1

Classical Information Theory and Classical Error Correction

Markus Grassl

Max-Planck-Institut für die Physik des Lichts, Staudstraße 2, 91058 Erlangen, Germany

1.1 Introduction

Information theory establishes a framework for any kind of communication and information processing. It allows us to derive bounds on the complexity or cost of tasks such as storing information using the minimal amount of space or sending data over a noisy channel. It provides means to quantify information. So one may ask, "How much does someone know about what somebody else knows?" – a question which is important in cryptographic context. Before studying the new aspects that quantum mechanics adds to information theory in later chapters, we will have a brief look at the basics of classical information theory in the next section.

While information theory provides an answer to the question how fast one can send information over a noisy channel, it usually does not give a constructive solution to this task. This is a problem error correction deals with. In Section 1.3, we give a short introduction to linear blocks codes, laying ground for the discussion of error-correcting codes for quantum systems in Chapter 7.

1.2 Basics of Classical Information Theory

1.2.1 Abstract Communication System

The foundations of information theory have been laid by Claude Shannon in his landmark paper "A Mathematical Theory of Communication" [1]. In that paper, Shannon introduces the basic mathematical concepts for communication systems and proves two fundamental *coding theorems*. Here, we mainly follow his approach.

The first important observation of Shannon is that although the process of communication is intended to transfer a message with some meaning, the design of a communication system can abstract from any meaning:

> The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another

Quantum Information: From Foundations to Quantum Technology Applications.
Edited by Dagmar Bruß and Gerd Leuchs.

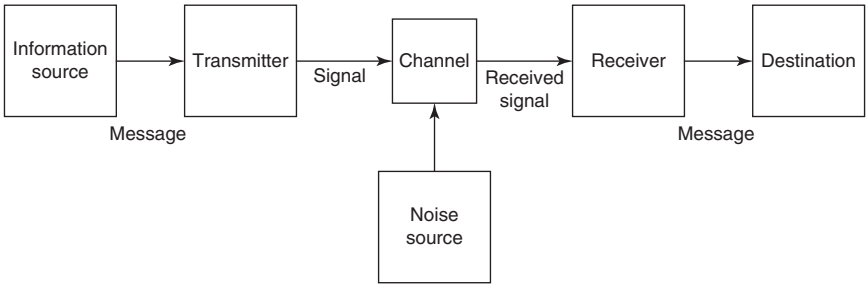

Figure 1.1 Schematic diagram of a general communication system.

> point. Frequently the messages have *meaning*; that is they refer to or are correlated according to some system with certain physical or conceptual entities. These semantic aspects of communication are irrelevant to the engineering problem. The significant aspect is that the actual message is *one selected from a set* of possible messages. The system must be designed to operate for each possible selection, not just the one which will actually be chosen since this is unknown at the time of design.

Additionally, we can, to some extent, abstract from the physical channel that is used to transmit the message. For this, we introduce a *transmitter* and a *receiver* that convert the messages into some physical signal and vice versa. The general layout of such a communication system is illustrated in Figure 1.1. Given a channel and an information source, the basic problem is to transmit the messages produced by the information source through the channel as efficient and as reliable as possible. Efficient means that we can send as much information as possible per use of the channel, and reliable means that, despite the disturbance due to the noise added by the channel, the original message is (with high probability) reproduced by the receiver. Shannon has shown that one can treat the two problems separately. First, we will consider a *noiseless channel*, which transmits every input perfectly, and then we will deal with *noisy channels*. For simplicity, we will consider only *discrete channels* here, that is, both the input and the output of the channel, as well as those of the transmitter and receiver, are symbols of a finite discrete set.

1.2.2 The Discrete Noiseless Channel

For a channel that transmits its inputs perfectly, the design of a communication system aims to maximize its efficiency, that is, the amount of information that can be sent through the channel per time. Usually, it is assumed that for each channel input, the transmission takes the same amount of time. Then, we want to maximize the throughput per channel use. Otherwise, we first have to consider how many symbols we can send through the channel per time. Following [1], we define

Definition 1.1 (capacity of a discrete noiseless channel) The capacity C of a discrete channel is given by

$$C = \lim_{T\to\infty} \frac{\log_2 N(T)}{T},$$

where $N(T)$ is the number of allowed signals of duration T.

If we use the symbols $x_1, \dots, x_n$ with durations $t_1, \dots, t_n$, then we get the recursive equation

$$N(t) = N(t - t_1) + N(t - t_2) + \cdots + N(t - t_n),$$

as we can partition the sequences of duration t by, say, the last symbol. For large t, $N(t)$ tends to X_0^t where X_0 is the largest real solution of the characteristic equation

$$X^{-t_1} + X^{-t_2} + \cdots + X^{-t_n} = 1. \tag{1.1}$$

Summarizing, we get

Lemma 1.1 The capacity of a discrete noiseless channel with symbols $x_1, \dots, x_n$ with durations $t_1, \dots, t_n$ is

$$C = \log_2 X_0,$$

where X_0 is the largest real solution of (1.1).

In order to maximize the efficiency of the communication system, we additionally need a measure for the amount of information that is produced by the source. Recall that we abstract from the meaning of a message, that is, a single message does not provide any information. Instead, we always consider a set of possible symbols, and each of the symbols will occur with some probability. The less frequent a symbol, the more surprising is its occurrence and hence it bears more information. The amount of information of a source is described as follows:

Definition 1.2 (Shannon entropy) Let a source S emit the symbols $x_1, \dots, x_n$ with probabilities $p(x_1), \dots, p(x_n)$. Then the *Shannon entropy* of the source is given by

$$H(S) = -\sum_{i=1}^{n} p(x_i)\log_2 p(x_i).$$

In this definition, we have assumed that the symbols are emitted independently, that is, the probability of receiving a sequence $x_{i_1}x_{i_2}\cdots x_{i_m}$ of length m is given by

$$p(x_{i_1}x_{i_2}\cdots x_{i_m}) = p(x_{i_1})p(x_{i_2})\cdots p(x_{i_m}).$$

If there are some correlations between the symbols, the entropy of the source decreases when the original symbols are combined to new symbols. This is due to the fact that the entropy is maximal when all probabilities are equal. As an

example, we may consider any natural language. The entropy depends on whether we consider the single-letter entropy, or the entropy of pairs or triples of letters, whole words, or even sentences. Of course, the entropy does also depend on the language itself.

From now on, we fix the alphabet of the information source and assume that there are no correlations between the symbols. A very important concept in information theory is that of *ε -typical words*.

Theorem 1.1 (ε-typical words) Let S be a source with entropy $H(S)$. Given any $\varepsilon > 0$ and $\delta > 0$, we can find an N_0 such that the sequences of any length $N \geq N_0$ fall into two classes:

1) A set whose total probability is less than ε.
2) The remainder, all of whose members have probability p satisfying the inequality

$$\left| \frac{-\log_2 p}{N} - H(S) \right| < \delta.$$

Asymptotically, this means that a sequence either occurs with negligible probability, that is, is nontypical, or it is a so-called ε-typical word, which are approximately equally distributed.

Fixing the length N one can order the sequences by decreasing probability. For $0 < q < 1$, we define $n(q)$ as the minimum number of sequences of length N that accumulate a total probability q. Shannon has shown that in the limit of large N, this fraction is independent of q:

Theorem 1.2

$$\lim_{N \to \infty} \frac{\log_2 n(q)}{N} = H(S) \quad \text{for} \quad 0 < q < 1.$$

The quantity $\log_2 n(q)$ can be interpreted as the number of bits that are required to describe a sequence when considering only the most probable sequences with total probability q. From Theorem 1.1, we get that even for the finite length N, almost all words can be described in this way. The bounds for sending arbitrary sequences through the channel are given by Shannon's first fundamental coding theorem:

Theorem 1.3 (noiseless coding theorem) Given a source with entropy H (in bits per symbol) and a channel with capacity C (in bits per second), we can encode the output of the source in such a way as to transmit at the average rate $\frac{C}{H} - \varepsilon$ symbols per second over the channel where $\varepsilon > 0$ is arbitrarily small.

Conversely, it is not possible to transmit at an average rate greater than $\frac{C}{H}$.

The small defect ε compared to the maximal achievable transmission speed is due to the small extra information that is needed to encode the nontypical words of the source. An efficient scheme for encoding the output of the source is for example, the so-called *Huffman coding* [2]. In view of Theorem 1.1, one can also ignore the nontypical words that have a negligible total probability ε in the encoding, resulting in a small error (*lossy data compression*).

1.2.3 The Discrete Noisy Channel

A discrete noisy channel maps an input symbol x_i from the (finite) input alphabet $\mathcal{X}$ to an output symbol y_j from the output alphabet $\mathcal{Y}$. A common assumption is that the channel is memoryless, that is, the probability of observing a symbol y_j depends only on the last channel input x_i and nothing else. The size of the input and output alphabets need not be the same, as depicted in Figure 1.2. Given the channel output y_j, the task for the receiver is to determine the most likely input x_i to the channel. For this, we consider how much information the channel output provides about the channel input. First, we define some general quantities for pairs of random variables (see, e.g., [3]).

Definition 1.3 (joint entropy) The *joint entropy* $H(X, Y)$ of a pair of discrete random variables X and Y with joint distribution $p(x, y)$ is defined as

$$H(X, Y) = -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x, y)\log_2 p(x, y).$$

For the joint entropy, we consider the channel input and the channel output together as one symbol.

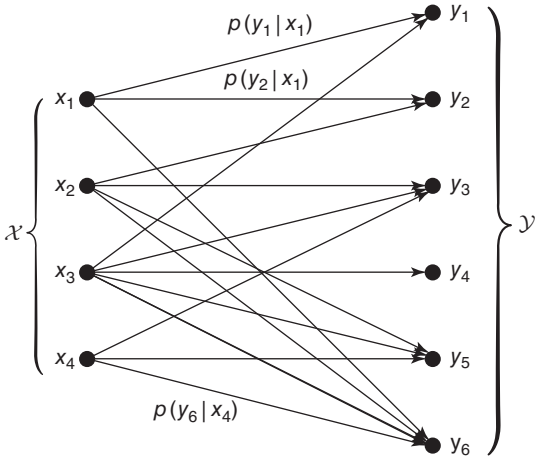

Figure 1.2 Schematic representation of a discrete memoryless channel. Arrows correspond to transitions with nonzero probability $p(y_j|x_i)$.

Definition 1.4 (conditional entropy) The *conditional entropy* $H(Y|X)$ of a pair of discrete random variables X and Y with joint distribution $p(x, y)$ is defined as

$$\begin{aligned} H(Y|X) &= \sum_{x\in\mathcal{X}} p(x)H(Y|X=x) \\ &= -\sum_{x\in\mathcal{X}} p(x) \sum_{y\in\mathcal{Y}} p(y|x)\log_2 p(y|x) \\ &= -\sum_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} p(x, y)\log_2 p(y|x). \end{aligned}$$

The conditional entropy is a measure for the information that we additionally get when considering both X and Y together and not only X. This is reflected by the following *chain rule* (see [3, Theorem 2.2.1]).

Theorem 1.4 (chain rule)

$$H(X, Y) = H(X) + H(Y|X).$$

Another important quantity in information theory is the *mutual information*.

Definition 1.5 (mutual information) The *mutual information* $I(X; Y)$ of a pair of discrete random variables X and Y is defined as

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

The relationship between entropy and mutual information is illustrated in Figure 1.3. From Theorem 1.4, we get the following equivalent expressions for the mutual information:

$$\begin{aligned} I(X; Y) &= H(X) + H(Y) - H(X, Y) \\ &= H(X) - H(X|Y) \\ &= H(Y) - H(Y|X). \end{aligned}$$

With this preparation, we are ready to define the capacity of a noisy discrete memoryless channel.

Definition 1.6 (capacity of a noisy discrete memoryless channel) The capacity of a discrete memoryless channel with joint input–output distribution $p(x, y)$ is

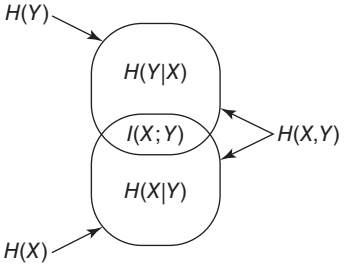

Figure 1.3 Relationship between entropy and mutual information.

defined as

$$C := \max_{p(x)} I(X;Y) = \max_{p(x)} (H(X) - H(X|Y)),$$

where the maximum is taken over all possible input distributions.

The justification of this definition is provided by Shannon's second fundamental coding theorem.

Theorem 1.5 (noisy coding theorem) Let S be a source with entropy $H(S)$ and let a discrete memoryless channel have the capacity C. If $H(S) < C$, then there exists an encoding scheme such that the output of the source can be transmitted over the channel with an arbitrarily small frequency of errors.

For the proof of this theorem, one considers a particular set of encoding schemes and then averages the frequency of errors. This average can be made arbitrarily small, implying that at least one of the encoding schemes must have a negligible error probability.

Before we turn our attention to the explicit construction of error-correcting codes, we consider a particular interesting channel.

Example 1.1 (binary symmetric channel (BSC)) The BSC maps the input symbols $\{0,1\}$ to the output symbols $\{0,1\}$. With probability $1-p$, the symbol is transmitted correctly; with probability p, the output symbol is flipped (see Figure 1.4).

For the capacity of the BSC, we compute

$$\begin{aligned} I(X;Y) &= H(Y) - H(Y|X) \\ &= H(Y) - \sum p(x) H(Y|X=x) \\ &= H(Y) - \sum p(x) H(p) \\ &= H(Y) - H(p) \\ &\leq 1 - H(p). \end{aligned}$$

Here we have used the *binary entropy function* $H(p)$ defined as

$$H(p) := -p\log_2 p - (1-p)\log_2(1-p).$$

The last inequality follows from the fact that the entropy of the binary variable Y is at most 1. From (1.2) it follows that the capacity of a BSC is at most $1 - H(p)$, and if the input distribution is uniform, this maximal capacity is achieved.

The generalization of the BSC to more than one input symbol is shown in Figure 1.4. Again, a symbol is transmitted correctly with probability $1-p$. If an error occurs, each of the, say, $m-1$ other symbols is equally likely, that is, it occurs with probability $q = p/(m-1)$. These types of channels are extremal in the sense that the transition probabilities only depend on whether a symbol is transmitted correctly or not. Hence, an incorrect symbol bears minimal information about the input symbol. Any deviation from this symmetry results in an increased capacity.

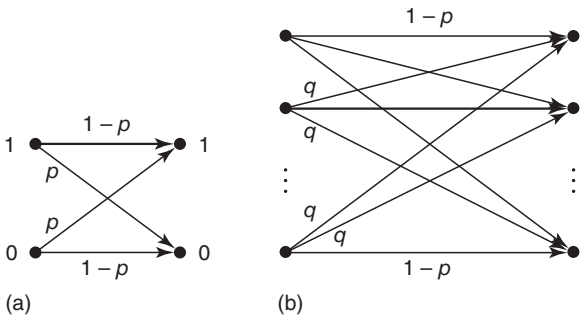

Figure 1.4 The binary symmetric channel (BSC) and its generalization, the uniform symmetric channel (USC). Each symbol is transmitted correctly with probability $1 - p$. If an error occurs, each of the other symbols is equally likely.

1.3 Linear Block Codes

1.3.1 Repetition Code

When sending information over a noisy channel, on the highest level of abstraction, we distinguish only the cases whether a symbol is transmitted correctly or not. Then the difference between the input sequence and the output sequence is measured by the *Hamming distance*.

Definition 1.7 (Hamming distance/weight) The *Hamming distance* between two sequences $\mathbf{x} = (x_1 \dots x_n)$ and $\mathbf{y} = (y_1 \dots y_n)$ is the number of positions where $\mathbf{x}$ and $\mathbf{y}$ differ, that is,

$$d_{\text{Hamming}}(\mathbf{x}, \mathbf{y}) := |\{i : 1 \le i \le n \mid x_i \ne y_i\}|.$$

If the alphabet contains a special symbol 0, we can also define the *Hamming weight* of a sequence that equals the number of nonzero positions.

In order to be able to correct errors, we use only a subset of all possible sequences. In particular, we may take a subset of all possible sequences of length n.

Definition 1.8 (block code) A *block code* $\mathcal{B}$ of length n is a subset of all possible sequences of length n over an alphabet $\mathcal{A}$, that is, $\mathcal{B} \subseteq \mathcal{A}^n$. The *rate* of the code is

$$R = \frac{\log|\mathcal{B}|}{\log|\mathcal{A}^n|} = \frac{\log|\mathcal{B}|}{n\log|\mathcal{A}|},$$

that is, the average number of symbols encoded by a codeword.

The simplest code that can be used to detect or correct errors is the *repetition code*. A repetition code with rate $1/2$ transmits every symbol twice. At the receiver, the two symbols are compared, and if they differ, an error is detected. Using this code over a channel with error probability p, the probability of an undetected error is p^2. Sending more than two copies of each symbol, we can decrease the probability of an undetected error even more. But at the same time, the rate of the code decreases since the number of codewords remains fixed while the length of the code increases.

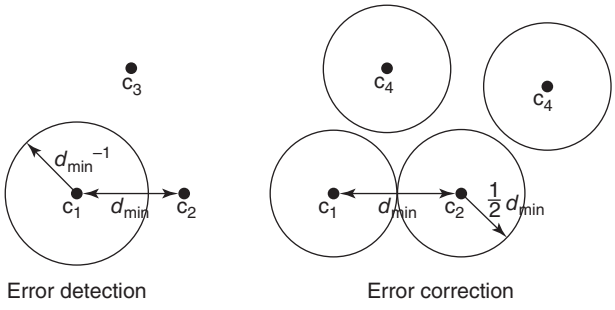

Figure 1.5 Geometry of the codewords. Any sphere of radius $d_{\min} - 1$ around a codeword contains exactly one codeword. The spheres of radius $\lfloor (d_{\min} - 1)/2 \rfloor$ are disjoint.

A repetition code can not only be used to detect errors but also to correct errors. For this, we send three copies of each symbols, that is, we have a repetition code with rate $1/3$. At the receiver, the three symbols are compared. If at most one symbol is wrong, the two error-free symbols agree and we assume that the corresponding symbol is correct. Again, increasing the number of copies sent increases the number of errors that can be corrected. For the general situation, we consider the distance between two words of the block code $\mathcal{B}$.

Definition 1.9 (minimum distance) The *minimum distance* of a block code $\mathcal{B}$ is the minimum number of positions in which two distinct codewords differ, that is,

$$d_{\min}(\mathcal{B}) := \min\{d_{\text{Hamming}}(\mathbf{x}, \mathbf{y}) : \mathbf{x}, \mathbf{y} \in \mathcal{B} \mid \mathbf{x} \neq \mathbf{y}\}.$$

The error-correcting ability of a code is related to its minimum distance.

Theorem 1.6 Let $\mathcal{B}$ be a block code with minimum Hamming distance d. Then one can either detect any error that acts on no more than d positions or correct any error that acts on no more than $\lfloor (d-1)/2 \rfloor$ positions.

Proof: From the definition of the minimum distance of the code $\mathcal{B}$ it follows that at least d positions have to be changed in order to transform one codeword into another. Hence, any error acting on less than $d-1$ positions can be detected. If strictly less than $d/2$ positions are changed, there will be a unique codeword that is closest in the Hamming distance. Hence, up to $\lfloor (d-1)/2 \rfloor$ errors can be corrected. The situation is illustrated in Figure 1.5.

1.3.2 Finite Fields

For a general block code $\mathcal{B}$ over a alphabet $\mathcal{A}$, we have to make a list of all codewords, that is, the description of the code is proportional to its size. In order to get a more efficient description—and thereby more efficient algorithms for encoding and decoding—we impose some additional structure. In particular, we require that the elements of the alphabet have the algebraic structure of a field, that is,

we can add, subtract, and multiply any two elements, and every nonzero element has a multiplicative inverse. First, we consider a finite field whose size is a prime number.

Proposition 1.1 (prime field) The integers modulo a prime number p form a finite field $\mathbb{F}_p$ with p elements.

Proof: It is clear that the modulo operation is a ring homomorphism, that is, it is compatible with addition, subtraction, and multiplication. It remains to show that any nonzero element has a multiplicative inverse. As p is a prime number, for any nonzero element b we have $\gcd(p, b) = 1$. By the extended Euclidean algorithm (see Table 1.1), there exist integers s and t such that $1 = \gcd(p, b) = sp + tb$. Hence, we get $tb = 1 \bmod p$, that is, t is the multiplicative inverse of b modulo p.

The smallest field is the binary field $\mathbb{F}_2$, which has only two elements 0 and 1. Note that the integers modulo a composite number do not form a field as some nonzero elements do not have a multiplicative inverse. For example, for the integers modulo 4 we have $2 \cdot 2 = 0 \bmod 4$.

In order to construct a field whose size is not a prime number, one uses the following construction.

Proposition 1.2 (extension field) Let $\mathbb{F}_p$ be a finite field with p elements, p prime. If $f(X) \in \mathbb{F}_p[X]$ is an irreducible polynomial of degree m, then the polynomials in $\mathbb{F}_p[X]$ modulo $f(X)$ form a finite field $\mathbb{F}_q$ with $q = p^m$ elements.

Proof: The remainder of the division by the polynomial $f(X)$ of degree m can be any polynomial of degree strictly less than m. Hence, we obtain p^m different elements. Again addition, subtraction, and multiplication of two elements are performed over the polynomial ring, and the result is reduced modulo $f(X)$. For the computation of the multiplicative inverse, we use the extended Euclidean

Table 1.1 The extended Euclidean algorithm (see [4]).

```
EUCLID(a_0,a_1)
  s_0 ← 1; t_0 ← 0;
  s_1 ← 0; t_1 ← 1;
  i ← 1;
  while a_i does not divide a_{i−1} do
    q ← a_{i−1} div a_i;
    a_{i+1} ← a_{i−1} − q a_i;
    s_{i+1} ← s_{i−1} − q s_i;
    t_{i+1} ← t_{i−1} − q t_i;
    i ← i + 1;
  end while
  return a_i, s_i, t_i;
end
```

algorithms of Table 1.1. The condition that $f(X)$ is an irreducible polynomial implies that $f(X)$ cannot be written as the product of two nonconstant polynomials. So again, for any nonzero element $b(X)$ we have $\gcd(b(X), p(X)) = 1$.

It can be shown that for any prime number p and for any positive integer m, there exists an irreducible polynomial of degree p over $\mathbb{F}_p$, that is, for any prime power $q = p^m$, there exists a finite field of that size. Furthermore, it can be shown that any finite field can be obtained by the construction of Proposition 1.2. Hence, we get (see, e.g., [5])

Theorem 1.7 A finite field of size s exists if and only if s is a prime power, that is, $s = p^m$ for some prime number p and some positive integer m.

Example 1.2 The polynomial $f(X) = X^2 + X + 1$ has no zero over the integers modulo 2 and is hence irreducible. The resulting field $\mathbb{F}_4 = \mathbb{F}_2[X]/(f(X))$ has four elements $\{0, 1, X, X + 1\}$ which may also be denoted as $\mathbb{F}_4 = \{0, 1, \omega, \omega^2\}$, where ω is a root of $f(X)$, that is, $\omega^2 + \omega + 1 = 0$.

Example 1.3 The polynomial $f(X) = X^2 + 1$ has no zero over the integers modulo 3 and is hence irreducible. The resulting field $\mathbb{F}_9 = \mathbb{F}_3[X]/(f(X))$ has nine elements $\{0, 1, 2, X, X + 1, X + 2, 2X, 2X + 1, 2X + 2\}$. Note that here the powers of a root α of $f(X)$ do not generate all nonzero elements as $\alpha^2 = -1$ and hence $\alpha^4 = 1$. Instead, we may use the powers of the element $\beta = \alpha + 1$.

1.3.3 Generator and Parity Check Matrix

In order to get a more efficient description of a block code $\mathcal{B}$ of length n, we consider only codes whose alphabet is a finite field $\mathbb{F}_q$. Furthermore, we require that the code $\mathcal{C}$ forms a linear vector space over the field $\mathbb{F}_q$, that is,

$$\forall \mathbf{x}, \mathbf{y} \in \mathcal{B}\ \forall \alpha, \beta \in \mathbb{F}\colon \alpha\mathbf{x} + \beta\mathbf{y} \in \mathcal{B}.$$

This implies that the code has q^k elements for some k, $0 \le k \le n$. We will use the notation $\mathcal{B} = [n, k]_q$. Instead of listing all q^k elements, it is sufficient to specify a basis of k linear independent vectors in $\mathbb{F}_q^n$. Alternatively, the linear space $\mathcal{B}$ can be given as the solution of $n - k$ linearly independent homogeneous equations.

Definition 1.10 (generator matrix/parity check matrix) A *generator matrix* of a linear code $\mathcal{B} = [n, k]_q$ over the field $\mathbb{F}_q$ is a matrix G with k rows and n columns of full rank whose row-span is the code.

A *parity check matrix* of a linear code $\mathcal{B} = [n, k]_q$ is a matrix H with $n - k$ rows and n columns of full rank whose row null-space is the code.

The generator matrix with $k \times n$ entries provides a compact description of a code with q^k elements. Moreover, encoding of information sequences $\mathbf{i} \in \mathbb{F}_q^k$ of length k corresponds to the linear map given by G, that is,

$$\mathbf{i} \mapsto \mathbf{c} := \mathbf{i}\, G.$$

The parity check matrix H can be used to check whether a vector lies in the code.

Proposition 1.3 (error syndrome) Let H be a parity check matrix of a linear code $\mathcal{B} = [n, k]_q$. Then, a vector $\mathbf{v} \in \mathbb{F}_q^n$ is a codeword if and only if the *error syndrome* $\mathbf{s}$ given by

$$\mathbf{s} := \mathbf{v}H^t$$

is zero. Moreover, the syndrome $\mathbf{s}$ depends only on the error.

Proof: The code $\mathcal{B}$ is the row null-space of H, that is, for any codeword $\mathbf{c} \in \mathcal{B}$ we get $\mathbf{c}H^t = \mathbf{0}$. If $\mathbf{v}$ is a codeword with errors, we can always write $\mathbf{v} = \mathbf{c} + \mathbf{e}$, where $\mathbf{v}$ is a codeword and $\mathbf{e}$ corresponds to the error. Then, we compute

$$\mathbf{s} = \mathbf{v}H^t = (\mathbf{c} + \mathbf{e})H^t = \mathbf{c}H^t + \mathbf{e}H^t = \mathbf{e}H^t.$$

The reason for defining the parity check matrix H as a matrix with n columns and $n - k$ rows and not as its transpose is motivated by the following.

Proposition 1.4 (dual code) Let $\mathcal{B} = [n, k]_q$ be a linear code over the finite field $\mathbb{F}_q$. Then the *dual code* $\mathcal{B}^\perp$ is a code of length n and dimension $n - k$ given by

$$\mathcal{B}^\perp = \{\mathbf{v} : \mathbf{v} \in \mathbb{F}_q^n \mid \mathbf{v} \cdot \mathbf{c} = 0 \quad \text{for all } \mathbf{c} \in \mathcal{B}\}.$$

Here $\mathbf{v} \cdot \mathbf{c} = \sum_{i=1}^n v_i c_i$ denotes the Euclidean inner product on $\mathbb{F}_q^n$. If G is a generator matrix and H a parity check matrix for $\mathcal{B}$, then G is a parity check matrix and H is a generator matrix for $\mathcal{B}^\perp$.

As we have seen in Theorem 1.6, the minimum distance of a code is a criterion for its error-correcting ability. For linear codes, the minimum distance equals the minimum Hamming weight of the nonzero codewords as

$$\begin{aligned} d_{\text{Hamming}}(\mathbf{x}, \mathbf{y}) &= d_{\text{Hamming}}(\mathbf{x} - \mathbf{y}, \mathbf{y} - \mathbf{y}) = d_{\text{Hamming}}(\mathbf{x} - \mathbf{y}, \mathbf{0}) \\ &= \text{wgt}_{\text{Hamming}}(\mathbf{x} - \mathbf{y}). \end{aligned}$$

The minimum Hamming weight of a linear code can be computed using the parity check matrix.

Proposition 1.5 If any $d - 1$ columns in the parity check matrix H of a linear code are linearly independent, then the minimum distance of the code is at least d.

Proof: Assume that we have a nonzero codeword $\mathbf{c}$ with Hamming weight $d - 1$, that is, there are $d - 1$ nonzero positions $i_1, \ldots, i_{d-1}$ in $\mathbf{c}$. From $\mathbf{c}H^t = \mathbf{0}$ it follows that $c_{i_1} h^{(i_1)} + \cdots + c_{i_{d-1}} h^{(i_{d-1})} = 0$, where $h^{(i)}$ denotes the ith column of H. This contradicts the fact that any $d - 1$ columns in H are linearly independent.

1.3.4 Hamming Codes

The last proposition can be used to construct codes. For a single-error-correcting code, we require $d \geq 3$. This implies that any two columns in H have to be linearly independent, that is, no column is a scalar multiple of another column. If we fix

the redundancy $m = n - k$, it is possible to find $(q^m - 1)/(q - 1)$ vectors with this property, which can be combined to a parity check matrix H. This construction gives the following class of single-error-correcting codes (see [6, 7]).

Proposition 1.6 (Hamming code) The mth Hamming code over $\mathbb{F}_q$ is a linear code of length $n = (q^m - 1)/(q - 1)$ and dimension $k = (q^m - 1)/(q - 1) - m$. The parity check matrix H is formed by all normalized nonzero vectors of length m, that is, the first nonzero coordinate of the vectors is 1. The minimum distance of the code is 3.

For binary Hamming codes, the parity check matrix H consists of all $2^m - 1$ nonzero vectors of length m. If we order those columns in such a way that the ith column equals the binary expansion $\mathrm{bin}(i)$ of i, error correction is particularly easy. If $\mathbf{e}$ is an error of weight 1, then the syndrome $\mathbf{s} = \mathbf{e}H^t$ equals the ith column of H and hence the binary expansion of i. Therefore, the syndrome directly provides the position of the error.

Example 1.4 The third binary Hamming code has parameters $[7, 4, 3]$. A parity check matrix is

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}.$$

For an error at the fifth position, we have $\mathbf{e} = (0, 0, 0, 0, 1, 0, 0)$ and $\mathbf{s} = \mathbf{e}H = (1, 0, 1) = \mathrm{bin}(5)$.

Usually, a received vector will be decoded as the codeword, which is closest in the Hamming distance. In general, decoding an arbitrary linear binary code is an NP hard problem [8]. More precisely, it was shown that it is an NP complete problem to decide whether there is a vector $\mathbf{e} \in \mathbb{F}_2^n$, which corresponds to a given syndrome $\mathbf{s} \in \mathbb{F}_2^k$ and whose Hamming weight is at most w. Hence, we cannot expect to have an efficient general algorithm for decoding.

Instead, by exhaustive search we can precompute an error vector of minimal Hamming weight corresponding to each syndrome. For this, we first arrange all codewords as the first row of an array, where the all-zero codeword is the first element. Among the remaining vectors of length n, we pick a vector $\mathbf{e}_1$ with minimal Hamming weight. This vector is the first element of the next row in our array. The remaining entries of this row are obtained by adding the vector $\mathbf{e_1}$ to the corresponding codeword in the first row. This guarantees that all elements of a row correspond to the same syndrome. We proceed until all q^n vectors have been arranged into an array with q^{n-k} rows and q^k columns, the so-called *standard array*. The elements in the first column of the standard array are called *coset leaders*, having minimal Hamming weight among all vectors in a row. Table 1.2 shows the standard array of a binary code $\mathcal{B} = [7, 3, 4]$, which is the dual of the Hamming code of Example 1.4. Actually, the code $\mathcal{B}$ is a subcode of the Hamming code. In the first row, 16 codewords are listed. For the next seven rows, the coset leader is the unique vector of Hamming weight 1 in each coset, reflecting the fact

Table 1.2 Standard array for decoding the code $\mathcal{B} = [7, 3, 4]$, the dual of a binary Hamming code.

0000000	**0001111**	**0110011**	**0111100**	**1010101**	**1011010**	**1100110**	**1101001**
0000001	0001110	0110010	0111101	1010100	1011011	1100111	1101000
0000010	0001101	0110001	0111110	1010111	1011000	1100100	1101011
0000100	0001011	0110111	0111000	1010001	1011110	1100010	1101101
0001000	0000111	0111011	0110100	1011101	1010010	1101110	1100001
0010000	0011111	0100011	0101100	1000101	1001010	1110110	1111001
0100000	0101111	0010011	0011100	1110101	1111010	1000110	1001001
1000000	1001111	1110011	1111100	0010101	0011010	0100110	0101001
0110000	0111111	0000011	0001100	1100101	1101010	1010110	1011001
1000001	1001110	1110010	1111101	0010100	0011011	0100111	0101000
1000010	1001101	1110001	1111110	0010111	0011000	0100100	0101011
1000100	1001011	1110111	1111000	0010001	0011110	0100010	0101101
1001000	1000111	1111011	1110100	0011101	0010010	0101110	0100001
1010000	1011111	1100011	1101100	0000101	0001010	0110110	0111001
1100000	1101111	1010011	1011100	0110101	0111010	0000110	0001001
1110000	1111111	1000011	1001100	0100101	0101010	0010110	0011001

that the code can correct a single error. For the next seven rows, the coset leader has weight 2, but each coset contains three vectors of weight 2. Hence, decoding succeeds only in one out of three cases. In the final row, we have even seven vectors of weight 3.

1.4 Further Aspects

We have seen that the Hamming code is a code for which the correction of errors is rather simple, but it can only correct a single error. On the other hand, using an arbitrary linear code, the problem of error correction is NP complete. But luckily, there are other families of error-correcting codes for which efficient algorithms exist to correct at least all errors of bounded weight. More about these codes can be found in any textbook on coding theory or the book by MacWilliams and Sloane [7], which is an excellent reference for the theory of error-correcting codes.

References

1 Shannon, C.E. (1948) A mathematical theory of communication. *Bell Syst. Tech. J.*, **27 (3)**, 379–423, 623–656, doi: 10.1002/j.1538-7305.1948.tb01338.x.

2 Huffman, D.A. (1952) A method for the construction of minimum-redundancy codes. *Proc. Inst. Radio Eng.*, **40**, 1098–1101.
3 Cover, T.M. and Thomas, J.A. (1991) *Elements of Information Theory*, John Wiley & Sons, Inc., New York.
4 Aho, A.V., Hopcroft, J.E., and Ullman, J.D. (1974) *The Design and Analysis of Computer Algorithms*, Addison Wesley, Reading, MA.
5 Jungnickel, D. (1993) *Finite Fields: Structure and Arithmetics*, BI-Wissenschaftsverlag, Mannheim.
6 Hamming, R.W. (1986) *Coding and Information Theory*, Prentice-Hall, Englewood Cliffs, NJ.
7 MacWilliams, F.J. and Sloane, N.J.A. (1977) *The Theory of Error-Correcting Codes*, North-Holland, Amsterdam.
8 Berlekamp, E.R., McEliece, R.J., and van Tilborg, H.C.A. (1978) On the inherent intractability of certain coding problems. *IEEE Trans. Inf. Theory*, **24** (3), 384–386.