1

Introduction and Background

This textbook is about jointly performing source and channel coding for an information source. The idea of coding is arguably the most significant contribution from Claude Shannon in his mathematical theory of communication [1] that gave birth to information theory. Up until Shannon's work, communication engineers believed that to improve reliability in transmitting a message, all they could do was to increase the transmit power or repeat the transmission many times until it was received free of errors. Instead, Shannon postulated that a message could be transmitted free of errors when not exceeding a capacity for the communication channel. All that was required to achieve this feat was to use a suitable coding mechanism. With coding, the message from the source is mapped into a signal that is matched to the channel characteristics. The ideal coding operation is able to both represent the message from the source in the most efficient form (removing redundant information) and add redundant information in a controlled way to enable the receiver to combat errors introduced in the channel. The efficient representation of the source message is called *source coding*, and the controlled introduction of redundant information to combat communication errors is called *channel coding*.

In [1], Shannon showed that under some ideal conditions, for example, for point-to-point "memoryless" channels, asymptotically in the codeword length, there is no performance difference whether the coding operation is performed as a single block or as a concatenation of a source coding operation and a separate channel coding operation. This is typically described as a *separation theorem*. The separation theorems are a class of asymptotic results of profound importance. It may not be an exaggeration to say that those forms of results motivated the development of today's digital revolution, where every form of information – media, gaming, text, video, multidimensional graphics, and data of all forms – can be encoded without worrying about the medium or channel where it will be shared, and shared without the concern for how it was encoded. Nevertheless, there are multiple practical scenarios where the ideal conditions given by Shannon do not hold. In these cases, we pay a performance penalty for doing the source and channel coding as two separate operations instead of jointly doing source and channel coding. This book focuses on some of these scenarios and helps the reader study the theory and techniques associated with performing source and channel coding as a single operation. Before delving into these topics, in this chapter, we provide an introduction and needed background for the rest of this book.

Joint Source-Channel Coding, First Edition. Andres Kwasinski and Vinay Chande.

1.1 Simplified Model for a Communication System

A communication system enables transmission of information so that a message that originates at one place is reproduced exactly, or approximately, at another location. To develop his mathematical theory of communication [1, 2], Claude Shannon considered a simplified model for a communication system, as shown in Figure 1.1. In this model, a communication system is formed by five basic elements:

- An *information source*, which generates a message that contains some information to be transmitted to the receiving end of the communication chain
- A *transmitter*, which converts the message into a signal that can propagate through a communication medium
- A *channel*, which constitutes the medium through which the signal propagates between the transmitter and receiver. This propagating signal is subject to different distortions and impairments, such as selective attenuation, delays, erasure, and the addition of noise.
- A *receiver*, which attempts to recover the original message (possibly affected by distortion and impairments) by reversing the sequence of operations done at the transmitter while also attempting to correct or compensate for the distortion effects introduced in the channel
- A *destination*, which receives the transmitted version of the message and makes sense of it.

The design of a communication system focuses on the transmitter and receiver. By designing the transmitter output to match the channel characteristics and requirements, the transmitter converts, or maps, the source message into a signal appropriate for transmission. This mapping operation is called *encoding*. The reverse operation, where a source message is estimated from the encoded signal, is called *decoding*. For reasons that will be seen later, the mapping operation at the encoder is frequently divided into two stages. The first stage, called *source encoding*, aims to represent the source output in a compact and efficient way that will require as few communication resources as possible while achieving some level of fidelity for the source message recovered after source decoding at the receiver. The compact and efficient representation that results from source encoding is matched

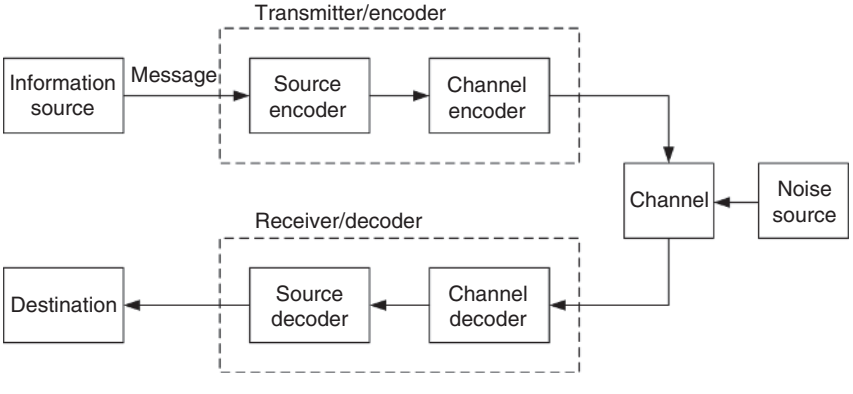

Figure 1.1 A block diagram of a general communication system.

to the source but is likely not matched to the channel characteristics and requirements. For example, the representation may be so compact that each of its parts may be critical for the recovery of the source message at the required level of fidelity, so any error introduced during transmission renders the received message completely unrecoverable. Because of this, the second stage of the transmitter, called *channel encoding*, has the function of converting the compact source representation into a signal matched to the channel. This likely results in a representation that is less compact but more resilient to the effects of channel impairments. On the receiver side, it is natural to think of the structure also separated into a sequence of decoding stages because the goal is to reverse the sequence of encoding operations that were performed at the transmitter.

Designing a communication system entails designing the transmitter and thus the mapper from the source message to the channel signal. Would it be better to design the mapping as a single operation from source directly to the channel? Or, would it be better to design the mapping as a first source encoding stage followed by the channel encoding stage? Is there any performance advantage with either approach? One answer to these questions is given in an asymptotic sense by Shannon's source–channel separation theorem, which states that under some conditions, there is no difference in performance between a transmitter designed as a single mapper and a transmitter designed as the two cascaded source and channel encoding stages. This result is appealing from a designer's perspective. It appears to be a simpler divide-and-conquer approach, involving design of the source and channel coder–decoder pairs (*codecs*) separately. Nevertheless, the tricky element of Shannon's source–channel separation theorem is that the conditions under which it holds are difficult to find in practical scenarios. To discuss this, we first need to establish some key principles from information theory. The next sections provide an overview of important information theory concepts that will be used throughout the rest of this book.

1.2 Entropy and Information

The concept of *information* as understood in Shannon's information theory originates in Hartley's paper [3] and provides a measure on how much the uncertainty associated with a random phenomenon (mathematically characterized as the outcome of a random variable) is reduced by the observation of a particular outcome. The information provided by the outcome x of a discrete random variable X is defined as:

$$I_X(x) = \log \frac{1}{P[X = x]} = -\log P[X = x], \tag{1.1}$$

where $P[X = x]$ is the probability of the outcome $X = x$ and the logarithm can be of any base in principle, but it is most frequently taken as base 2 (in which case the unit of information is the *bit*, a condensed merging of the words *binary* and *digit*). Intuitively, the rarer an event is, the more information its occurrence provides.

The notion of information as introduced in (1.1) provides a measure that is related to a specific outcome of a random variable. To measure the uncertainty associated with a random variable, Shannon introduced the concept of *entropy* (in an information theoretic

context) as the expectation of the information function associated with a random variable. Then the entropy $H(X)$ of a discrete random variable X is defined as:

$$H(X) = E[I_X(x)] = -\sum_{x \in X} P[X = x] \log P[X = x], \tag{1.2}$$

where $P[X = x]$ is, in this context, the probability mass function (PMF). The entropy of a random variable can be interpreted as the mean value of the information provided by all its outcomes.

In the case of a continuous random variable with probability density function (PDF) $f_X(x)$, the concept of entropy has been extended to become the *differential entropy*:

$$H(X) = E[-\log f_X(x)] = \int_{-\infty}^{\infty} f_X(x) \log \frac{1}{f_X(x)} dx. \tag{1.3}$$

For example, it can be shown that differential entropy of a zero-mean Gaussian random variable with variance σ^2 is $(1/2)\log(2\pi e\sigma^2)$ [4].

By using joint probability distributions, one can extend the definition of entropy to calculate the joint entropy between multiple random variables. In the case of two discrete random variables, X and Y, their *joint entropy* is

$$\begin{aligned} H(X, Y) &= -E\left[\log P[X = x, Y = y]\right] \\ &= -\sum_{x \in X} \sum_{y \in Y} P[X = x, Y = y] \log P[X = x, Y = y], \end{aligned} \tag{1.4}$$

where $P[X = x, Y = y]$ is the joint PMF and $P[X = x]$ and $P[Y = y]$ are marginal PMFs. Similarly, the *conditional entropy* of X, $H(X|Y)$ is

$$H(X|Y) = -\sum_{x \in X} \sum_{y \in Y} P[X = x, Y = y] \log P[X = x|Y = y], \tag{1.5}$$

where $P[X = x|Y = y]$ is the conditional PMF of X given that $Y = y$. Consequently, the conditional entropy can be regarded as the mean value of the information provided by all the outcomes of a random variable (X) given that the outcome of a second random variable (Y) is known.

The entropy presents a number of useful properties that in the interest of maintaining the introductory nature of this chapter we enumerate next without a detailed proof:

1. Entropy is nonnegative: $H(X) \geq 0$.
2. $H(X) = 0$ if and only if there exists a value x_0 such that $X = x_0$ almost surely (that is, the random variable X behaves as a deterministic constant).
3. Distributions with maximum entropy: $H(X) \leq \log n$ if the discrete random variable X takes with nonzero probability $n < \infty$ different values. Equality, $H(X) = \log n$, is achieved if and only if X is uniformly distributed. For a continuous random variable X with variance σ_X^2, we have the differential entropy $H(X) \leq (1/2)\log(2\pi e\sigma_X^2)$, with equality achieved when X is a Gaussian random variable.
4. Subadditivity of joint entropy: $H(X) + H(Y) \geq H(X, Y)$ with equality if and only if X and Y are statistically independent random variables.
5. $H(X, Y) \geq H(X)$.
6. Conditional entropy is the remaining uncertainty: $H(X|Y) = H(X, Y) - H(Y)$. This property states that conditional entropy measures how much uncertainty about a random variable (X) remains after knowing the outcome of a second random variable (Y).

7. Nonnegativity of conditional entropy: $H(X|Y) \geq 0$.
8. Chain rule for Shannon's entropy and conditional entropy:

$$H(X_1, X_2, \dots, X_n) = H(X_1) + \sum_{i=2}^{n} H(X_i|X_1, \dots, X_{i-1}). \tag{1.6}$$

   In a simpler form: $H(X, Y) = H(X) + H(Y|X) = H(Y) + H(X|Y)$.
9. Conditioning reduces entropy: $H(X) > H(X|Y)$.

Communication is inherently a process relating two or more random variables to each other (e.g. the input and output of a channel, an uncompressed and a compressed representation of a signal). A quantity that can measure the closeness of two random variables in terms of information shared between them is the *mutual information*. For two discrete random variables X and Y, it is defined as:

$$I(X;Y) = \sum_{x\in X}\sum_{y\in Y} P[X = x, Y = y] \log \frac{P[X = x, Y = y]}{P[X = x]P[Y = y]}, \tag{1.7}$$

Following Bayes's theorem ($P[X = x, Y = y] = P[X = x|Y = y]P[Y = y]$), the mutual information can also be written as

$$I(X;Y) = \sum_{x\in X}\sum_{y\in Y} P[X = x, Y = y] \log \frac{P[X = x|Y = y]}{P[X = x]}.$$

We can also write

$$\begin{aligned} I(X;Y) &= -\sum_{x\in X} \log P[X = x] \sum_{y\in Y} P[X = x, Y = y] \\ &\quad + \sum_{x\in X}\sum_{y\in Y} P[X = x, Y = y] \log P[X = x|Y = y] \\ &= -\sum_{x\in X} P[X = x] \log P[X = x] \\ &\quad + \sum_{x\in X}\sum_{y\in Y} P[X = x, Y = y] \log P[X = x|Y = y]. \end{aligned} \tag{1.8}$$

Note that the first term in this result is the entropy of the random variable X and the second term can be written in terms of the conditional entropy of X. Therefore, the mutual information as in (1.8) can now be written as:

$$I(X;Y) = H(X) - H(X|Y).$$

This quantity has an intuitive interpretation as follows. If $H(X)$ was the measure of uncertainty in random variable X and $H(X|Y)$ is the remaining uncertainty in X on observing (i.e. on learning of the outcome about) Y, then $I(X;Y)$ is the amount of uncertainty reduction about X on observing Y. $I(X;Y)$, therefore, is the information about X contained in or shared by Y. By its definition, it is seen that the quantity is symmetric (i.e. $I(X;Y) = I(Y;X)$). Therefore, it is called *Mutual Information* between two random variables. The analogous quantity of mutual information can be defined for continuous random variables too – by replacing the notion of entropy and conditional entropy by differential entropy and conditional differential entropy, respectively.

Analogous to the introduction of conditional entropy, we can also think of a *conditional mutual information* between two random variables X and Y given that the outcome of a

random variable Z is known. This conditional mutual information, denoted $I(X;Y|Z)$, is defined as:

$$I(X;Y|Z) = H(X|Z) - H(X|Y,Z). \tag{1.9}$$

The conditional mutual information can be understood as being the average amount of information shared by two random variables after a third random variable is known. In this case, the third random variable is often interpreted as representing side information that has been revealed through some process. The conditional mutual information allows for the definition of a chain rule to calculate the mutual information between a random variable X and two other random variables Y and Z:

$$I(X;Y,Z) = I(X;Z) + I(X;Y|Z) = I(X;Y) + I(X;Z|Y), \tag{1.10}$$

or in a more general case when having n random variables $X_1, X_2, \dots, X_n$,

$$I(X_1, X_2, \dots, X_n; Y) = \sum_{i=1}^{n} I(X_i; Y|X_{i-1}, \dots, X_1). \tag{1.11}$$

1.3 Introduction to Source Coding

The communication process starts with a source generating a message intended for transmission. The message may be of very different types. For example, it may be a text (a sequence of letters and symbols), a file residing in a computer, or somebody speaking. In all cases, the message is just a container of information that needs to be transmitted. Messages of all types usually need to go through a step that efficiently represents the information contained in the message. Further, transmission of a message through a communication system as conceptualized by Shannon requires a few extra steps when the message is formed by a continuous-time signal. All these steps are collectively called "source coding." We provide next a brief overview on source coding, leaving for Chapter 2 to cover more details.

1.3.1 Sampling and Quantization of Signals

Today's "digital" age derives its name from the revolution brought about by digital representation of information – which permits unprecedented flexibility in storage, reproduction, computing, and communication. When the source of information is "analog" (that is, continuous-time and continuous-amplitude), the analog signals need to be converted to a digital form for transmission or storage. The first steps in this process are sampling and quantization.

The conversion of an analog signal into a digital form starts with a sampling operation. If the signal is a function of time, sampling consists in recording samples of the continuous-time analog signal, usually at constant time intervals. (If the signal is visual, such as an analog photograph, the sampling occurs at regular spatial intervals, but the basic principles explained here remain the same.) The process is illustrated in Figure 1.2. The left plot shows a speech signal lasting about 3 s. The plots in the middle and right

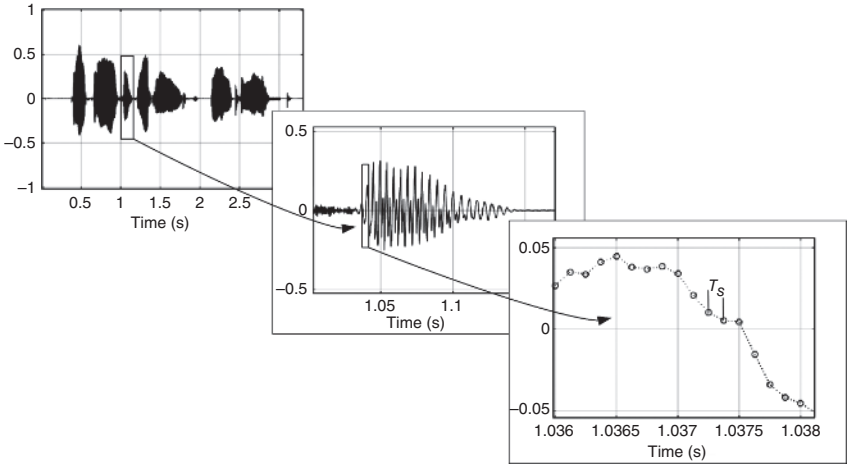

Figure 1.2 Illustration of the sampling process.

show a progressive zooming in to a portion of the signal. The signals in the first two plots can be regarded as continuous time; the signal has an infinite number of values no matter how small a time interval we examine. In the third plot, the signal is made discrete time by taking samples (the circles) of the infinite number of values in the continuous-time signal (the dotted line). For example, the first five samples in the portion of the discrete-time signal shown in Figure 1.2 are $\{0.027, 0.035, 0.034, 0.041, 0.045\}$. The samples are taken at a constant time interval, which is called the *sampling period*, denoted T_S in Figure 1.2. The inverse of the sampling period is called the *sampling frequency*, $f_S = 1/T_S$.

An idealized version of the sampling operation can be modeled as a multiplication of a continuous-time signal $x(t)$ and an impulse train of infinite duration, $p(t)$, resulting in a still continuous-time signal $x_s(t)$ that represents the sampled signal $x(t)$ after sampling:

$$x_s(t) = x(t)p(t) = x(t) \sum_{n=-\infty}^{\infty} \delta(t - nT_S).$$

The final conversion into a discrete-time sequence $x[n]$ of samples from $x(t)$ is achieved by passing the signal $x_s(t)$ through an ideal processing block that selects the values of the signal $x_s(t)$ at the instants when the impulses occur, resulting in a sequence with values $x[n] = x(nT_s)$ at discrete times $n \in \mathbb{Z}$.

The continuous-time signal $x_s(t)$ is an idealization that is useful for understanding the process of sampling and the relationship between samples and the sampled signal. An initial question of interest is how to choose the sampling period or the sampling frequency. The theory of Fourier analysis suggests that the spectrum of the signal $x_s(t)$ is an addition of an infinite number of shifted replicas of the spectrum of the original signal (see Figure 1.3). This arises from the properties that (i) the Fourier transform of an infinite train of impulses in time domain is another train of impulses and (ii) multiplication in the time domain implies a convolution in the frequency domain. Now if the original signal $x(t)$ is band-limited, i.e. its spectrum has nonzero power components at frequencies f restricted to a bounded frequency band $f \in [-B, B]$, then it is possible to recover or reconstruct it

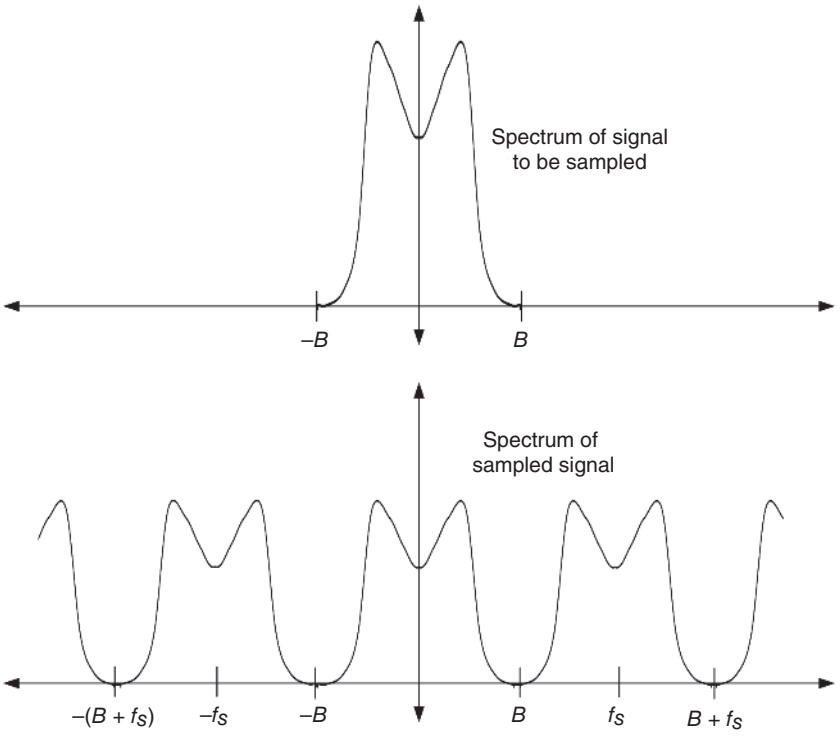

Figure 1.3 The spectrum of a sampled signal.

fully – in continuous time – without distortion from the samples $x[n] = x(nT_s)$, provided the sampling frequency is larger than twice the maximum frequency component in the spectrum of $x(t)$:

$$f_S > 2B.$$

This result, known as the Nyquist–Shannon sampling theorem, determines the conditions for the sampling frequency under ideal assumptions.

In practice, there are multiple factors of non-idealness built into the sampling process. First, it is practically impossible to generate an infinite train of impulses $\delta(t - nT_S)$ that form $x_s(t)$. Second, no real world signal is perfectly band-limited. Third, there is always noise present that prevents the samples from being perfect instantaneous representations of the input signal $x(t)$.

In a practical setting, the choice of sampling frequency needs to account for the effect of noise in the sampled signal, the performance of the filter to limit the sampled signal maximum frequency, the goals for sampled signal quality, and a number of other factors. Therefore, the sampling frequency is frequently set to a value much larger than $2B$. Also note that the Nyquist–Shannon theorem specifies a bound on the sampling frequency based on preventing the overlap of copies of the spectrum of $x(t)$. There are applications that allow the use of sampling at frequencies lower than $2B$, but this subject is out of the scope of this chapter. More information about sampling can be found in digital signal processing textbooks such as [5–7].

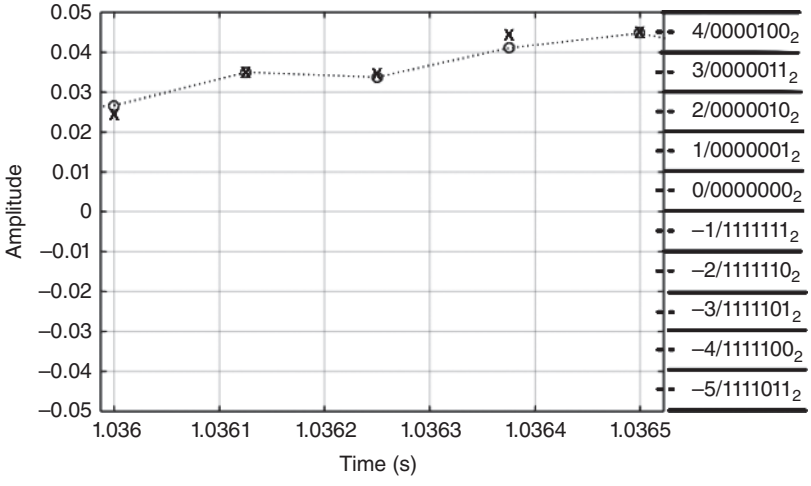

Figure 1.4 Quantization of the five left-most samples in Figure 1.2.

After sampling, the second operation in the conversion of an analog signal into a digital signal is *quantization*. While sampling converts a continuous-time signal into a discrete-time sequence of samples, quantization converts the continuous amplitudes of the samples into values from a discrete set. Figure 1.4 illustrates this operation using the five left-most samples in Figure 1.2. The values of these samples before quantization are real numbers. During quantization, these values are approximated by integer multiples of some value (in Figure 1.4, it was chosen to be 0.01). This operation divides the range of values taken by the input signal into equal-sized (0.01 in this case) intervals, known as *quantization intervals*. For the example in Figure 1.4, using prior knowledge of the signal, the quantizer was designed under the assumption that the samples to be quantized were bounded between ± 0.64 (note that Figure 1.2 shows a small portion of the full signal range). Consequently, the decision to have a quantization interval equal to 0.01 divides the complete range of sample values into $1.28/0.01 = 128$ quantization intervals of equal size. A quantizer with equal-sized intervals is called a *uniform quantizer*. During quantization, the approximation of the real-valued samples to a multiple of the quantization interval size is accomplished by a simple operation of rounding to the closest value. This operation is shown in Figure 1.4 with the original real-valued samples represented as circles and their approximation to the closest interval values as x's. The last step in quantization consists of assigning a label that identifies the quantization interval. In the case of Figure 1.4, a different seven-bit number is assigned to each of the 128 quantization intervals. All samples that are within the same quantization interval are represented by the label of the corresponding quantization interval.

1.3.2 Source Coding of Quantized Signals

If we now pause to reflect on what has been achieved through sampling and quantization, we see that these operations have converted analog signals into discrete-valued functions of discrete-valued variables. The analog source is behaving as a generator of messages that can be broken down into elementary indivisible elements drawn from a set of discrete values

we call *source symbols*. We see an example of this in Figure 1.4, where the source generates messages drawn from a discrete set of 10 source symbols. In Figure 1.4, we labeled these symbols with the numbers "−5," "−4," …, "0," …, "4," but we could just as well have labeled the symbols with the letters "A," "B," …, "J," or even use more cumbersome but wholly descriptive labels such as "Amplitude in the interval [−0.04,−0.03)." Yet, whether we use the label "−4," "B," or "Amplitude in the interval [−0.04,−0.03)," the source symbol remains the same in its elementary indivisible nature (given by the quantization operation). The label itself is just an indicator of the information contained in the symbol, which in this case is that the analog signal sample amplitude is in the interval [−0.04,−0.03). At this point, after quantization, the analog source is behaving as a digital source. A textbook is also a source that generates messages composed of elementary indivisible symbols called "letters," and because of this, it is common to call the source symbols *letters* and call the set of all source symbols the *alphabet*.

As sampling and quantization convert an analog source into a digital source, at this point we will focus exclusively on the common block for all sources called the *source encoder* (and the corresponding *source decoder* on the receiver side). Figure 1.4 shows that the source symbols are represented also by their corresponding two's complement 7-bit binary number (e.g. "−4" is represented by 1111100). This operation of defining the sequence of bits that correspond to each source symbol is called *source encoding*. Formally, source encoding is the operation of mapping the blocks of source symbols (a block may be composed of a single symbol) into sequences of bits, grouped into blocks called *codewords*. The mapping from blocks of source symbols to codewords is called a *source code*.

If we pretend that the alphabet of the source for Figure 1.4 consists of only the 10 symbols shown, the source encoding operation illustrated is wasteful. It uses seven bits when only four would be needed. Moreover, in general, there is no reason why all codewords have to be of the same length; since it is reasonable to assume that not all symbols have the same probability of occurrence, we could strive for more efficiency in the source encoding by mapping more frequent symbols to shorter codewords and rare symbols to longer codewords (as in Morse code). In more formal terms, we would like that the source code will result in a source representation that is efficient in requiring the least number of bits on average (for long sequences of source messages). In the case of *lossless* source coding, the source output is represented with the least number of bits in such a way that after source decoding the reconstructed source message is identical to the original one that was encoded. However, there are cases where it is desirable to have a representation that uses even fewer bits than those needed for the ideal lossless source coding. In this case, perfect reconstruction of the source message after decoding is not possible; some of the information conveyed by the source has been irreplaceably lost. The result presents some level of distortion. This second case is called *lossy* source coding to indicate that some information is lost in the process of source encoding and decoding. Over the next few paragraphs, we examine the minimum number of bits needed to represent the source using lossless source coding, and the magnitude of source distortion when using lossy source coding.

In Figure 1.1, the source encoder compresses the representation of the source with the goal of generating a bitstream that is compatible for transmission over a channel of limited capacity. In the case of lossless coding, the compression operation involves the removal of redundant information from the source, leaving only the source's essential information

content, which is quantified as the source entropy. In the case of lossy compression, the removal of information includes some of the essential information, leading to the introduction of source distortion. The source entropy stems from the stochastic behavior of the source.

A key metric in source coding is the *mean codeword length*, which measures the expected length of a codeword. Let a *sourceword* x^n be a sequence $x_1, x_2, \ldots, x_n$ of source symbols, and suppose that $l(x^n)$ is the length of the codeword resulting from source encoding x^n. Then the mean codeword length is as follows:

$$L_n = E[l(X^n)] = \sum_{x^n} P[X^n = x^n] l(x^n). \tag{1.12}$$

The mean codeword length is measured in units of bits per codeword. When it is multiplied by the rate at which codewords are generated per unit time, we obtain the source code rate in units of bits per unit time (usually bits per second).

The link between the source entropy (that is, the quantification of the source stochastic characteristics we saw in previous sections) and the minimum source code rate needed for its lossless coding stems from a theorem called the *asymptotic equipartition property* (AEP). For the collection of sourcewords x^n of length n, the log-likelihood of the sequence $-\log_2(P[X^n = x^n])$ itself is a random variable.

At its simplest, the information theoretic result, AEP is an application of the law of large numbers to the log-likelihood random variable $-\log_2(P[X^n = x^n])$ for increasing n.

Consider a source (e.g. one in Figure 1.1) that generates an output in the form of a sequence of i.i.d. discrete random variables $X_1, X_2, \ldots, X_n$. Such a source is called a *discrete memoryless source* (DMS). Using $P_X(x) = P[X = x]$ to denote the PMF for any of the random variables in the sequence, the joint PMF for a DMS output sequence (a sourceword) is as follows:

$$P_{X^n}(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} P_X(x_i). \tag{1.13}$$

The AEP studies the asymptotic properties of the information contained in the sequence $X_1, X_2, \ldots, X_n$ as the sequence length grows to infinity. The AEP states that if the sequence $X_1, X_2, \ldots, X_n$ is the output from a DMS with entropy $H(X)$, then for any $\epsilon > 0$,

$$\lim_{n\to\infty} \left\{ \left| -\frac{1}{n} \log_2 P_{X^n}(X_1, X_2, \ldots, X_n) - H(X) \right| > \epsilon \right\} = 0. \tag{1.14}$$

Using $\mathcal{X}^n$ to denote the set of length-n sequences $x_1, x_2, \ldots, x_n$ generated by a DMS, we define the *typical set* $A_\epsilon^{(n)}$ with respect to $P_{X^n}(x^n)$ as the set that contains sequences that meet the property:

$$2^{-n(H(X)+\epsilon)} \le P_{X^n}(x^n) \le 2^{-n(H(X)-\epsilon)}. \tag{1.15}$$

The typical set $A_\epsilon^{(n)}$ has a number of properties that will be useful in characterizing the source code for lossless compression that is optimal in yielding the smallest mean codeword length:

- A first property follows directly from (1.15) and states that all elements of the typical set are nearly equiprobable. Formally, the property indicates that the probability of a

sequence $x^n = (x_1, x_2, \dots, x_n) \in A_\epsilon^{(n)}$ is narrowly squeezed by close-by upper and lower bounds $2^{-n(H(X)-\epsilon)}$ and $2^{-n(H(X)+\epsilon)}$.

- A second property indicates that $P[A_\epsilon^{(n)}] > 1 - \epsilon$ for a sequence length n large enough, which means that the probability of the typical set is almost equal to 1.
- Lastly, a third property states that the cardinality of the typical set, denoted $|A_\epsilon^{(n)}|$, is approximately equal to $2^{nH(X)}$. Mathematically, this property is expressed as $(1-\epsilon)2^{n(H(X)-\epsilon)} \leq |A_\epsilon^{(n)}| \leq 2^{n(H(X)+\epsilon)}$ for n sufficiently large on the lower bound and for all n on the upper bound.

In summary, through the AEP and its derived properties we learn that for source sequences long enough, there is a set of probability almost one composed of nearly $2^{nH(X)}$ length-n sequences that are all approximately equiprobable.

The AEP motivates a conceptually simple source code for the DMS that generates source-words of length n. The source code is constructed by splitting all length-n sequences into those that are and those that are not in the typical set. The codewords for entries in the typical set are of length nearly $nH(X)$ bits. Those not in the typical set are sorted and assigned a code based on the order the sequence occupies in the sorted list.

Following this procedure, it can be shown, [4], that the resulting code is a lossless one-to-one (a decodable) mapping with the property that for sufficiently large n we have that $E[(1/n)l(X^n)] \leq H(X) + \epsilon$. In other words, a sufficiently long sequence of length n can be represented with no loss of information using a code with mean codeword length $nH(X)$.

The AEP and its implications are useful to understand the behavior of the source codes – lower and upper bounds on the mean length achievable – in the limit of the sourceword length going to infinity (that is, for sufficiently long sequences). A part of this conclusion that has important consequences for the subject of this book is that for any source coding involving finite and small length source sequences, the AEP has limited use in code construction.

The AEP as stated earlier applies to sequences of i.i.d. random variables. The notion and the techniques have been extended for sequences with characteristics other than i.i.d. For certain sequences of random variables, it is possible to calculate a magnitude called the *entropy rate*, which leads to results akin to the AEP. For a discrete-time random process $\{X_i\}$, the entropy rate, $H(\mathcal{X})$, or simply the *entropy* of the random process, is defined as:

$$H(\mathcal{X}) = \lim_{n\to\infty} \frac{1}{n} H(X_1, X_2, \dots, X_n), \tag{1.16}$$

when the limit exists. The entropy rate should be thought of as a definition of entropy for general random processes. Consistent with this, when the random process is a sequence of i.i.d. random variables we have that:

$$H(\mathcal{X}) = \lim_{n\to\infty} \frac{1}{n} H(X_1, X_2, \dots, X_n) = \lim_{n\to\infty} \frac{1}{n} nH(X_i) = H(X_i). \tag{1.17}$$

The definition of an entropy rate allows the characterization of a general AEP for a stationary ergodic random process. Specifically, for a stationary ergodic random process $\{X_i\}$, the general AEP states that:

$$-\frac{1}{n}\log_2 P_{X^n}(X_1, X_2, \dots, X_n) \to H(\mathcal{X}) \quad \text{with probability one.} \tag{1.18}$$

Following the parallels between the general AEP and the AEP for an i.i.d. sequence, it can be shown that for a stationary ergodic random process there exists a typical set composed of approximately $2^{nH(\mathcal{X})}$ sequences of length n, which can be represented using approximately $nH(\mathcal{X})$. From here, what is known as the *source coding theorem* follows; it states that for lossless source coding, the minimum mean codeword length L_n^* is upper and lower bounded as

$$\frac{H(X_1, X_2, \dots, X_n)}{n} \leq L_n^* \leq \frac{H(X_1, X_2, \dots, X_n)}{n} + \frac{1}{n}, \tag{1.19}$$

and, as n grows to infinity (source sequences growing infinitely long), we have that $L_n^* \to H(X)$ for a DMS, and $L_n^* \to H(\mathcal{X})$ for a stationary stochastic process.

1.3.3 Distortion and Rate-distortion Theory

The source coding theorem describes the minimum mean codeword length for a source code that would be optimal for lossless coding. In many cases, this minimum mean codeword length (or the one that can actually be achieved under practical conditions as, for example, with small source sequence lengths n) is still too large for a particular application. In these cases, it is necessary to use a source code with a smaller mean codeword length. The use of such source codes, however, comes at the cost of losing information in the encoding. We call this operation lossy source coding. It leads to distortion of the source message recovered after source decoding. We encountered a simple example of this situation in the quantization operation illustrated in Figure 1.4. In this example, information is lost when mapping a sample's value measured with infinite resolution (the analog sample amplitude) into a label drawn from a finite set that identifies quantization intervals. Information is lost and distortion is introduced when all the analog source values belonging to the same quantization interval are mapped to the same reconstruction value during decoding. With fewer quantization intervals, fewer labels are needed to identify each of them, but also the quantization distortion is larger because the range of source analog values associated with each quantization interval has increased. Quantization as a lossy source coding operation will be discussed more in detail in the next chapter. At this point, we merely emphasize that there is a relation between quantization distortion and the alphabet size of the quantized source. Of course, quantization is not the only instance of lossy coding. To transmit a digital source through a communication channel of limited capacity, it is common to do lossy source coding on the digital source by removing information in a controlled way. The removal of some information during source coding leads to a distortion observed in the decoded source. The relation between the amount of information that is kept after source encoding and the distortion of the decoded source is characterized through the *rate-distortion function*.

The first step in the study of the rate-distortion function is the definition of distortion metrics. A distortion metric, measure, or function, is a mapping from the combined domains of source alphabet and reconstruction alphabet into nonnegative real numbers. A distortion metric is a function that takes in as arguments a source symbol x and a source decoded symbol $\hat{x}$ and outputs a measure of the cost associated with representing x by $\hat{x}$. We use $d(x, \hat{x})$ to denote a distortion metric. Two common distortion measures

are the Hamming distortion and the squared-error distortion. The Hamming distortion is defined as:

$$d(x,\hat{x}) = \begin{cases} 0 & \text{if } x = \hat{x}, \\ 1 & \text{if } x \neq \hat{x}. \end{cases} \tag{1.20}$$

The Hamming distortion essentially measures the probability of error because its expected value is $P[x \neq \hat{x}]$. The squared-error distortion is as follows:

$$d(x,\hat{x}) = (x-\hat{x})^2. \tag{1.21}$$

These two distortion metrics measure the distortion between two symbols, the source symbol and its reconstructed version after decoding. However, since source codecs tend to operate on sequences of source symbols, it is more commonplace to encounter the per-symbol distortion metrics used to measure the distortion between the source symbol sequence and the reconstructed version of the source sequence after decoding. For this, the distortion between the two length-n sequences x^n (the source sequence) and $\hat{x}^n$ (its reconstructed version) is defined as:

$$d(x^n,\hat{x}^n) = \frac{1}{n}\sum_{i=1}^{n} d(x_i,\hat{x}_i). \tag{1.22}$$

The distortion between two sequences is the average distortion between their constituent symbols. When the symbol distortion metric is the squared error as in (1.21), the distortion between two sequences is the *mean squared error (MSE)* distortion measure.

In continuing toward the definition of a rate-distortion function, it turns out that the AEP and its relation to lossless representation of a sequence of source symbols can be extended to the case when source distortion is introduced during lossy coding. For pairs of length-n sequences $(x^n,\hat{x}^n)$ of source and reconstructed symbols that are drawn i.i.d. following the joint probability $P_{X^n,\hat{X}^n}(x^n,\hat{x}^n)$ from a support product set $\mathcal{X}\times\hat{\mathcal{X}}$, we can define the *distortion ϵ-typical set* as the set satisfying

$$\begin{aligned} \mathcal{D}_\epsilon^{(n)} = \Bigg\{ & (x^n,\hat{x}^n) \in \mathcal{X}\times\hat{\mathcal{X}} \ : \\ & \left| -\frac{1}{n}\log_2 P_{X^n}(x^n) - H(X) \right| < \epsilon, \\ & \left| -\frac{1}{n}\log_2 P_{\hat{X}^n}(\hat{x}^n) - H(\hat{X}) \right| < \epsilon, \\ & \left| -\frac{1}{n}\log_2 P_{X^n,\hat{X}^n}(x^n,\hat{x}^n) - H(X,\hat{X}) \right| < \epsilon, \\ & \text{and } \left| \frac{1}{n} d(x^n,\hat{x}^n) - E[d(x,\hat{x})] \right| < \epsilon \Bigg\}, \end{aligned} \tag{1.23}$$

where $E[d(x,\hat{x})]$ is the expectation with respect to the probability distribution $P_{X^n,\hat{X}^n}(x^n,\hat{x}^n)$. From this, the following convergence properties hold:

$$\lim_{n\to\infty} \left\{ \left| -\frac{1}{n}\log_2 P_{X^n}(X_1,X_2,\ldots,X_n) - H(X) \right| > \epsilon \right\} = 0 \tag{1.24}$$

$$\lim_{n\to\infty} \left\{ \left| -\frac{1}{n}\log_2 P_{\hat{X}^n}(\hat{X}_1,\hat{X}_2,\ldots,\hat{X}_n) - H(\hat{X}) \right| > \epsilon \right\} = 0 \tag{1.25}$$

$$\lim_{n\to\infty}\left\{\left|-\frac{1}{n}\log_2 P_{X^n,\hat{X}^n}(X_1,\hat{X}_1,\ldots,X_n,\hat{X}_n)-H(X,\hat{X})\right|>\epsilon\right\}=0 \tag{1.26}$$

$$\lim_{n\to\infty}\left\{\left|\frac{1}{n}d(x^n,\hat{x}^n)-E[d(x,\hat{x})]\right|>\epsilon\right\}=0. \tag{1.27}$$

Note that these properties expand the description of the AEP in two ways compared with the characterization of the AEP in the lossless representation case. The first difference is that now the typical set is defined in terms of the joint statistics for two sequences of i.i.d. random variables: the sequence of symbols generated by the source (X) and the sequence of reconstructed symbols after decoding ($\hat{X}$). This extra element of the definition makes this case a *joint* AEP. The second difference is a statement of consistency in the measurement of the distortion between the two sequences of random variables when using the metric defined in (1.22).

Serving as a preliminary step, the AEP for lossy representation leads to *Shannon's rate-distortion theorem* for memoryless sources. The theorem states that for a sequence $X_1, X_2, \ldots, X_n, \ldots$ generated by a DMS with alphabet $\mathcal{X}$ that is encoded and subsequently decoded into a sequence $\hat{X}_1, \hat{X}_2, \ldots, \hat{X}_n, \ldots$ with alphabet $\hat{\mathcal{X}}$, given a distortion measure as in (1.22) satisfying $\max_{(x,\hat{x})\in\mathcal{X}\times\hat{\mathcal{X}}} d(x,\hat{x}) < \infty$ (bounded per-source symbol distortion), the source's rate-distortion function, $R(D)$, is given by:

$$R(D)=\min_{P_{\hat{X}|X}:E[d(X,\hat{X})]\le D} I(X;\hat{X}), \tag{1.28}$$

where $P_{\hat{X}|X}$ denotes the probability distribution of $\hat{X}$ conditioned on X. In essence, the rate-distortion function characterizes the convex hull resulting from reducing as much as possible the average amount of information shared by a source output and its reconstruction after coding and decoding, subject to a maximum distortion. As the rate-distortion function is defined in terms of the mutual information between a source symbol and its reconstruction after decoding, the rate is measured in units of bits per source symbol (the units of bits follows our use of logarithm base 2 when measuring information and entropy).

While in many cases it is not possible to compute the rate-distortion function as a closed-form expression, there are a few notable (and fortunately useful) cases where this is possible. One such case is a binary DMS source, modeled with an alphabet $\mathcal{X}=\{0,1\}$ with X being a Bernoulli random variable with PMF $P[X=1]=p$. Under the Hamming distortion measure, the rate-distortion function is given by:

$$R(D)=\begin{cases} H(p)-H(D), & \text{for } 0\le D\le \min\{p,1-p\},\\ 0, & \text{for } D>\min\{p,1-p\}.\end{cases} \tag{1.29}$$

Figure 1.5 illustrates this rate-distortion function for different values of the parameter p. Due to the symmetry in this rate-distortion function, the cases with $p=0.25$ and $p=0.75$ yield the same function. Of all the choices for p, the case with $p=0.5$ yields the largest value of rate for a given distortion. This is intuitively natural, since we expect that a source exhibiting the largest uncertainty in the symbol that it generates will require the largest rate to represent its output for the same level of source encoding distortion. When distortion is made equal to zero, this becomes the lossless coding case, and the rate is equal to $H(p)$, consistent with the previously discussed source coding theorem.

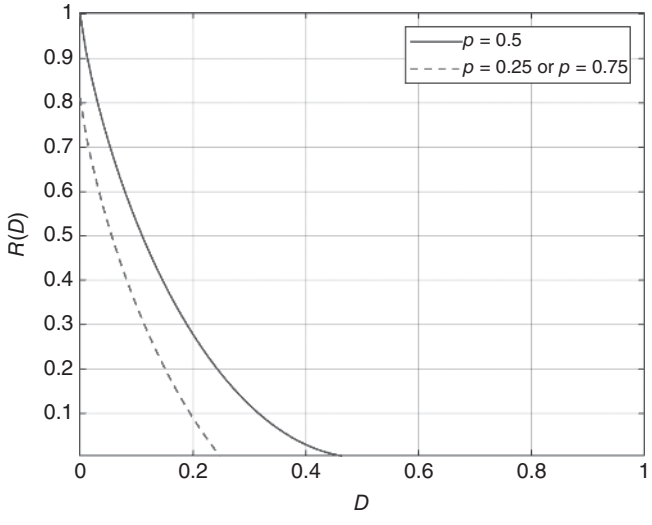

Figure 1.5 Rate-distortion function for binary sources following a Bernoulli distribution with parameter p.

While in this section we have so far focused on sources with a discrete alphabet and assumed a bounded distortion measure, the definition of the rate-distortion function in (1.28) can be extended to sources with a continuous alphabet and unbounded distortion measures. We could encounter this situation when the source output consists of samples from an analog signal. In this case, the rate-distortion function will characterize the source encoding operation as encompassing the quantization operation and the proper source encoding operation all in one operation block. A useful member of the class of sources with continuous alphabets is the Gaussian source. The Gaussian source output consists of i.i.d. samples from a random process that follows a Gaussian distribution with mean μ and variance σ^2. Using a squared-error distortion, the rate-distortion function for a zero mean, variance σ^2, Gaussian ($\mathcal{N}(0, \sigma^2)$) source is given as follows:

$$R(D) = \begin{cases} \frac{1}{2}\log_2\left(\frac{\sigma^2}{D}\right), & \text{for } 0 \le D \le \sigma^2, \\ 0, & \text{for } D > \sigma^2. \end{cases} \tag{1.30}$$

It is often useful to work with the inverse of the rate-distortion function, that is, a function which tells the distortion that will be expected when encoding at a certain rate. This inverse function, called the *distortion-rate function*, is:

$$D(R) = \sigma^2 2^{-2R} \tag{1.31}$$

for the case of the i.i.d. Gaussian source. From this expression, we learn that the distortion is reduced by a factor of 4 with each one-bit increase in the source coding rate. This improvement is independent of the variance of the source. In fact, considering the variance as the power of the source signal, we can see in (1.31) that source coding rate R results in a distortion that equals the source power divided by the rate R-power of 4. Figure 1.6 illustrates rate-distortion and distortion-rate functions for Gaussian sources with different variances.

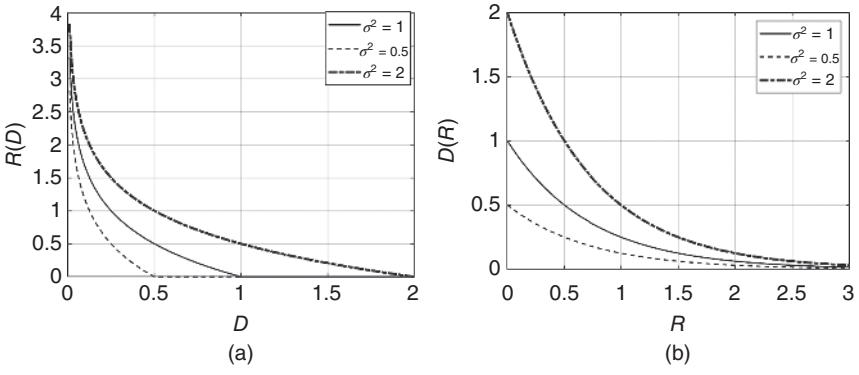

Figure 1.6 (a) Rate-distortion functions and (b) distortion-rate functions for Gaussian sources of different variances.

Now, recall from (1.28) that the functions $R(D)$ or $D(R)$ are defined based on the minimum rate to achieve a limit on distortion. There are many source codes that will not perform as well as the one associated with the rate-distortion function and will need a higher rate to achieve the same distortion. It can be shown that the limit performance characterized by the rate-distortion function is achieved when using source codes on very long blocks of source symbols. We encountered this condition for achieving a limit performance earlier: the AEP is achievable in the limit of very long sequences. This recurrent condition of achievability is of note; it leads to the main motivation for using joint source–channel coding techniques.

1.4 Channels, Channel Coding, and Capacity

One of the important elements in Shannon's model for a communication system (Figure 1.1) is the channel over which the transmission of information between transmitter and receiver takes place. As the source–channel coding operation maps source messages into signals appropriate for the channel characteristics, the channel plays a critical role in the source–channel codec design. Many different channel models exist; they differ by the physical conditions they model, their level of abstraction, or their level of simplifying assumptions. This section provides a brief introduction to a selected subset of important channel models.

1.4.1 Channel Models

A common channel model involves noise added to the transmitted signal; the channel output is written as $Y = X + Z$, where Y, X, and Z are the samples of, respectively, the channel output signal, the channel input signal, and the additive noise signal. The use of signal samples implies that this is a discrete-time model. The noise is often modeled as a random process originating from the combined effect of multiple noise sources (e.g. thermal noise at the receiver's multiple electronic components, background radiation), in which case it is assumed that each noise sample follows a Gaussian distribution. Furthermore, it is

assumed that noise samples are independent from the channel input sample and that at different times they are also uncorrelated from each other or, equivalently, the noise process power spectrum density is constant across all frequencies on which the receiver operates. Consequently, this channel model is called the *additive white Gaussian noise* (AWGN) model. This model only accounts for the effects of additive noise and does not consider the effects of signal attenuation or interference, for example.

Other models take an approach that abstracts the effects of the channel on the transmitted information. One such case is the *discrete memoryless channel* (DMC). Assume that the input to the channel is a discrete random variable X drawn from a set of symbols represented as letters from the alphabet $\mathcal{A}_I = \{x_0, x_1, x_2, \dots, x_M\}$ and the output is a discrete random variable Y from a set given by the alphabet $\mathcal{A}_O = \{y_0, y_1, y_2, \dots, y_N\}$. The DMC is described by the collection of conditional probabilities $P(Y = y_O|X = x_I)$ that characterize the probability of measuring at the channel output the letter $y_O \in \mathcal{A}_O$, given that the channel input was $x_I \in \mathcal{A}_I$. The "memoryless" in the model's name indicates that when the input is a sequence of letters, the channel affects the transmission of each letter independently of each other. Because of this, if the input sequence is $\vec{X} = \{X_0, X_1, \dots, X_K\}$ and the output sequence is $\vec{Y} = \{Y_0, Y_1, \dots, Y_K\}$, the conditional probability $P(\vec{Y}|\vec{X})$ that describes the channel effect can be calculated as:

$$P(\vec{Y}|\vec{X}) = \prod_{i=0}^{K} P(Y_i|X_i). \tag{1.32}$$

When transmitting a single symbol at a time, and when the input alphabet size M is equal to the output alphabet size N, the DMC can be used to model situations where after demodulating and detecting a block of $\log_2 M$ received bits there is a probability (given by the conditional probabilities $P(Y_i = y_O|X = x_I)$) that some of these bits may be estimated with a value different from the transmitted ones. An important case of this model is when the input/output alphabets have two possible values, which are usually the binary digits "0" and "1." It is usually assumed that the probability of introducing an error during the transmission over the channel is the same regardless if a "0" or a "1" was transmitted (that is, $P(Y_i = \text{'0'}|X = \text{'1'}) = P(Y_i = \text{'1'}|X = \text{'0'})$). This model is called the *binary symmetric channel* (BSC) and is illustrated in Figure 1.7a, where ϵ is the probability of making a receiver decision error on the transmitted bit (a probability that often is considered equivalent to the *bit error rate* [BER]).

The case of the DMC when $N = M + 1$ allows us to model what is known as an *erasure* channel by introducing an extra letter in the output alphabet. Rather than being a true letter, this addition to the output alphabet can be seen as a state of the receiver operation where there has been a transmission error and the received symbol is discarded (that is, erased). The simplest case, illustrated in Figure 1.7b is the *binary erasure channel* (BEC), where the input alphabet has two letters ("0" and "1") and the output alphabet has three letters ("0," "1," and "e" for "erasure"). Figure 1.7b shows a general case for the BEC case where the erasure probability is different depending on whether the transmitted symbol was a "0" or a "1" (probabilities equal to α and β, respectively). If the two probabilities are equal, the channel becomes a binary symmetric erasure channel.

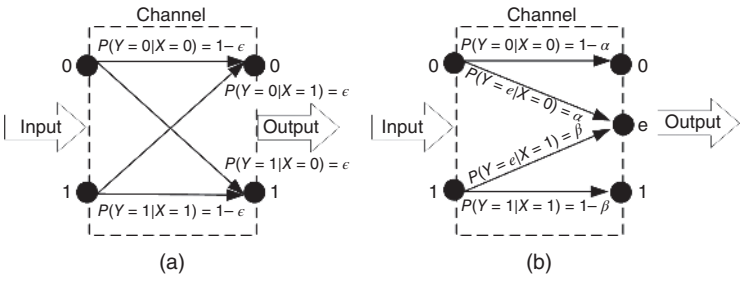

Figure 1.7 (a) The binary symmetric channel and (b) the binary erasure channel.

1.4.2 Wireless Channels

Wireless channels are affected by noise, often modeled as AWGN. In addition, wireless propagation of electromagnetic – radio – signals is characterized by complex physical phenomena of reflection, refraction, absorption, scattering, and diffraction from surrounding environmental components that lead to unknown and varying attenuation, or *fading*. The characterization of fading is divided into *large-scale propagation effects* and *small-scale propagation effects*. These refer to the scale of distance over which the effect is noticeable. This distance is relative to the radio signal wavelength (which is related to the signal frequency by $\lambda = c/f$, where λ is the wavelength, f is the frequency, and $c = 3 \times 10^8$ m/s is the velocity of light in a vacuum). Small-scale fading creates differences that are noticeable at distances in the order of half the wavelength, while large-scale effects are noticeable at distances of several times the wavelength.

Large-scale Propagation Effects

Three large-scale propagation effects are usually considered, with each of them causing a loss of power in the received radio signal. The effects are measured as the ratio between the transmitted and received signal power. The first of these large-scale propagation effects is the *path loss*. For a radio signal propagating in free space, the path loss effect arises because the energy of the emitted radio wave is spread over a surface of space that expands as the radio wave propagates away from the emitter. After measurement at a receiver, the net effect is an attenuation of the signal. A radio wave emitted or received near the Earth's surface is more difficult to model because of the presence of copies of the emitted signal that arrive to the receiver after reflecting off surfaces or for other reasons. In all these cases, the received power is attenuated as a function of the distance from the transmitter at a larger rate than that of free space propagation. In general, for a system comprising a transmitter and a receiver each with a single omnidirectional antenna, the path loss is typically modeled as a function of the following form:

$$\Gamma_{dB} = 10\nu \log(d/d_0) + C_p, \tag{1.33}$$

where Γ_{dB} is the path loss Γ measured in decibels (dB), d is the distance between transmitter and receiver, ν is the path loss exponent, C_p is a constant, and d_0 is the distance to a power measurement reference point (sometimes embedded within the constant C_p). In many practical scenarios, this expression is not an exact characterization of the path loss but is used as a sufficiently good and simple approximation. The path loss exponent ν is a

statistical characterization of the rate that signal power decays with distance, taking values in the range of 2 (corresponding to free space decay) to 6. The magnitude of the path loss exponent, which captures the effects of reflection, refraction, and scattering, is also roughly a function of the relative size of the "clutter" – the elements contributing to fading – and the wavelength of the signal. In a city environment, typical values for the path loss exponent are 4 for distances of several hundred meters and 3 for shorter distances for radio frequencies of the order of a few gigahertz. The constant C_p includes parameters related to the physical setup of the transmission such as signal wavelength and antennas height.

A second path loss effect characterizes the loss of signal power when propagating through sizeable objects in the path between the transmit and receive antennas. This impairment is named *shadowing loss* or *shadow fading*. Since the nature and location of the obstructions causing shadow loss cannot be known in advance, the path loss introduced by this effect is modeled as a random variable, typically with a log-normal distribution (a random variable that follows a Gaussian distribution when measured in decibels). Together the path and shadowing losses are modeled through the power loss expression:

$$\Gamma_{dB} = 10v \log(d/d_0) + S + C_p, \tag{1.34}$$

where S denotes the shadowing loss (a zero-mean Gaussian random variable because the power loss expression is given in decibels).

In a particular case when at least the receiver is indoors and one wants to include in the model a more detailed characterization of the wireless signal propagating into the building, a third term is added to (1.34) yielding:

$$\Gamma_{dB} = 10v \log(d/d_0) + S + L_p + C_p. \tag{1.35}$$

The new term L_p is called the *penetration loss* and models, usually as a constant value, the attenuation in the received radio signal power after penetrating a building (following assumptions of the structure and materials for the walls of the building).

Small-scale Propagation Effects

Small-scale propagation effects model differences in radio signal fading that can be noticed at the scale of a wavelength. Often called small-scale fading, it arises because radio signals arrive at the receive antennas after being affected by countless random reflectors, scatterers, and attenuators encountered during propagation. Radio signals do not follow a single direct path between transmitter and receiver. Instead, multiple copies of the transmitted signal arrive at the receiver antenna after each having followed a different path. Because paths are of different lengths, the copies of the transmitted radio signal arrive at the receiver at slightly different times. Such a channel where a transmitted signal arrives at the receiver with multiple copies is known as a *multipath channel*. Several factors influence the behavior of a multipath channel. One is the random presence of reflectors, scatterers, and attenuators. The speed of the mobile terminal, the speed of surrounding objects, and the transmission bandwidth of the signal are other factors determining the behavior of the channel. If there is motion of the transmitter, the receiver, or the surrounding objects, the multipath channel changes over time. The effect of multiple copies of the radio signal arriving at the

receiver is modeled as a linear system (a time-varying tap-delay line) through the following expression:

$$y(t) = \sum_{i=1}^{L} h_i(t)x(t - \tau_i(t)), \tag{1.36}$$

where $y(t)$ is the received signal, $x(t)$ is the transmitted signal, $h_i(t)$ is the attenuation of the ith path at time t, $\tau_i(t)$ is the corresponding path delay, and L is the number of resolvable paths at the receiver. As can be seen in (1.36), because each of the multiple copies of the transmitted signal is received with a different amplitude, phase, and delay, when they are combined at the receiver they create patterns of constructive or destructive interference between them. As (1.36) is a linear model, the channel impulse response is given as follows:

$$h(t, \tau) = \sum_{i=1}^{L} h_i(t)\delta(t - \tau_i(t)). \tag{1.37}$$

In many scenarios, it is possible to assume that the channel will not change for the duration of the transmission of interest. In this case, (1.36) and (1.37) can be written in a simpler, not time-varying form, as follows:

$$y(t) = \sum_{i=1}^{L} h_i x(t - \tau_i), \tag{1.38}$$

$$h(t) = \sum_{i=1}^{L} h_i \delta(t - \tau_i). \tag{1.39}$$

When studying digital communication systems, it is convenient to convert the channel characterization into a discrete-time baseband-equivalent model, for which the input–output relation derived from (1.36) can be written as:

$$y[m] = \sum_{k=l}^{L} h_k[m]x[m - k], \tag{1.40}$$

where $x[m]$ is the discrete-time equivalent of the transmitted signal (a sampled version of $x(t)$), $y[m]$ is the discrete-time equivalent of the received signal, and $h_k[m]$ are the channel coefficients. This is the same type of modeling seen in Figure 1.1, although the model considered here has gained in mathematical detail. The conversion to a discrete-time model combines all the paths that have an arrival time within one sampling period into a single-channel response coefficient $h_l[m]$. Since the nature of each path, its length and the presence of reflectors, scatterers, and attenuators are all random, the channel coefficients h_k of a time-invariant channel are random variables (and note that the redundant time index need not be specified). If, in addition, the channel changes randomly over time, then the channel coefficients $h_k[m]$ are random processes. Channel coefficients could be considered non-time-dependent when the duration of a transmission is shorter than the time it takes for the channel to exhibit noticeable changes. The rate at which the channel (and its impulse response coefficients) may be changing over time depends on

the radio environment settings for the transmitter, the receiver, and objects that affect the multiple signal paths. In particular, a major factor influencing the change over time of a channel characteristic is the velocity at which the transmitter, the receiver, and the objects engendering the multiple signal paths are moving with respect to each other. The duration of time over which the channel coefficients can be considered to practically remain unchanged is quantified through the *channel coherence time*. The channel coherence time is the time difference that makes the correlation between two realizations of the channel impulse response be approximately zero.

If we assume that the channel is not changing over time (that is, we consider the channel model over a time shorter than the coherence time), we can write (1.40) based on the non-time-varying channel impulse response coefficients h_k as:

$$y[m] = \sum_{k=l}^{L} h_k x[m-k]. \tag{1.41}$$

Multipath effects arise from combining copies of the same signal that arrive at the receiver through paths of different lengths. Because of the path length differences, the copies arrive with different attenuations and phase shifts. This effect is reflected in (1.41) by modeling the channel impulse coefficients h_k as complex-valued random variables. The coefficients can be represented using Cartesian coordinates as $h = h_I + jh_Q$ (where h_I and h_Q are the in-phase and quadrature-phase components, respectively) or using polar coordinates as $h = r\, e^{j\theta}$ (where r is the coefficient magnitude and θ the phase).

Several models exist to characterize the statistical properties of the coefficients. One of the most common models for the random channel coefficients, known as the *Uniform Scattering Environment*, is derived from an assumed configuration for the transmitter, the receiver, and the objects that affects the multiple signal paths. This model is also known as Clarke's model or Jakes' model in honor of R.H. Clarke who first introduced it and W.C. Jakes who developed it. The model assumes that the radio signal arrives at a receiver after being scattered by a very large number of scatterers randomly located on a circle centered on the receiver. The radius of this circle is much smaller than the distance between transmitter and receiver. As a result, the received radio signal (1.41) is made of the superposition of many copies of the transmitted signal arriving from the scatterers at angles that are uniformly distributed between 0 and 2π. The model assumes that there is no radio signal that is received through a direct path between transmitter and receiver (known as the line-of-sight [LOS]). It can be shown [8] that under the uniform scattering model, the in-phase and quadrature-phase components of the coefficients are each a zero-mean Gaussian random variable all with the same variance σ^2, which means that the coefficients themselves are circularly symmetric complex Gaussian random variables with zero mean and variance $2\sigma^2$. By applying a change of variables to convert the coordinate system from Cartesian to polar, the phase of a channel coefficient θ is a uniform random variable taking values between 0 and 2π and the magnitude of the channel coefficient (the amplitude gain for the corresponding path), r, is a Rayleigh random variable with PDF given by:

$$f(r) = \frac{r}{\sigma^2} e^{-r^2/(2\sigma^2)}, \quad r \geq 0. \tag{1.42}$$

Because the magnitude of the channel coefficients follows a Rayleigh distribution, this model is frequently called a *Rayleigh fading* model.

Since the power gain for a path is a random variable $G = r^2$, its PDF can be calculated by applying the transformation of random variables theory and find that it is distributed as an exponential random variable with PDF:

$$f_G(g) = \frac{1}{\sigma^2} e^{-g/\sigma^2}, \quad g \geq 0. \tag{1.43}$$

Moreover, the sum of the squared magnitudes of the channel coefficients, $\sum_i |h_i|^2$, results in a Chi-square random variable with $2L$ degrees of freedom, where L is the number of channel coefficients in the sum. The PDF of this distribution is as follows:

$$f(x) = \frac{x^{L-1}}{(L-1)!} e^{-x}, \quad x \geq 0. \tag{1.44}$$

If we assume that the uniform scattering environment model also includes an LOS path, approximating both the in-phase and quadrature-phase components of a channel coefficient as zero-mean Gaussian random variables is no longer valid. Now, the two components are Gaussian random variables but one has mean A, which is the peak amplitude of the signal from the LOS path, and the other still has mean zero. The analysis for this case leads to the conclusion that the magnitude of a channel coefficient follows a Rician distribution with a PDF given by:

$$f_r(z) = \frac{z}{\sigma^2} e^{-\left(\frac{z^2}{2\sigma^2} + K\right)} I_0\left(\frac{2Kx}{A}\right), \quad z \geq 0, \tag{1.45}$$

where $I_0(x)$ is the modified Bessel function of the first kind and zero order, defined as:

$$I_0(x) = \frac{1}{2\pi} \int_0^{2\pi} e^{x\cos\theta} d\theta. \tag{1.46}$$

In (1.45), K is a parameter of the Rician distribution, defined as $K = A^2/(2\sigma^2)$. When $K = 0$, the Rician PDF becomes equal to the Rayleigh PDF, consistent with the fact that $K = 0$ means there is no LOS path.

1.4.3 Channel Coding and Channel Capacity

The revolutionary innovation that Shannon introduced with his mathematical theory of communication was the use of coding (the mapping from a source message into an appropriate signal for the channel characteristics) to improve the communication reliability of a message. The central result of Shannon's mathematical theory of communication is an indisputable justification for the use of coding: when the code used follows certain characteristics in relation to the channel, it is possible to achieve an arbitrarily small probability of error when transmitting a message. This central result, known as Shannon's noisy channel coding theorem, [1], is the subject of this section.

Figure 1.8, which emphasizes the encoding and decoding operations as well as their relation with the channel, is a simplified version of the general communication system diagram in Figure 1.1. We assume that the source is a DMS and that, without loss of generality, its alphabet is the set of indexes $\{1, 2, \ldots, M\}$. A message W formed by a single letter (index) from the source is encoded, resulting in a codeword $\vec{X}(W) = \{X_0, X_1, \ldots, X_n\}$. This codeword is transmitted through the channel and received as a sequence $\vec{Y} = \{Y_0, Y_1, \ldots, Y_n\}$, which, because of the stochastic impairments introduced into the transmitted signal

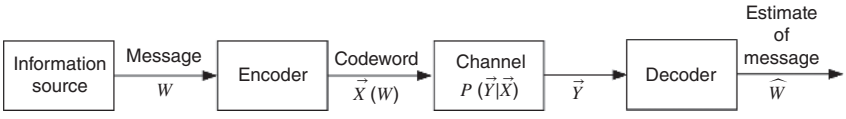

Figure 1.8 Distilled view of the message communication process.

by the channel, is a random sequence that follows the probability distribution $P(\vec{Y}|\vec{X})$. The received sequence $\vec{Y}$ is input to a decoder that outputs an estimate $\hat{W}$ of the transmitted message. We can describe this process with an emphasis on the role of the coding and its mathematical formulation. Given an index set $\{1, 2, \dots, M\}$, the encoding operation of an (M, n) code is a function $X^n : \{1, 2, \dots, M\} \to \mathcal{X}^n$, where $\mathcal{X}^n$ is the set of sequences $\{x^n(1), x^n(2), \dots, x^n(M)\}$, each called a codeword and associated through the encoding operation with one of the indexes that forms the source output. The set of codewords is called a *codebook*. On the receiver side, the decoding operation is a function $g : \mathcal{Y}^n \to \{1, 2, \dots, M\}$ that estimates an index from the source for each received sequence $\vec{Y}$. For an (M, n) code, its rate is defined as:

$$R = \frac{\log_2(M)}{n} \qquad \text{[bits per transmission]}. \tag{1.47}$$

We now introduce the concept of information capacity of a channel. Let the random variables X and Y be the channel input and output, respectively. The *information capacity* of this channel is defined as the maximum of the mutual information between the input and output:

$$C = \max_{p_X(x)} I(X; Y), \tag{1.48}$$

where the maximum is taken over all possible PMFs $p_X(x)$ for the input and subject to a power constraint on the input. For a continuous channel, the capacity definition involves the PDF $f_X(x)$:

$$C = \max_{f_X(x), E(|X|^2) \le P} I(X; Y). \tag{1.49}$$

From the definition of mutual information, the channel capacity measures the maximum average amount of information that could be shared between the input and the output of the channel. Informally, the channel capacity indicates the maximum amount of information that could be retained from the random variable at the input of the channel after being transmitted through a channel. Linking these thoughts and the use of a code for transmission, Shannon established the relation between the channel capacity and the rate of the code used when seeking a communication process with an arbitrarily small probability of error. This relation is established in Shannon's *noisy channel coding theorem* [1], which states that it is possible to communicate over a DMC with an arbitrary small probability of error by using a code that has a rate R that is less than the channel capacity C. In this way, through his noisy channel coding theorem, Shannon provided a functional meaning to the channel capacity as the maximum transmission rate (measured in bits per channel use) at which it is possible to send information over the channel with an arbitrary low probability of error. Without delving into details (which the interested reader can find in [4]), it is important to note that the proof for the noisy channel coding theorem follows the definition of a

jointly typical set between the random variables X and Y at the channel input and output, respectively, and a resulting joint AEP. The joint AEP in this case is analogous to what we have seen earlier with the rate-distortion theory, but without the presence of a distortion metric, and, importantly, it is now defined in terms of the joint statistics between the random sequences at the input and output of the channel. Specifically, we can define a *jointly typical set* as the set satisfying:

$$\begin{aligned}\mathcal{A}_{\epsilon}^{(n)} = \big\{(x^n, y^n) \in \mathcal{X}^n \times \mathcal{Y}^n \ : \\ \left|-\frac{1}{n}\log_2 P_{\vec{X}}(\vec{x}) - H(X)\right| < \epsilon, \\ \left|-\frac{1}{n}\log_2 P_{\vec{Y}}(\vec{y}) - H(Y)\right| < \epsilon, \\ \left|-\frac{1}{n}\log_2 P_{\vec{X},\vec{Y}}(\vec{x},\vec{y}) - H(X,Y)\right| < \epsilon \Big\}.\end{aligned} \tag{1.50}$$

The noisy channel coding theorem uses the identification of a jointly typical set between the input and output of the channel. It does this to establish that for codewords of large length, a sequence at the output of the channel will be jointly typical with a codeword at the input of the channel with a probability very close to 1, and for the remaining codewords the probability of being jointly typical will be very small. This property allows for a decoding of the received sequence that undoes transmission errors resulting in an arbitrarily small probability of transmission error. The condition $C > R$ together with (1.48) or (1.49) simply determines the cardinality of the jointly typical set that can be allowed by the channel characteristics. For the purpose of this book, the important implication of these observations is that the condition of an arbitrarily small probability of error seen in the noisy channel coding theorem implicitly depends on the use of codes of infinite decoding complexity and operation with sequences of length n that approach infinity.

Having introduced the concept of capacity and the noisy channel coding theorem, we can now apply the definition of channel capacity and derive its expression for some of the channel models discussed earlier. For the AWGN channel, we can write (using differential entropy properties):

$$I(X;Y) = H(Y) - H(Y|X) = H(Y) - H(X+Z|X) = H(Y) - H(Z|X).$$

Since noise and channel input samples are independent of each other, we have that $I(X;Y) = H(Y) - H(Z)$. Also, for noise that is a Gaussian random variable with zero mean and variance σ_Z^2, we have $H(Z) = (1/2)\log_2(2\pi e\sigma_Z^2)$. Next, recall the property that the differential entropy is bounded as $H(Y) \leq (1/2)\log_2(2\pi e\sigma_Y^2)$. Since Y has zero mean, its variance is:

$$\sigma_Y^2 = E[Y^2] = E[(X+Z)^2] = E[X^2] + 2E[XZ] + E[Z^2] = \sigma_X^2 + \sigma_Z^2,$$

where we have used the independence between X and Z, and $E[Z] = 0$, to see that $E[XZ] = E[X]E[Z] = 0$. Using $P = \sigma_X^2$ to denote the signal power and $N = \sigma_Z^2$ to denote the noise power, we have $\sigma_Y^2 = P + N$ and $H(Y) \leq (1/2)\log_2(2\pi e(P+N))$. By collecting these results, we get:

$$I(X;Y) = H(Y) - H(Z) \leq \frac{1}{2}\log_2(2\pi e(P+N)) - \frac{1}{2}\log_2(2\pi eN) = \frac{1}{2}\log_2\left(1 + \frac{P}{N}\right),$$

where P/N is the signal-to-noise ratio (SNR) and, since $C = \max_{f_X(x)} I(X;Y)$, the channel capacity for the AWGN channel is:

$$C_{AWGN} = \frac{1}{2}\log_2\left(1 + \frac{P}{N}\right) \text{ [bits per channel use]}. \tag{1.51}$$

If we now consider the case of a channel that is AWGN and also behaves as a low-pass linear filter, the output of the channel is the convolution of the channel impulse response with a signal formed from the addition of noise to the transmitted signal. Since the channel has a low-pass characteristic, the output signal is band-limited (suppose to a frequency W) and can be sampled at a sampling rate of $2W$ samples per second. The sampling operation turns the channel model into the discrete-time AWGN channel for which we have just discussed the capacity. Therefore, the capacity for the band-limited AWGN channel can be written also as in (1.51), or as the popular equivalent expression:

$$C_W = \frac{1}{2}\log_2\left(1 + \frac{P}{N_0 W}\right) \text{ [bits per channel use]}, \tag{1.52}$$

where N_0 is the power spectral density of the AWGN. Also, since each channel use corresponds to the transmission of one sample and there are $2W$ samples per second, the channel capacity can also be written in units of bits per second as:

$$C_W = W\log_2\left(1 + \frac{P}{N_0 W}\right) \text{ [bits per second]}. \tag{1.53}$$

The maximization that defines the channel capacity in (1.48) and (1.49) is an operation of finding the supremum of all channel code rates that achieve an arbitrarily small probability of error during the process of communication. The channel code that achieves this supremum rate is ideal in the sense of involving infinite length and coding complexity. Communication systems realized with practical codes will exhibit a performance loss with respect to this ideal, manifested by the need for a larger SNR in order to achieve the same channel throughput. This effect is often modeled by approximating the throughput of a practical communication link with the same expression as in (1.53) but introducing the concept of an *SNR gap*, denoted Υ, that models the need for higher SNR. With this approach, the throughput T of a practical communication system is approximated as:

$$T = W\log_2\left(1 + \frac{\Upsilon P}{N_0 W}\right) \text{ [bits per second]}. \tag{1.54}$$

1.5 Layered Model for a Communication System

We will now pause on the abstractions of information theory and consider how practical systems are usually architected. In particular, we consider communicating a message through a network. Multiple interoperating components are needed for a network to effectively deliver a message from source to destination. Historically, network design components were organized into modular functional groups. This divide-and-conquer approach (simplifying a problem by breaking it apart into simpler subproblems) also exposed a hierarchy in the functions of the modular groups. Some groups provide a function needed by another group. For example, a component that deals with transmitted

bit errors depends on another component that converts a stream of bits into electrical signals that are sent into the communication channel. Collectively, the symbiosis of the functions provided by the different groups build up a network from the most basic functions of electrical signals to complex functions of managing traffic load at network nodes. The modular grouping of functions and the relations between them define a stack of layers, where each layer groups components with similar functions and dependencies. Layers lower in the stack provide services to higher layers. The layer stack forms a bottom-up building of network functionality, where basic lower-layer functions are needed by the more complex functions at higher layers.

In the 1970s, recognizing a widespread use of layered architectures for network protocol design, the International Organization for Standardization (ISO) led the development of the reference standard for the definition of the different layers and the features they provide to the overall network functioning. This standard is known as the *Open Systems Interconnection* (OSI) reference model for the layered architecture of networks [9]. The well-thought-out design behind the model has led to its robust longevity. The OSI model influence extends to the present time; all networks are based on the same layered architecture and retain most of the layers and their specification from the OSI model. Figure 1.9 shows the five-layer protocol stack used in the Internet. From the base of the stack to the top, the five layers are as follows:

- *Physical layer*: The *PHY layer* provides the services involved in the transmission of individual bits. It is at the physical layer that logical bits are converted into an electrical signal to be sent over a communication medium.
- *Data link layer*: The data link layer, or link layer, provides functions for establishing a connection between a transmitter and a receiver, and for the reliable communication of a structured block of bits (called a *frame*). The direct point-to-point communication connection between a transmitter and receiver is called a *link* and forms the minimum unit of interconnection in a network. In some protocol layer architectures, the data link layer is divided into further layers. Some of the functions provided by this layer are checking for bit errors at the receiver, retransmitting frames with unrecoverable errors, and arbitrating between multiple potential transmitters on access to the communication medium (called *Medium access control* [MAC]). With the services provided by the physical and link layers, it is possible to set up reliable communication between a transmitter and receiver.
- *Network layer*: This layer connects multiple transmitter–receiver links into a network with multiple nodes. The chaining of links establishes a *path* in the network. The network layer implements functions, such as routing, that enable information to

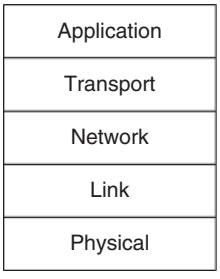

Figure 1.9 The Internet protocol stack consisting of five layers.

reach the intended destination by following a multi-node path through the network. The functions implemented at the network layer expand the topology of the network from a point-to-point (transmitter–receiver) connection to a network consisting of two end nodes (an originator of information and a receiver of the information) and multiple interconnected intermediate nodes. While devices that implement the bottom two layers of the protocol stack are only able to establish a direct connection with another device, those devices that also implement a network layer protocol become able to communicate information over a true network of paths consisting of many links.

- *Transport layer*: This layer enables an abstracted view of the network between the end nodes by providing functions for the reliable end-to-end transfer of information. At the transport layer level, the network in between the two end nodes is seen as a single entity that encompasses all the networks with their devices and components over which the two end nodes are interconnected. At the link layer, the concept of reliability concerns errors that occur during signal transmission over a medium. In contrast, end-to-end reliability at the transport layer addresses communication errors associated with considering the whole network between the end nodes as a single element. Such errors include packet losses at congested intermediate nodes or packets belonging to a stream that arrive in a different order from how they were sent.
- *Application layer*: At the top of the protocol stack, the application layer is associated with protocols for the different applications being run by end users. Applications intended to run over a network are designed with the implicit configuration of running in a distributed manner. For example, a distributed web browsing application has one part running on the end-user side, where web page queries are issued, and a complementary part that runs on the web server side, responding to the user queries. The protocols at the application layer provide the network interface for applications that run at the end nodes. Because of this, these applications are often considered part of the application layer. The source encoders and decoders from Shannon's model for a communication system are one example of applications.

Layers are often simply named with a number (e.g. PHY layer is Layer 1, data link layer is Layer 2). The modular approach combined with the high complexity of the functions implemented at each layer has led to a specialization of both layer design and implementation. In a typical networked device, one finds the two lower layers implemented within a chipset or circuit board (known as the network interface card [NIC]), while layers 3 and 4 would be a group of processes running in the kernel of the device's operating system, and the application layer would be a group of processes running within the user space of the device's operating system. Other than the particular case of control messaging, a message to be communicated originates at the application layer. From the application layer, the message is passed to the transport layer and then the network layer through a series of procedure and functions calls. The transfer of the processed message from the network layer (which resides in the device's operating system kernel) to the data link layer (which resides in a peripheral chipset) is done through function calls to the device driver (software in the operating system kernel that provides interfacing functions). The modular approach in the layered protocol architecture allows for the design or adaptation of certain layers to be better suited to specific communication environments.

Modern-day communication systems extend the basic layered architecture in multiple ways: capturing additional nuances of function, information ownership, service guarantees, and protocol termination points. The layered architecture also follows the software paradigms of separation of responsibilities, encapsulation, and hiding of details, multiple layers of abstraction, and separation of control procedures and data transmissions. Considerable thought is devoted to the design of the functions and responsibilities in each layer and the services they provide to and expect from other layers. Further, communication procedures and protocols between nodes and abstract entities are designed completely within a layer. With various network components owned, designed, and developed by altogether different entities, many of these architectures go through a process of standardization. Today's wireless communication standards are a triumph of technological innovation and large-scale multiparty collaboration in a common interest, introducing spectacular new technological features at a rapid pace, while leaving sufficient room for creativity and differentiation for all the parties involved. This is in no small part due to the idea of clean layered architectures. The layered architectures of today enable multiple access, unicast and multicast, single- and multihop communications, various quality-of-service (QoS) offerings on throughput, latency and reliability, security and privacy, and mixing of highly disparate physical deployments, e.g. frequency bands and heterogeneous nodes and form factors. As an example, Figure 1.10 shows the schematic layered architecture 5G-NR stack.

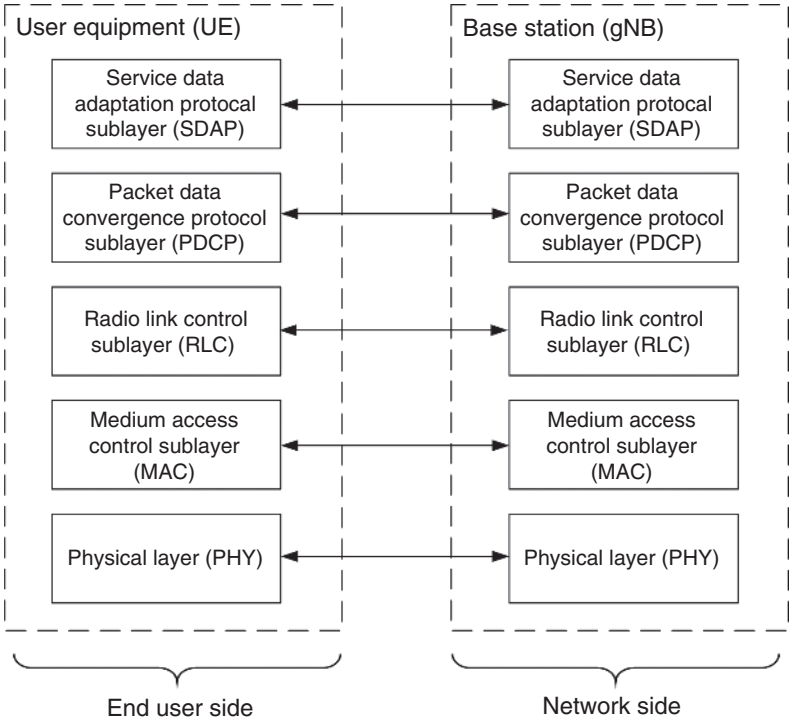

Figure 1.10 5G-NR User Plane stack as an example of layered architecture.

Yet, despite the layered architecture and the presumed independence of the layers, higher efficiency is often achieved when the layers are designed *jointly*, keeping in consideration the end-to-end service/application use case and the requirements of their QoS.

Joint source–channel coding paradigms understand the ground reality from the outset that joint designs may not only be better, but may even be unavoidable. Most designs of layered architectures today have a strong joint design element in their evolution. To appreciate this, it is useful to learn the notions of QoS and quality of experience (QoE) and their relationship with distortion metrics.

1.6 Distortion, Quality of Service, and Quality of Experience

Assessing communication system performance through a measurement of distortion or quality is of fundamental importance for the subject of this book. Measurements of distortion or quality can be divided along many different lines, according to whether they are objective or subjective, attempting to hew closely to properties of human perception or not, or involving comparison to a reference or not. Objective measurements of distortion are quantifiable calculations of the degradation of a signal or message in comparison with a reference. For example, an image compressed by JPEG is compared to the original image (the reference) as captured by the digital camera, and the distortion measure aims to quantify the degradation between them.

The term *quality* is often taken to be, roughly speaking, a kind of inverse of distortion. So an image that has low distortion typically has high quality. However, an important difference between distortion and quality is that, while distortion implies a comparison with a reference, the term quality is very often used without a reference. For example, one might consider an image to be high quality if it is sharp with good contrast, and say it is low quality if it is blurry and low contrast, irrespective of any comparison with a reference. Also, if one is given a reference image that is blurry and low contrast, and after some processing the output image is still blurry and low contrast, one would have to say that the output image is rather faithful to its reference, which means it is both low distortion (relative to its reference) and low quality (in an absolute sense).

A distortion metric, in comparing a processed signal with a reference signal, implicitly assumes that any deviation from the reference is a bad thing. In a case where the processing is, for example, image sharpening or contrast enhancement, the resulting processed image might be better looking than the original reference image. One might then have a situation where the processed output image is both high distortion (relative to the low-quality reference) and also high quality (in an absolute sense).

So these are examples where quality and distortion are not acting like inverses. A quality metric in which a processed signal is compared with a reference is more properly called a *fidelity metric*. So, a distortion metric measures in some sense how far away one is from a reference, and a fidelity metric measures how close one is to a reference, so these tend to move in opposite directions, or be roughly inverses. In this book, we will tend to use the term *quality metric* both to mean a *fidelity metric* where there is an explicit comparison with a reference, and also use it in the more general sense.

1.6.1 Objective Measurements of Distortion or Quality

For objective measurements of distortion, the calculations of signal or message degradation with respect to a reference are based on the mathematical framework of measurements. A measurement is a function from the product space formed from the supports of the two quantities to be compared, into a positive real number. Using the set $\mathcal{X}$ to denote the support of both a message and its reconstructed estimate, the distortion measure d is a function:

$$d : \mathcal{X} \times \mathcal{X} \rightarrow [0, \infty). \tag{1.55}$$

Let $x, \hat{x}, y \in \mathcal{X}$. To be a proper measure (in the mathematical sense), a distortion measure needs to satisfy the following three axioms:

- $d(x, \hat{x}) = 0$ if and only if $x = \hat{x}$.
- $d(x, \hat{x}) = d(\hat{x}, x)$.
- $d(x, \hat{x}) \leq d(x, y) + d(y, \hat{x})$.

There exist numerous measurements of distortion or quality, some tailored to specific cases. We already introduced the MSE between two sequences in (1.22). When calculating the MSE between two random variables (e.g. a source symbol X and its received estimate $\hat{X}$) the MSE is:

$$MSE = E[(X - \hat{X})^2], \tag{1.56}$$

where the expectation is over the joint probability distribution of X and $\hat{X}$. For calculating the MSE between two sequences of random variables, $\vec{X}$ and $\hat{\vec{X}}$, the expression is:

$$MSE = E[(\vec{X} - \hat{\vec{X}})^T(\vec{X} - \hat{\vec{X}})] = \sum_{k=1}^{N} E[(X_k - \hat{X}_k)^2]. \tag{1.57}$$

The MSE is the most commonly used distortion measure. Another common measure is the mean absolute error between two sequences of random variables, $\vec{X}$ and $\hat{\vec{X}}$:

$$MAE = E[|\vec{X} - \hat{\vec{X}}|]. \tag{1.58}$$

These measures of distortion are not simply converted into measures of quality (or fidelity) by calculating an inverse. Earlier, we saw that the SNR provides a sense of channel quality by relating the transmitted signal power to the added noise power; we can extend this idea to measure the quality of a signal that has undergone some distortion. This requires that the distortion could be modeled as modifying the signal through some additive process. Let us first consider the case of a random variable X modeling a source symbol. After undergoing some distortion, it becomes the random variable $\hat{X}$. Assuming for simplicity that all random variables are zero mean, the SNR is calculated as:

$$SNR = \frac{\sigma_X^2}{E[(X - \hat{X})^2]}, \tag{1.59}$$

where σ_X^2 is the variance of X and the denominator is the MSE between X and $\hat{X}$. In other circumstances, we may be interested in calculating the SNR for a signal that is presented as a sequence of N samples $\{x_1, x_2, \ldots, x_N\}$, which has been distorted into the sequence

of samples $\{y_1, y_2, \ldots, y_N\}$. In this case, the signal power is $\sum_{k=1}^{N} x_k^2$, and the SNR is calculated as:

$$SNR = \frac{\sum_{k=1}^{N} x_k^2}{\sum_{k=1}^{N} (x_k - y_k)^2}. \tag{1.60}$$

For signals that are images or videos, instead of calculating the SNR, it is more common to calculate the peak signal-to-noise ratio (PSNR). The PSNR measures the square of the maximum value that the signal could take, relative to the power associated with the distortion (measured as the MSE). For the common case where the image or video frame is an eight bit per pixel grayscale image, the maximum value that a pixel can have is 255. The PSNR is typically calculated in decibels as:

$$PSNR = 10\log_{10}\left[\frac{255^2}{\frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}(x_{i,j} - y_{i,j})^2}\right], \tag{1.61}$$

where $x_{i,j}$ are the pixels of the original image or video frame and $y_{i,j}$ the pixels of the distorted image or video frame. The PSNR is also used to measure video quality either by examining the PSNR across time for each frame in a video sequence or by calculating a PSNR for the whole video. As having any single frame to be nearly (or completely) free of distortion causes the PSNR for that frame to be huge (or infinite), averaging PSNR values over video frames can be problematic. Instead, calculating a single PSNR value for an entire video usually involves averaging the MSE (the denominator) across all frames and then converting that to PSNR by taking the ratio of 255^2 to that average MSE.

1.6.2 Subjective and Perceptually Based Measurements of Distortion or Quality

Since many messages are intended for people, and people experience a signal or message through the senses (e.g. sight for images or video, or hearing for audio or speech), the measure of distortion or quality should often be related to human perception. One approach is to retain simple objective measurements and rely on individual experience to develop a correspondence between the objective measure and human perception. For example, a person with experience examining images and measuring their PSNR may know that images with a PSNR of 40 dB have very good perceived quality, while ones with a PSNR of 32 dB are good (better than mediocre). However, for some images (such as ones with smooth and uniform texture), 32 dB corresponds to very good quality, while for other images (possibly those with more texture and sharp or contrasting edges) 32 dB might correspond to mediocre perceived quality. So, while simple objective measurements such as MSE and PSNR may provide some sense of the distortion or quality perceived by a generic human subject, they are not an accurate way to assess quality or distortion as perceived by a person.

Because of this, a great deal of research and development has involved both *subjective measurements* of distortion or quality (in which people provide subjective ratings, based on their perceptions) and *perceptually based objective measurements* (in which computable objective metrics are refined, aiming for good correspondence with human subjective ratings). Perceptually based measures of quality or distortion tend to incorporate the sensitivity of human senses to different stimuli. Certain levels of distortion may be

imperceptible to human senses, and humans do not have uniform sensitivity. For example, people can tolerate (or not even perceive) distortion of a sound with two nearby frequency components where one of them has a distinctly larger power than the other. This property is exploited by the MP3 audio codec to achieve compression with distortion that is very hard to perceive. For information communication, factors beyond signal representation also impact people's end-to-end perception of quality. For example, end-to-end delays of more than about 150 ms in interactive conversational communication result in mediocre or poor perceived end-to-end quality. As a result, various means to measure the subjective quality or distortion of a communication process as perceived by human end users were developed over time.

Subjective measurements of distortion have been used for quite some time in communication systems. One of the first uses was in landline (wireline) telephone systems to evaluate voice quality over different systems. The established procedure in the telephone industry was to ask a group of people to listen to voice recordings processed by the equipment under test (for example, a new speech codec at the time, such as ITU-T G.711) and rate the speech quality. The trained listeners used a scale of five possible quality scores to provide their individual opinions: Excellent (5), Good (4), Fair (3), Poor (2), and Bad (1). The scores were averaged across listeners, producing the *mean opinion score* (MOS) as the final measurement result. Over time, this evaluation procedure was standardized by the International Telecommunications Union (ITU) as ITU-T recommendation P.800 "Methods for subjective determination of transmission quality."

With the acceleration of technology, the need arose for a more automated way to measure or estimate MOS without the laborious process of convening and training listeners. Algorithms were developed to estimate the MOS of a signal that had been degraded by some processing operations compared to the signal reference in an unprocessed form. Figure 1.11 illustrates the Perceptual Evaluation of Speech Quality *(PESQ) model*, an automated MOS estimation algorithm. This model was defined in ITU-T recommendation P.862 "Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs." Its block diagram in Figure 1.11 is typical of this type of algorithm; it has stages of signal conditioning, filtering, and transforms that modify the signals based on the sensitivity of human hearing to

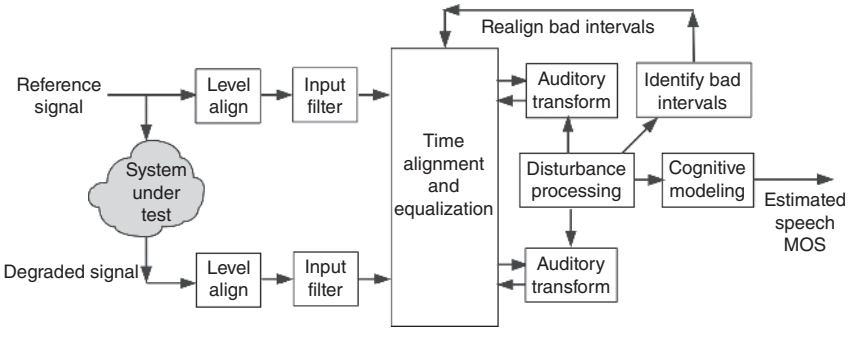

Figure 1.11 Block diagram for speech MOS estimation using the ITU-T recommendation P.862 Perceptual Evaluation of Speech Quality (PESQ).

different stimuli. The aim is to boost, or put more weight, on sounds for which human hearing is more sensitive and remove sounds that humans cannot hear. Interestingly, while the MOS scale is between 1 and 5, the PESQ algorithm generates results between 1 and 4.5 so that PESQ results better correlate with the normal range of MOS values seen in tests.

The PESQ speech MOS estimation algorithm is known as an *intrusive* or *full-reference* (FR) method of measuring signal quality. With intrusive or FR methods, the estimation of quality requires access to the original signal. With the evolution of telephony into digital technology and, specially, packet voice in the form of voice-over-IP (VoIP) services, network operators needed a nonintrusive method to estimate MOS speech quality to manage the network based on the inferred quality. *Nonintrusive* or *No-Reference* (NR) methods to estimate quality do not require access to the original, unprocessed, signal, allowing instead for subjective quality estimation to be done using parameters of the system (e.g. codec used) and objective measurements of traffic and network performance. These objective performance measurements, known as *QoS* metrics, include transmission throughput, packet (or bit) error rate, and transmission delays. One nonintrusive method is the E-model (defined in the ITU-T recommendation G.107), which estimates speech quality using the *R-factor*, which is calculated using the formula:

$$R = R_0 - I_s - I_d - I_e + A, \tag{1.62}$$

where R_0 is related to the SNR, I_s represents the combined effects of different impairments that may occur during voice communication (including quantization distortion), I_d represents the combined effects of communication delay (e.g. echo or too long end-to-end delay), I_e represents the performance of specific voice codecs against packet loss errors (e.g. error concealment), and A describes the end-user tolerance to impairments due to the convenience of the technology in use. While the R-factor is not in the same range as MOS, it is possible to convert from one to the other through the relation $MOS = 1 + 0.035R + 7 \times 10^{-6} R(R-60)(100-R)$ for $0 < R < 100$, $MOS = 1$ for $R \leq 0$, and $MOS = 4.5$ for $R \geq 100$, as illustrated in Fig. 1.12.

The term *QoS* refers to an objective performance measurement of network parameters (e.g. packet loss rate, bit rate, throughput, delay, jitter, outage rate) that indicates the

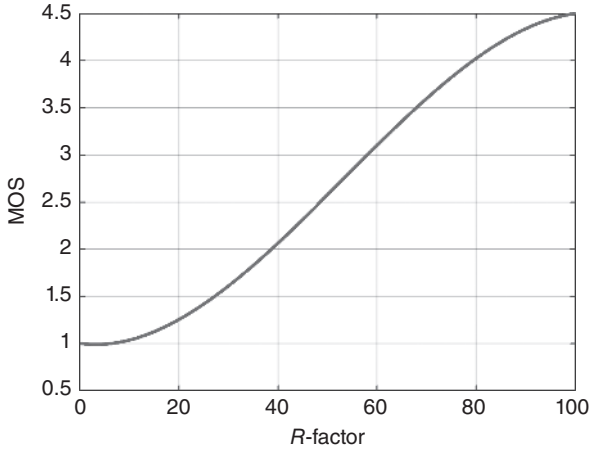

Figure 1.12 Conversion from *R*-factor into MOS.

quality of the networking service provided by the network. Driven by an increased focus on end-user network design and management, subjective and perceptually based quality measurements grew into the concept of *QoE*, which are measurements (derived from people or derived from computable metrics seeking to emulate human responses) of the quality of a network service (or traffic type) as subjectively experienced by end users. The QoE depends on the QoS the network provides. A direct method to measure QoE in voice traffic is the MOS. As with the development of VoIP leading to MOS estimation algorithms, the growth of non-voice services over networks drove the development of MOS estimation algorithms for other types of network traffic/service, not only for those related to human perception (e.g. video streaming) but also for those not experienced through senses (e.g. a file download). For MOS estimation in the case of a file download service, one formula that has been used is as follows:

$$MOS = a\log_{10}(bR(1-\epsilon)), \tag{1.63}$$

where R is the bit rate and ϵ is the packet error rate. The constants a and b are parameters to adjust the regression estimate in the curve to the concept behind an MOS result. They are calculated using the conditions that MOS should equal 4.5 when R equals the rate that the end user has subscribed to and the packet error rate is zero, and that MOS equals 1 when the rate is at its minimum assumed value. For video streaming services, many models have been developed. One simple approach uses the following mapping between PSNR and MOS [10]:

$$MOS = \begin{cases} 5, & \text{for } PSNR > 37, \\ 4, & \text{for } PSNR \text{ between 31 and 37}, \\ 3, & \text{for } PSNR \text{ between 25 and 31}, \\ 2, & \text{for } PSNR \text{ between 20 and 25}, \\ 1, & \text{for } PSNR < 20. \end{cases} \tag{1.64}$$

Other approaches improve the MOS estimate by using other functions, e.g. logarithmic. In [11], the approach to model MOS of video streaming first characterizes the relation between PSNR (measured in decibels) and source encoding rate R as:

$$PSNR_{dB} = a + b\left(\sqrt{\frac{R}{c}} - \sqrt{\frac{c}{R}}\right), \tag{1.65}$$

where a, b, and c are parameters that fit the model to a video stream or sequence. Next, the PSNR is mapped to MOS through the following formula:

$$MOS = \begin{cases} 1 & \text{for } PSNR \leq PSNR_1, \\ d\log(PSNR) + e & \text{for } PSNR_1 < PSNR < PSNR_{4.5}, \\ 4.5 & \text{for } PSNR \geq PSNR_{4.5}, \end{cases} \tag{1.66}$$

where the model fitting parameters d and e are defined as:

$$d = \frac{3.5}{\log(PSNR_{4.5}) - \log(PSNR_1)},$$
$$e = \frac{\log(PSNR_{4.5}) - 4.5\log(PSNR_1)}{\log(PSNR_{4.5}) - \log(PSNR_1)}.$$

Combining (1.65) into (1.66) results in a model that relates source encoding rate and MOS. An alternative model between PSNR and source encoding rate of the form $\text{PSNR} = a\log(R) + b$ can be found in [12]. Using this logarithmic relation between PSNR and source encoding rate and the logistic regression model between MOS and the logarithm of the source encoding rate presented in [13], the relation between video MOS and PSNR can be expressed as [14]:

$$\text{MOS} = \frac{a}{1 + e^{b(\text{PSNR}-c)}},$$

where a, b, and c are model fitting parameters.

There exist many more proposed alternative measures to measure quality and distortion, with different approaches for capturing perceptual effects. The simple objective measurements (such as MSE, PSNR, and mean absolute error (MAE)) continue to be used because of mathematical tractability or simplicity. But the subjective measurements and complex perceptually based computable metrics perform better at describing how human end users will subjectively experience the quality of a communication process. In connection with Section 1.5, it is worth noting that the inherent end-to-end nature (or "mouth-to-ear" as it is sometimes called in telephone technology) of QoE measures lend their use naturally to cross-layer design approaches, of which joint source–channel coding (JSCC) represents one important subclass.

1.7 Shannon's Separation Principle and Joint Source–Channel Coding

The alert reader may have realized that earlier parts of this chapter ordered topics aligned with the general communication system block diagram in Figure 1.1. We started with the foundational concepts of entropy and information in Section 1.2, followed by a discussion in Section 1.3 of information content in a source, how to represent this information, and the consequences of removing some of it. Then we focused in Section 1.4 on channel models and how much information can be communicated through them. All these elements now come together in the study of a complete system for communicating a source.

We start with lossless communication. We have seen that to be lossless, the source encoding of a source of entropy H bits per source sample needs to use a code with a rate R_s bits per source sample that satisfies $R_s > H$. Consider the communication of this source over a channel where n channel uses are permitted per source sample. We have also seen that to achieve transmission with an arbitrarily low probability of error over a channel with capacity C bits per channel use, the channel code used needs to satisfy the condition on the information rate R_c (bits per channel use) as $R_c < C$. The two conditions, one for lossless representation of source information and one for a mapping matched to the capability of a channel to reliably carry information, can be combined into the condition $H < R_s = nR_c < nC$. If, without loss of generality, $n = 1$, then the above condition becomes $H < R < C$, where $R_s = R_c \stackrel{def}{=} R$. This condition leads to two questions. Is the condition $H < C$ necessary and sufficient for the transmission of a given source over a channel with arbitrarily small

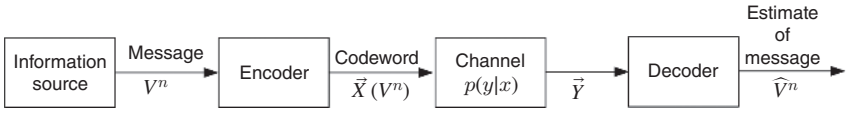

Figure 1.13 Distilled view of the message communication process with joint source and channel coding.

probability of error to be realizable? And how does one approach the design of a code with rate R, such that $H < R < C$, which succeeds in losslessly representing the source and matching the channel capacity so that the source transmission has an arbitrarily small probability of error? These questions are addressed by Shannon's *source–channel coding theorem* and its implications.

Consider the setup in Figure 1.13. A discrete source associated with a finite alphabet $\mathcal{V}$ generates a random sequence of symbols V^n. The sequence is encoded into a codeword $\vec{X}(V^n)$ that is transmitted through a DMC of capacity C and characterized with the probability distribution $P(\vec{Y}|\vec{X})$, where $\vec{Y}$ is the received codeword. This received codeword is decoded to obtain an estimate $\hat{V}^n$ of the original message. Shannon's source–channel coding theorem states that when the sequence V^n satisfies the AEP and $H(\mathcal{V}) < C$, there exists a source–channel code for which it is possible to send the source message through the channel with an arbitrarily small probability of error (probability that $\hat{V}^n \neq V^n$). Conversely, when $H(\mathcal{V}) > C$, there is no code that would make possible the transmission of the source message with an arbitrarily small probability of error.

Shannon's source–channel coding theorem states that the condition to be able to transmit a stationary ergodic source with an arbitrarily small probability of error is that its entropy rate needs to be less than the channel capacity. The proof of achievability for the theorem has important implications for the subject of this book. In the proof, we first consider that since the source sequence V^n satisfies the AEP, there will be a typical set $A_\epsilon^{(n)}$ of probability $P[A_\epsilon^{(n)}] > 1 - \epsilon$ formed by $2^{n(H(\mathcal{V})+\epsilon)}$ distinct length-n sequences. Next, it is assumed that only the sequences in the typical set are encoded. Sequences not in the typical set are discarded. This will introduce an error with probability at most ϵ (the probability of the set that is the complement of the typical set). To encode the sequences in the typical set, they are listed and indexed. The lossless representation of the index can be achieved using $n(H(\mathcal{V}) + \epsilon)$ bits because there are at most $2^{n(H(\mathcal{V})+\epsilon)}$ sequences to be indexed. We can now draw from the noisy channel coding, where it was assumed that the source alphabet was a set of indexes and assert that we can transmit the source message, an index, with a probability of error no larger than ϵ if the rate of the code used, R bits per channel use, is less than the channel capacity, or, by including the representation of a transmitted index, if $H(\mathcal{V}) + \epsilon = R < C$. As the length n of the sequence grows to infinity, the probability of error introduced when encoding the source (at most ϵ) and the probability of transmission error (also at most ϵ) go to zero. As a consequence, messages from the source can be transmitted with an arbitrarily low probability of error when $H(\mathcal{V}) < C$.

The proof of achievability for the source–channel coding theorem gravitates around the existence of a typical set with properties as described in the AEP. The proof applies the AEP in two instances. The first is the lossless encoding of the source using at most $n(H(\mathcal{V}) + \epsilon)$ bits. The second instance uses the joint AEP to ensure that if the code rate R is less than

the channel capacity C, it will be possible to achieve an arbitrarily small probability of error during transmission. The two instances of applying the AEP are separate because they do not depend on any common information quantity; the first instance depends on the statistical characteristics of the source, and the second depends on the channel statistics (as given in terms of the input and output of the channel). This application of the AEP in two separate instances implicitly introduces a design for the communication system where the message undergoes two separate encoding operations before transmission. The first encoding, which we have called *source encoding*, performs a representation of the message that is as compact as possible while being lossless. The second encoding, which we called *channel encoding*, represents the source-encoded message in such a way that the transmitted codeword will have a probability of almost 1 of being jointly typical with the received sequence, and, thus, there will be a very high probability of decoding the received sequence with no errors. While there is nothing new in what we are describing here (these concepts were described earlier in the chapter, see Figure 1.1), the important new implication that follows from the source–channel coding theorem (and its proof) is that *there is no performance penalty to be paid* when doing the encoding in two separate stages: a first stage optimized for source encoding and a second stage tuned to undoing transmission errors during decoding. This principle –there is no performance penalty in designing source and channel coding separately – is known as the *source–channel separation theorem*.

Separating the design of source and channel coding subsystems has been prevalent in the development of communication systems. This prevalence has been motivated more by considerations of system modularity, flexibility, and design simplicity, rather than by an interest in taking advantage of the lack of a performance penalty promised by the theorem, because it is generally acknowledged that the separate design of source and channel coding does entail a performance penalty for practical communication systems. The lack of a performance penalty stated in the source–channel separation theorem is achieved at the hidden cost of using ideal codes of infinite complexity and delay because they operate on sequences of infinite length.

Furthermore, codes operating on very large sequences may not turn out to have such ideal high performance under practical circumstances. A practical source may change its statistical characteristics over time frames shorter than those associated with a source encoder input sequence, turning an initially optimal source coder design into a suboptimal one. For example, a speech source could change its statistical characteristics over a few tens of milliseconds, which might correspond to no more than two thousand bits. Similarly, a wireless channel may change its statistical characteristics on time frames shorter than those associated with a channel encoder input sequence, turning an initially optimal channel coder design into a suboptimal one. Depending on the transmitter–receiver velocities, the coherence times for a wireless channel may be as small as a few milliseconds, and typical values may be close to a hundred milliseconds.

Also, it might not be the best design approach to strive for a source encoder capable of extracting all redundant information in the source, achieving the most compact representation possible. The resultant source-encoded data would be extremely fragile to transmission errors because every part is indispensable for the reconstruction of the source at the receiver. Residual source redundancy that remains in the source-encoded data stream could be useful in dealing with transmission errors at the receiver. Moreover, separating the design of

source and channel coding subsystems entails another non-apparent performance penalty in terms of the rate that an optimal code's probability of error decreases with its block length – also known as the *error-exponent*. It has been shown in [15] that even for discrete memoryless sources and channels, the joint source–channel codes can achieve as much as twice the rate of error-exponent decrease as a separate source and channel coding design, even in the regime of an infinite input sequence length. We will explore all of these concepts in great detail later in this book.

As seen in the foregoing discussion, Shannon's source–channel coding theorem follows from combining the implications of the AEP in the lossless source coding theorem and the joint AEP in the channel capacity theorem. We can also combine in a similar way the joint AEP in the rate-distortion theorem with the joint AEP in the channel capacity theorem and obtain a *lossy source–channel coding theorem*. Assume that a discrete source associated with a finite alphabet $\mathcal{V}$ generates a random sequence of symbols V^n. The sequence is encoded into a codeword $\vec{X}(V^n)$ that is transmitted through a DMC of capacity C and characterized with the probability distribution $P(\vec{Y}|\vec{X})$, where $\vec{Y}$ is the received codeword. This is decoded to obtain an estimate $\hat{V}^n$ of the original message. In this system, it is now assumed that the encoding and decoding operations perform following a rate-distortion function $R(D)$. Let

$$D = E[d(V^n, \hat{V}^n)] = \frac{1}{n}\sum_{i=1}^{n} E[d(V_i, \hat{V}_i)] \tag{1.67}$$

be the average *end-to-end distortion* resulting from the combined source and channel coding as in Figure 1.13. The lossy source–channel coding theorem states that the distortion D is achievable if and only if $R(D) < C$. Of course, since higher rates lead to smaller distortions, the theorem could be regarded as establishing the smallest source distortion that could be achieved for a channel of a known capacity. Together, the lossless and lossy source–channel coding theorems say what source distortion can be achieved over a channel. If the channel capacity is larger than the source entropy, source distortion will be zero. If the channel capacity is less than the source entropy, lossy compression of the source message will be needed leading to some distortion.

In (1.67), we introduced the concept of end-to-end distortion that measures the difference between the sequence of symbols generated by the source and the estimate of those symbols after the message has been decoded by the recipient. In measuring the message difference between the two ends of the communication system, the end-to-end distortion incorporates the effects of the transmission through the channel as well as all the encoding and decoding stages in the processing chain of the communication system. Consequently, the end-to-end distortion is a metric that encapsulates the performance trade-offs between different stages in the communication of a message, including the trade-off within the source–channel coding theorem. Thus, the distortion measure is at the center of most joint source–channel design techniques and will be of frequent use in this book.

The lossy source–channel coding theorem also leads to the definition of the *optimal performance theoretically attainable* (OPTA) curve. The OPTA curve represents the smallest distortion that can be achieved over a channel as a function of the received signal power (or, equivalently, the channel SNR). It is defined as the distortion value D that satisfies

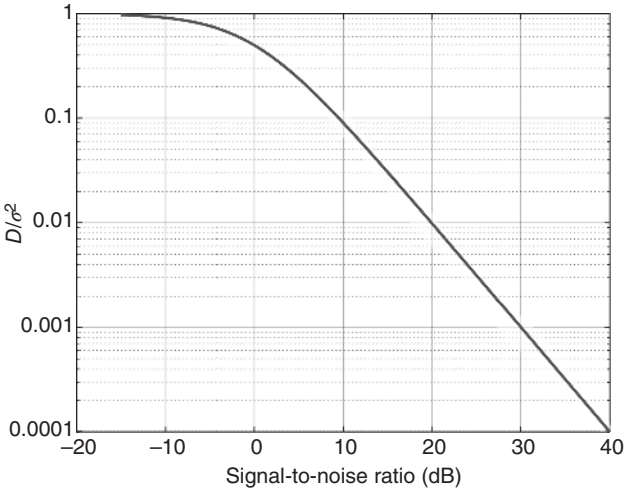

Figure 1.14 The Optimal Performance Theoretically Attainable (OPTA) curve for the Gaussian source over AWGN.

$R(D) = C(P)$. For Gaussian sources over AWGN channels, combining (1.30) and (1.52) through the definition of the OPTA curve (for nonzero rate and channel capacity) yields:

$$\frac{1}{2}\log_2\left(\frac{\sigma^2}{D}\right) = \frac{1}{2}\log_2\left(1 + \frac{P}{N_0 W}\right).$$

Thus, after denoting the channel SNR as $\gamma = P/(N_0 W)$, the OPTA curve is the function:

$$\frac{D}{\sigma^2} = \frac{1}{1+\gamma}. \tag{1.68}$$

where the distortion is expressed in the customary normalized form D/σ^2. Figure 1.14 illustrates the OPTA curve as expressed in (1.68). More discussion on the OPTA curve can be found in [16].

1.8 Major Classes of Joint Source–Channel Coding Techniques

In the previous section, we discussed a number of circumstances under which Shannon's source–channel coding theorem does not hold. These conditions are common in practical systems and include source and channel dynamic change, and coding complexity and delay constraints. When Shannon's source–channel coding theorem does not hold, there is a performance penalty to be paid when designing the source codec separately from the channel codec, and it is advantageous to follow a *JSCC* design approach. There are many approaches to joint design. We can identify the following classes of JSCC:

- *Concatenated joint source–channel coding*: This approach aims at striking a balance between retaining the advantages of separate source and channel codes design, in terms of modularity, flexibility, and design specialization and at the same time attaining the

performance improvement associated with joint source–channel coding design. In this approach, the source codec and channel codec are two separate processing blocks within the communication system, and joint design is realized by jointly determining their configuration parameters. The joint design entails distributing the channel capacity (or achievable throughput) between the source and channel encoders, aiming to minimize end-to-end distortion. Since end-to-end distortion measures the difference between the source message and its reconstruction after undergoing source and channel encoding and decoding, as well as transmission through a channel, it couples the source and channel coding operation by incorporating in a single metric the effects associated with source coding and with transmission errors. This coupling effectively establishes a trade-off between the source and channel coding rates, which is balanced through this JSCC design approach. As a result, the essence of the design in this approach consists in finding the amount of source compression that reduces the bit rate from the source (by eliminating redundancy and possibly introducing some distortion) enough to make room for the controlled addition of structured redundant bits that will help combat transmission errors so that end-to-end distortion is minimized while the resulting overall source and channel coding rate equals the channel achievable throughput.

- *Unequal error protection source–channel coding*: In many scenarios, the output from the source encoder is frequently not uniform but exhibits sections of different impact on the reconstructed source distortion. Some sections of source encoder output may result in large reconstruction distortion after being affected by transmission errors, while under the same circumstances other sections may yield small reconstruction distortion. Unequal error protection source–channel coding incorporates the nonuniform characteristics of the source-encoded output into the design by applying channel codes with stronger protection against transmission errors to those sections of the source-encoded output that would result in large distortion if received with transmission errors, and applying weaker channel codes to sections that, if received with errors, have a small impact on the reconstructed source distortion. Typical examples of more sensitive parts of compressed sources are control or synchronization information or portions where errors may lead to error propagation. This type of paradigm is also useful when the channel is changing or not fully known at the instant of source/channel encoding.
- *Constrained joint source–channel coding*: This class encompasses JSCC techniques where the source codec design incorporates constraints that account for the transmission of the source-encoded codeword over a noisy channel. One JSCC technique within this class is channel-optimized quantizer design, where the distortion introduced by transmission errors is included within the performance metrics used to design a quantizer. In another approach to constrained JSCC, the source encoder is designed so that its output presents an inherent resiliency against channel errors. Constrained JSCC techniques are not limited to the source encoder design; they also include receiver-side techniques that exploit the fact that under conditions when the source–channel separation theorem does not hold, the source and channel decoders do not behave as completely independent entities and can benefit from information exchange.
- *Analog and hybrid digital–analog JSCC*: In this case, the communication system does not use digital symbols and, instead, applies the analog model of the channel across the source–channel design.

References

1 Shannon, C.E. (1948). A mathematical theory of communication. *Bell System Technical Journal* 27: 379–423.
2 Shannon, C.E. (1949). Communication in the presence of noise. *Proceedings of the IRE* 37 (1): 10–21.
3 Hartley, R.V.L. (1928). Transmission of information. *Bell System Technical Journal* 7 (3): 535–563.
4 Cover, T. and Thomas, J. (1991). *Elements of Information Theory*. Wiley.
5 Oppenheim, A.V. and Schafer, R.W. (2013). *Discrete-time Signal Processing*. Pearson.
6 Manolakis, D.G. and Ingle, V.K. (2011). *Applied Digital Signal Processing: Theory and Practice*. Cambridge University Press.
7 Vetterli, M., Kovačević, J., and Goyal, V.K. (2014). *Foundations of Signal Processing*. Cambridge University Press.
8 Goldsmith, A. (2005). *Wireless Communications*. Cambridge University Press.
9 Zimmermann, H. (1980). OSI reference model - the ISO model of architecture for open systems interconnection. *IEEE Transactions on Communications* 28 (4): 425–432.
10 Gross, J., Klaue, J., Karl, H., and Wolisz, A. (2004). Cross-layer optimization of OFDM transmission systems for MPEG-4 video streaming. *Computer Communications* 27 (11): 1044–1055.
11 Saul, A. and Auer, G. (2009). Multiuser resource allocation maximizing the perceived quality. *EURASIP Journal on Wireless Communications and Networking* 2009: 1–15.
12 Kwasinski, A. and Kwasinski, A. (2015). Integrating cross-layer LTE resources and energy management for increased powering of base stations from renewable energy. *2015 13th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, pp. 498–505.
13 Hanhart, P. and Ebrahimi, T. (2014). Calculation of average coding efficiency based on subjective quality scores. *Journal of Visual communication and image representation* 25 (3): 555–564.
14 Mohammadi, F.S. and Kwasinski, A. (2018). QoE-driven integrated heterogeneous traffic resource allocation based on cooperative learning for 5G cognitive radio networks. *2018 IEEE 5G World Forum (5GWF)*, pp. 244–249.
15 Zhong, Y., Alajaji, F., and Campbell, L.L. (2006). On the joint source-channel coding error exponent for discrete memoryless systems. *IEEE Transactions on Information Theory* 52 (4): 1450–1468.
16 Berger, T. (1971). *Rate Distortion Theory*. Prentice-Hall.