $V_p = -36$ dB, which is $V_p = 64$ in the ± 4096 normalised range, the signal is confined entirely within segments $s = 0$ and $s = 1$ where the step size is constant at $\Delta = 2$. Thus the signal sees a linear ADC and hence SQNR decreases linearly with peak input level in this region.

- The SQNR of A-law PCM is slightly higher than that of μ-law PCM for inputs > -38 dB, but at small inputs < -48 dB μ-law PCM achieves a 6 dB better SQNR than A-law due to its use of a 13-bit/sample coding resolution for small samples compared to A-law's 12 bits/sample.

5.6 Lossy compression

Speech quality is often classified using a measure called *mean opinion score* (MOS), which is the average of the subjective judgement passed by a large group of listeners on the quality of the speech using a scale from 1 to 5, where 1 is bad, 2 is poor, 3 is fair, 4 is good and 5 is excellent. Alternatively, the scale of 1 to 5 gives a degradation category rating (DCR) that indicates the level of the perceived disturbance in the speech as very annoying (1), annoying (2), slightly annoying (3), audible but not annoying (4), and inaudible (5). An MOS of 4 and above indicates speech of *near transparent quality*, also described as *toll quality*. A score between 3.5 and 4 specifies speech of *communication quality*; a score between 3.0 and 3.5 is described as *professional quality*; and a score below 3.0 corresponds to speech of *synthetic quality*.

PCM at 64 kb/s is the toll quality benchmark for *telephone speech* – the name for an audio signal restricted to a maximum frequency component of 3400 Hz. Other audio signals include *wide band speech*, which has a maximum frequency component of 7 kHz, and *wide band audio* (also known as *high fidelity audio*) of bandwidth that covers the entire audible range of frequencies up to a maximum of 20 kHz. Using 64 kb/s PCM for telephone speech however places a high demand on transmission bandwidth, storage capacity and battery life which can be difficult to satisfy in applications such as mobile wireless communication where these resources are usually in very short supply. A wide variety of lossy compression techniques have been developed for application to audio and (both still and moving) image signals. These techniques allow telephone speech for example to be reduced from a bit rate of 64 kb/s to various standardised lower rates such as 32, 16, 4 and 2.4 kb/s. The following trade-offs (illustrated in Fig. 5.11) are involved in the lossy compression of speech in particular.

- *Quality*: As compression ratio increases, leading to lower bit rate and hence lower transmission bandwidth requirement, perceived quality generally decreases. The decline in quality however depends on the processing algorithm. In fact there are some codecs that produce compressed or lower-bit-rate speech having comparable quality to less compressed or higher-bit-rate speech.
- *Processing delay*: This is the time lag between the instants at which the speech signal enters and leaves the coder, and consists of a buffering delay τ_b plus computational delay τ_c. Most compression algorithms work at a time on blocks or segments of the speech signal of duration $\tau_b \geq 20$ ms. The first speech

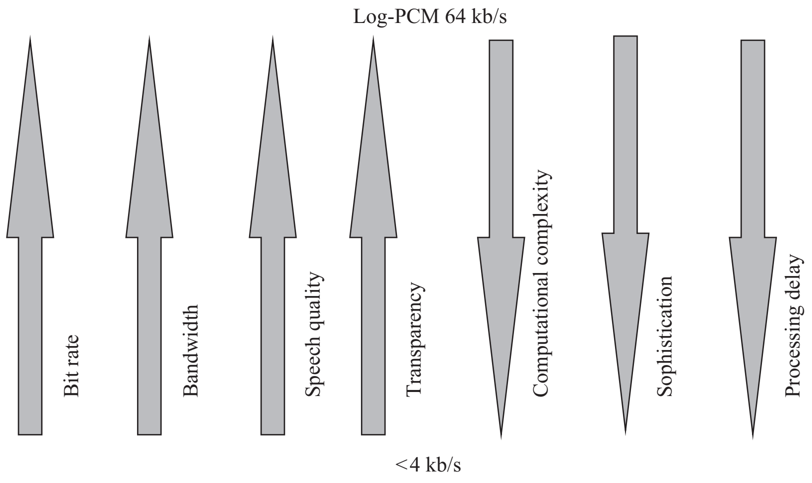


Fig. 5.11 Trade-offs in speech processing

sample in a segment will be τ_b ms old before commencement of computations of duration τ_c needed by the algorithm to produce the codeword representation of the speech segment. The maximum processing delay $\tau_b + \tau_c$ at a coder tends to increase with codec complexity and there will be a further propagation delay τ between transmitter and receiver as well as additional processing delay at the receiver before the speech segment can be delivered to the information sink at destination. For real-time or interactive communication it is vital that total delay is kept at a tolerable level and this requirement can sometimes be a critical factor in delaying the adoption of a given algorithm until advances in DSP technology make sufficiently fast codecs available.

- *Channel errors*: The effect of bit errors tends to be more drastic as bit rate decreases since information is highly compressed. For example, an error in a bit that indicates whether a segment represents voiced or unvoiced speech will have a significant impact on the interpretation and rendering of that entire segment at the decoder.
- *Transparency*: Highly speech-specific codecs might not pass non-speech signals and this lack of transparency generally increases as bit rate decreases.
- *Algorithmic complexity*: This generally increases as more processing is introduced to further increase data compression and hence reduce bit rate. The result is more expensive hardware as well as higher power consumption and hence a shorter battery life for portable communication devices.

Lossy data compression is a wide subject, a detailed treatment of which is beyond the scope of a general digital communications book such as this one. We will therefore limit our interest to the lossy compression of telephone speech and briefly highlight the main types, including waveform coders, vocoders and hybrid coders.

5.6.1 Waveform coders

Waveform coders produce a bit stream that conveys a representation of the original signal waveform. At the receiver, the decoder attempts to reproduce the original waveform from the coded bit stream, but there will always be some added noise arising from the error introduced by quantisation at the encoder. If the original signal is speech and this noise is practically inaudible, the MOS of the reconstructed speech will be between ~4.5 and 5.0 and the coder is said to produce toll quality speech. At the transmitter a waveform coder performs the following processing steps on the original signal:

- *Filtering*: The original signal is first passed through an anti-alias filter, designed as discussed in Section 4.8.
- *Sampling*: The filtered signal $g(t)$ is then sampled at a suitable rate, as discussed in Section 4.4. In this Section, we will refer to the sample of $g(t)$ taken at time $t = nT_s$ as the current sample, denoted simply as $g(n)$. Thus the p^{th} previous sample, taken at an earlier time $t = nT_s - pT_s$ will be denoted by $g(n - p)$.
- *Quantisation*: A k-bit quantiser is employed to map each input $\varepsilon(n)$ to a quantised output $\varepsilon_q(n)$, which is selected from 2^k quantisation levels on the basis of whichever level is the closest to $\varepsilon(n)$. This introduces an inevitable quantisation error $e_q(n)$, where

  $$\varepsilon_q(n) = \varepsilon(n) + e_q(n) \tag{5.46}$$

  The quantisation process was discussed in detail in Sections 5.2–5.4. The PCM discussed in Section 5.5 is a special case of waveform coders for which quantiser input $\varepsilon(n)$ is the sampled signal $g(n)$ and the quantiser is non-uniform, employing $k = 8$ bits to code each sample $g(n)$ taken at 8000 samples/s, which results in the standard 64 kb/s bit rate. In that case the linear prediction discussed next is not required. Alternatively, the quantiser might be a *differential quantiser* (introduced in Section 5.3.1) in which case quantiser input $\varepsilon(n)$ is the difference between $g(n)$ and a linear prediction $\hat{g}(n)$.
- *Linear prediction*: A weighted sum of the last p samples, from $g(n - 1)$ to $g(n - p)$, is computed as a prediction of what the current sample $g(n)$ should be. The computation is a linear combination of samples and so the process is referred to as *linear prediction*. The prediction $\hat{g}(n)$ is obtained as follows:

  $$\hat{g}(n) = \sum_{j=1}^{p} w_j g_q(n-j) = \sum_{j=1}^{p} w_j \left[g(n-j) + e_q(n-j)\right] \tag{5.47}$$

  where w_j are the predictor (also called filter) coefficients or weights, $e_q(n)$ is the quantisation error introduced into the output $\varepsilon_q(n)$ of the quantiser whose input $\varepsilon(n)$ is the difference between the current sample $g(n)$ and the result of Eq. (5.47). That is

  $$\text{Quantiser input, } \varepsilon(n) = g(n) - \hat{g}(n) \tag{5.48}$$

  Notice that the predictor input is

  $$g_q(n) = g(n) + e_q(n) \tag{5.49}$$

  and not $g(n)$ which could cause an accumulation of quantisation errors.
- *Encoding*: The quantiser output $\varepsilon_q(n)$ is coded as a bit sequence by an encoder. A discussion of how the encoder maps each of the 2^k levels of the quantiser into a binary codeword was presented in Section 5.5.1 for A-law and μ-law PCM.

Alternatively, the quantiser can be treated as a discrete source that emits 2^k levels or symbols to which codewords can be assigned according to the source coding methods discussed in the next chapter. Whatever the encoding process employed, the resulting bit stream conveys a digital representation of the original waveform $g(t)$ to the receiver. Since k is typically kept small to reduce coding bits and the recovered signal is never an identical copy of $g(t)$ due to the quantisation errors $e_q(n)$ introduced, the encoder bit stream output is described as a *lossy compression* of $g(t)$, or (if $g(t)$ is a speech signal) as *low bit rate speech coding*.

A waveform decoder is required at the receiver which performs the following functions:

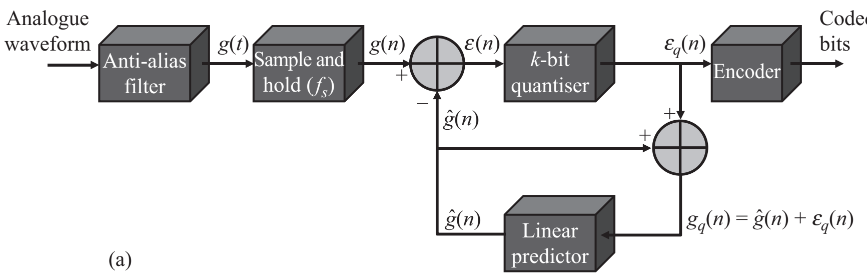
- *Decoding*: The decoder receives the coded bit stream and converts it into a sequence of quantised values $\varepsilon_q(n)$, using a procedure dictated by the encoding process at the transmitter.
- *Linear prediction*: The receiver employs an exactly identical prediction algorithm – Eq. (5.47) – as used at the transmitter to predict the current sample from the last p samples. Barring any transmission error, then the sum of the predictor output $\hat{g}(n)$ and decoder output $\varepsilon_q(n)$ gives the current sample $g(n)$ correct to within the quantisation error introduced at the transmitter. This is the case since using Eqs. (5.46), (5.48) and (5.49)

$$\begin{aligned}\hat{g}(n) + \varepsilon_q(n) &= \hat{g}(n) + \varepsilon(n) + e_q(n)\\ &= g(n) + e_q(n)\\ &= g_q(n)\end{aligned} \tag{5.50}$$

- *Reconstruction filter*: The sequence of quantised samples $g_q(n)$ is passed through a reconstruction filter that smooths it to produce the reconstructed analogue waveform $g_r(t)$. The design and operation of the reconstruction filter was discussed in Chapter 4. A subjective measure of the quality of the reconstructed signal $g_r(t)$, giving an indication of how close it is to the original waveform $g(t)$ and how free it is from quantisation noise, is provided by a mean opinion score (MOS) as earlier discussed, whereas an objective measure of the quality of $g_r(t)$ is given by the SQNR discussed in Section 5.5.2.

Fig. 5.12 shows block diagrams of the above operations at the transmitter and receiver. The linear predictor is a finite impulse response (FIR) filter of order p, a block diagram of which is shown in Fig. 5.13. Optimum values of the filter coefficients w_j, $j = 1, 2, 3, \ldots, p$ are determined to minimise the mean square value $\overline{\varepsilon^2(n)}$ of prediction error in a zero-mean stationary segment of the signal $g(t)$ spanning N samples. Equating to zero each of the p partial derivatives of $\overline{\varepsilon^2(n)}$ taken with respect to w_j leads to the Wiener–Hopf equations for the coefficients expressed in matrix form as

$$\begin{bmatrix} R_g(0) & R_g(1) & \cdots & R_g(p-1)\\ R_g(1) & R_g(0) & \cdots & R_g(p-2)\\ \vdots & \vdots & \ddots & \vdots\\ R_g(p-1) & R_g(p-2) & \cdots & R_g(0)\end{bmatrix}\begin{bmatrix} w_1\\ w_2\\ \vdots\\ w_p\end{bmatrix} = \begin{bmatrix} R_g(1)\\ R_g(2)\\ \vdots\\ R_g(p)\end{bmatrix} \tag{5.51}$$



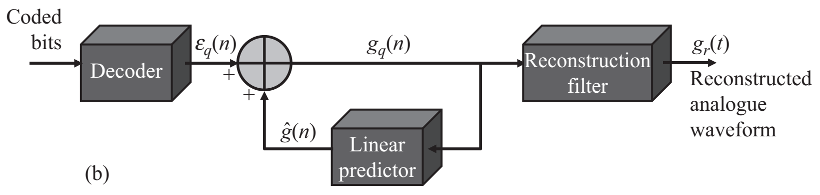


Fig. 5.12 Waveform coder: (a) encoding operation at transmitter; (b) decoding operation at receiver

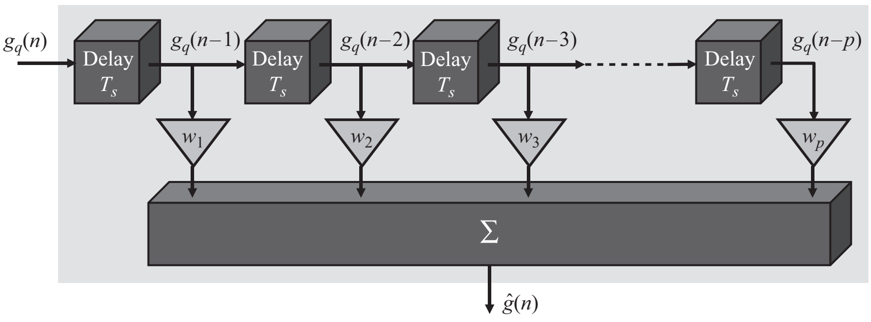


Fig. 5.13 Linear predictor realised as a finite impulse response (FIR) filter of order p

where

$$R_g(m) = \frac{1}{N}\sum_{n=0}^{N-1-m} g_q(n)g_q(n+m) \tag{5.52}$$

is the autocorrelation of $g_q(n)$ computed over the N samples of the signal segment. It gives an indication of how two samples of $g_q(n)$ (separated by m sampling intervals) are related, a value of zero meaning that the two samples are not in any way related. Although the p-by-p autocorrelation matrix of Eq. (5.51) can be inverted to yield the filter coefficients, the autocorrelation values of the signal sequence $g_q(n)$ are not always known. In such situations the *least mean square* (LMS) *algorithm* is an excellent way to iteratively compute the coefficients starting from an arbitrary set of values: The estimate $\hat{w}_{j,n+1}$ of the j^{th} filter coefficient at the next sampling instant is given as follows [2] in terms of its current estimate $\hat{w}_{j,n}$, the j^{th} past quantised signal sample $g_q(n-j)$ and the current prediction error

$$\begin{aligned}\hat{w}_{j,n+1} &= \hat{w}_{j,n} + \mu g_q(n-j)\varepsilon(n)\\ &= \hat{w}_{j,n} + \mu g_q(n-j)\left[g(n) - \sum_{j=1}^{p}\hat{w}_{j,n}g_q(n-j)\right]\end{aligned} \tag{5.53}$$

In addition to PCM (Section 5.5), other special cases and standards of waveform coders include the following:

- *Delta modulation (DM)*: When in Figs. 5.12 and 5.13, $k=1, p=1$ and $w_1=1$, we have the special case of delta modulation (DM). The prediction is simply the previous sample and the linear predictor simplifies to a one-sample delay. In order to keep the difference $\varepsilon(n)$ between adjacent samples small, sampling rate f_s is chosen higher than required by the sampling theorem. Values of $f_s = 16$ kHz and 32 kHz have been used for telephone speech. SQNR increases with sampling frequency by 9 dB per octave [3]. The quantiser output is Δ when the current sample $g(n)$ exceeds the previous sample $g(n-1)$ and $-\Delta$ otherwise, which the encoder codes using binary 1 for Δ and binary 0 for $-\Delta$. At the receiver the quantised sequence $g_q(n)$ is recovered using the following algorithm:

$$g_q(n) = \begin{cases} g_q(n-1)+\Delta, & b(n)=1\\ g_q(n-1)-\Delta, & b(n)=0\end{cases} \tag{5.54}$$

  where $b(n)$ denotes the current bit, and Δ is a fixed step size that approximates the amount by which the signal changed from the previous sample. The choice of this step size is a compromise between on the one hand the need for a sufficiently large value to avoid *overload distortion* when the analogue waveform $g(t)$ changes so rapidly that adding or subtracting a step size is insufficient to make the current sample catch up with the previous, and on the other hand the need for a sufficiently small value to minimise *granular noise* during a passage when $g(t)$ changes very slowly so that the reconstructed signal hunts about the slowly changing level of $g(t)$. One solution would be to use a small fixed step size to minimise granular noise and combine that with a very high sampling frequency to minimise overload distortion since this ensures that samples are taken before $g(t)$ has had the time to change by much. This ensures that the DM system retains one of its great features, namely circuit simplicity, but it increases bandwidth requirement towards that of standard PCM. An alternative solution, which increases circuit complexity but keeps bit rate at 32 kb/s or lower (thereby saving bandwidth when compared to PCM) is to adopt a variable step size in what is then referred to as *adaptive delta modulation* (ADM). The step size is increased when necessary to avoid overload distortion and reduced when necessary to minimise granular noise. Encoder and decoder both start with a small step size having an agreed value and then follow the same algorithm for selecting the step size $\Delta(n)$ of the current interval. A simple algorithm might be the following which increases the step size by a factor of λ (>1) when the current bit is the same as the previous and reduces the step size by the same factor if the two bits are different:

$$\Delta(n) = \begin{cases} \Delta(n-1)\lambda, & b(n) \oplus b(n-1) = 0 \\ \Delta(n-1)/\lambda, & \text{Otherwise} \end{cases} \tag{5.55}$$

Further discussion of delta modulation can be found in [3].

- *Adaptive differential PCM (ADPCM)*: If in Fig. 5.12 sampling rate $f_s = 8$ kHz then the use of $k = 2$, 3, 4 and 5 bits/sample at the quantiser leads to coding of the analogue signal at respective bit rates of 16, 24, 32 and 40 kb/s. The ITU-T G.726 standard [4] is the most widely used ADPCM implementation. It uses a combination of one second order and one sixth order linear predictor whose coefficients are constantly updated based on prediction error and reconstructed signal. Adaptive non-uniform quantisation is employed in which the spacing of quantiser levels is adaptively scaled as necessary to maintain SQNR at approximately the same level for different signals. The speech quality of 32 kb/s ADPCM matches that of 64 kb/s PCM. A detailed specification of ADPCM can be found in [4].

Worked Example 5.6: SQNR of Delta Modulation

We wish to determine the SQNR of a delta modulation (DM) signal that conveys telephone speech at 16 kb/s.

Since bit rate $R_b = kf_s$, and $k = 1$ in a DM coder, it follows that sampling frequency $f_s = R_b = 16$ kb/s. The maximum frequency component of telephone speech is $f_m = 3.4$ kHz. To minimise granular noise while avoiding overload distortion, we must choose step size Δ_{min} equal to the maximum change in voltage level in one sampling interval ($T_s = 1/f_s$) experienced by the highest frequency component of telephone speech. In the worst case scenario this frequency component also has the maximum amplitude A_m, and its sinusoidal expression is $v_m(t) = A_m \sin(2\pi f_m t)$. Thus,

$$\begin{aligned} \Delta_{\text{min}} &= T_s \frac{dv_m(t)}{dt}\Big|_{\text{max}} \\ &= T_s 2\pi f_m A_m \cos(2\pi f_m t)|_{\text{max}} \\ &= 2\pi f_m A_m / f_s \end{aligned} \tag{5.56}$$

In DM, a positive difference signal ε is quantised to step size Δ and a negative difference signal $-\varepsilon$ is quantised to $-\Delta$, which gives quantisation error $|\varepsilon - \Delta|$ and hence (assuming the difference ε is uniformly distributed in the range $-\Delta$ to $+\Delta$) a mean square quantisation error (MSQE) given by

$$\begin{aligned} \text{MSQE} &= \frac{1}{\Delta} \int_{\varepsilon=0}^{\Delta} (\varepsilon - \Delta)^2 d\varepsilon \\ &= \Delta^2/3 \end{aligned} \tag{5.57}$$

Using this expression and the step size specified in Eq. (5.56), and making the reasonable assumption that this quantisation noise power is spread equally over the frequency range $f = 0$ to f_s so that at the receiver only a fraction B/f_s of it will pass through the reconstruction filter whose bandwidth $B = f_m$, we obtain

$$\begin{aligned}
\text{SQNR} &= \frac{\text{Signal power}}{\text{Effective quantisation noise power}} \\
&= \frac{A_m^2/2}{(f_m/f_s)(\Delta^2/3)} = \frac{A_m^2/2}{(f_m/f_s)(2\pi f_m A_m/f_s)^2/3} \\
&= \frac{3}{8\pi^2}(f_s/f_m)^3 \\
&= 30\log_{10}\left(\frac{f_s}{f_m}\right) - 14.2\ (\text{dB}) \qquad (5.58)
\end{aligned}$$

Eq. (5.58) is an important result for the SQNR of DM systems. It is remarkable that (unlike PCM) the SQNR of DM does not depend on input signal amplitude. It depends only on sampling frequency f_s (since we really don't have any control over f_m which is fixed), improving by 9 dB per octave increase in f_s.

Substituting $f_s = 16000$ and $f_m = 3400$ yields SQNR = 5.98 dB, which is quite poor. To improve SQNR we could increase f_s to gain a 9 dB improvement each time fs is doubled. Thus a 32 kb/s DM has SQNR = 15 dB and a 64 kb/s DM has SQNR = 24 dB and so on. Alternatively, we note that the reason for poor SQNR is because we chose a large step size (Eq. (5.56)) which was the minimum required to avoid overload distortion. To improve SQNR without increasing sampling frequency (and hence bit rate) we must resort to ADM to realise a 6 dB improvement in SQNR each time step size Δ is halved. Thus, if we could design a 16 kb/s ADM system that operates mostly with a step size that is a factor of 16 smaller than what is stipulated in Eq. (5.56) it would deliver speech quality at SQNR $\sim$30 dB which is quite good.

5.6.2 *Vocoder*

Unlike waveform coders, a vocoder (short for vocal tract coder) does not code the speech waveform, but it codes the mechanism that produces the speech signal to enable reproduction of a synthetic quality of the speech at the receiver. This mechanism involves the flow of air (that has been forced from the lungs by muscular action) being modified in the vocal tract to produce different sounds. The vocal tract can be modelled as a tube of non-uniform cross-section. Vowel sounds are produced by resonance of an open vocal tract under various shapes, whereas consonants are produced by turbulence generated in the vocal tract at various points of constriction. A vocoder explicitly models the vocal tract as a filter, partitions the sequence of speech samples $g(n)$ into short segments and extracts (and codes) from each segment the parameters necessary to emulate the air input into the vocal tract and the modifications imposed by the vocal tract to convert that input into output sound for that segment. At the receiver a close version of the original speech signal in each segment is reproduced by taking an appropriate input signal (e.g. white noise) and passing it through a filter (designed as dictated by the coded vocal tract

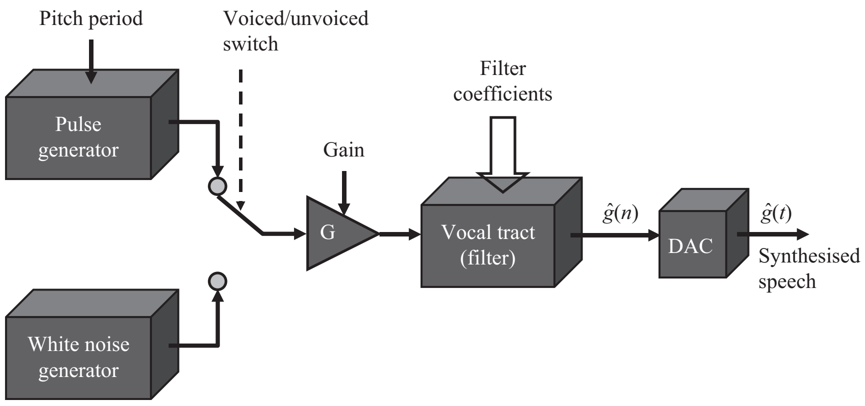


Fig. 5.14 Synthetic speech production in a linear predictive coding (LPC) vocoder

parameters) to give an output that mimics the sound in that segment. Fig. 5.14 shows a block diagram of this process for the simple case of linear predictive coding (LPC) of the vocal tract.

The main source of degradation in reproduced speech quality in vocoders is not quantisation error (although there is a little bit of that) but the use of a model that does not accurately represent the speech production process. For example, each segment of speech (of duration say 20 ms) may be classified either as voiced or unvoiced and reproduced at the receiver by exciting the filter with a pulse train or white noise respectively. In practice each speech frame will be partially voiced and unvoiced, so this binary classification into voiced or unvoiced gives a synthetic quality (of MOS $\sim$2.2) to vocoder speech. Vocoders do however give good intelligibility at very low bit rates, and therefore are widely used in applications that require intelligibility without a serious need for the recognition of speaker identity or emotion. They however involve complex processing and introduce delays $>$20 ms.

One simple vocoder example is the LPC-10 developed by the US Department of Defence in 1984 and adopted as a federal standard (FS-1015). The vocal tract is modelled using a linear predictor and the speech signal is partitioned into 22.5 ms segments or frames from which the following parameters are extracted: (a) Voiced/unvoiced flag (one bit); (b) gain factor to model the energy of the frame (five bits); (c) pitch period between 51.3 and 400 Hz (six bits); (d) linear predictor coefficients w_1, w_2, w_3 and w_4 quantised using five bits per coefficient (20 bits), and a further 21 bits devoted to error protection if the frame is classified as unvoiced, or, if the frame is voiced, to code a further six coefficients, namely w_5, w_6, w_7, w_8 (four bits each) and w_9 (three bits) and w_{10} (two bits). This sums to 53 bits per frame to which one bit, alternating between 0 and 1, is appended for synchronisation at the receiver, giving a total of 54 bits per 22.5 ms, or a bit rate of 2.4 kb/s.

Improvements to the LPC-10 vocoder approach include

- Using *vector quantisation* (VQ), rather than scalar quantisation, for the filter coefficients: VQ entails coding each set of coefficients by an index to the closest in a list of standard entries known as a *codebook*. Using this method and processing three frames together to exploit their correlations, Wang and

Jay Kuo [5] reported producing synthetic speech of comparable quality to LPC-10 but at a much lower bit rate of 800 b/s.

- Using *mixed excitation linear prediction* (MELP): The input signal used to reproduce each speech segment is a mixture of periodic pulses and white noise (rather than one or the other as in LPC-10). This improves the synthesised speech to a professional quality level of MOS ~3.0 at the same bit rate (2.4 kb/s) as LPC-10.
- Using *Sub-band coding*: This recognises that each speech frame can be more accurately modelled by looking at its content within various frequency sub-bands. So each frame is analysed to extract characteristic parameters in various sub-bands. These parameters are then coded and used at the receiver to reproduce the speech signal. The sub-band analysis and coding can be done in various ways leading to different implementations, such as *multi-band linear predictive* (MB-LPC) [6] and *improved multi-band excited* (IMBE) vocoders that produce professional quality speech (MOS ~3.2) at respective bit rates of 2.4 and 4.15 kb/s. More generally, a fast Fourier transform (FFT) analysis can be carried out on each frame to extract the sets of amplitudes and phases of constituent harmonics. At the receiver the synthesised speech for each segment is obtained as the sum of outputs produced using each set of coded parameters.

5.6.3 Hybrid coder

The general idea of hybrid coders is to employ waveform coding to code the difference between the original speech signal and a synthesised version of the signal. By combining the advantages of waveform coders and vocoders in this way, good reproduction of speech at low bit rates can be achieved. Hybrid coders use a technique called *analysis-by-synthesis*. At the transmitter various combinations of the model parameters are used to synthesise the speech as would be done at the receiver, and the combination yielding the best perceptual match to the original speech signal is selected. Hybrid codecs are therefore quite complex and introduce processing delays that are at least of the same order as vocoders. There are numerous designs of hybrid coders, the distinction being in excitation signal pattern, analysis procedure and type of information transmitted to the receiver. One example is the ITU-T G.729 standard [7] which achieves toll quality speech at 8 kb/s using a technique described as *conjugate-structure algebraic-code-excited linear prediction* (CS-ACELP). Speech signal is processed in 10 ms frames that are analysed to extract various parameters coded using a total of 80 bits, resulting in the 8 kb/s bit rate. The excitation signal at the receiver is constructed using a weighted sum of contributions from an adaptive and a fixed codebook.

5.7 Summary

Transmission of voice and other signals including video and a wide variety of sensor signals that are analogue by nature is now an all-digital process in modern communication networks. These signals must therefore be converted at source into a sequence of numbers or bit stream that contains all the significant information of the original analogue signal. In Chapter 4, we established the minimum rate at which such signals must be sampled in order to capture their information content and permit reconstruction without distortion. We followed that up in this chapter by studying in detail the final digitisation steps, involving quantisation and PCM, that represent the signal samples as a sequence of numbers and ensure that the noise introduced, while unavoidable, is completely under the system designer's control. We examined the intricacies of the concept of quantisation and studied the interplay of design parameters in PCM. Our main focus was on strategies for achieving the two main goals of minimum bandwidth and consistent and acceptable SQNR of the digital signal. In the end we made use of a piecewise linear approximation to a practically non-feasible logarithmic signal compression function and achieved a companding gain of 24 dB that allowed a saving of four bits per sample when compared to uniform quantisation of a similar low-amplitude signal.

PCM speech is a worldwide standard digital signal that conveys near transparent quality voice at 64 kb/s having an SQNR of around 38 dB, which drops by only 2.5 dB over a 36 dB reduction in analogue input signal amplitude. In some applications however, such as mobile telephony, this bit rate is excessive and measures must be found for bit rate reduction through data compression. We discussed lossy compression of telephone speech, also known as low bit rate speech coding, and highlighted techniques such as ADPCM and CS-ACELP that support toll quality voice at only 32 kb/s and 8 kb/s respectively, as well as vocoders that produce synthetic quality speech at bit rates from $\sim$4 kb/s down to $\sim$800 b/s. Lossless compression of non-voice data is the subject of the next chapter.

5.8 References

[1] ITU-T Recommendation G.701. (1993). Vocabulary of digital transmission and multiplexing, and pulse code modulation (PCM) terms. Geneva: International Telecommunications Union

[2] Haykin, S. (2013). *Digital communications systems*. Hoboken: Wiley, p. 301

[3] Otung, I. (2001). *Communication engineering principles*. Basingstoke: Palgrave Macmillan, pp. 359–364

[4] ITU-T Recommendation G.726. (1990). *40, 32, 24, 16 kbit/s adaptive differential pulse code modulation (ADPCM)*. Geneva: International Telecommunications Union

[5] Wang, X., Jay Kuo, C.-C. (1998). 'An 800 bps VQ-based LPC voice coder', *Journal of the Acoustical Society of America*, 103(5), p. 2778

[6] Yeldener, S., Kondoz, A., Evans, B. (1994). 'Multiband linear predictive coding at very low bit rates', *IEE Proceedings – Vision Image and Signal Processing*, 141(5), pp. 289–296

[7] ITU-T Recommendation G.729. (2012). Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear prediction (CS-ACELP). Geneva: International Telecommunications Union