Chapter 10

Coding Theory[1]

10.1 INFORMATION TRANSMISSION

In this chapter we provide a brief overview of coding theory. Our concern will be with two aspects of the use of codes: to ensure the secrecy of transmitted messages and to detect and correct errors in transmission. The methods we discuss have application to communications with computers, with distant space probes, and with missiles in launching pads; to electronic commerce; to optical/magnetic recording; to genetic codes; and so on. Today, the development of coding theory is closely related to the explosion of information technology, with applications to the Internet and the "next generation of network technologies." The rapid development of a myriad of networked devices for computing and telecommunications presents challenging and exciting new issues for coding theory. For many references on the applications of coding theory, and in particular to those having to do with communication with computers, see MacWilliams and Sloane [1983]. For more detailed treatments of coding theory as a whole, in addition to MacWilliams and Sloane [1983], see Berlekamp [1968], Blake and Mullin [1975], Cameron and van Lint [1991], Goldie and Pinch [1991], Hill [1986], Peterson [1961], Peterson and Weldon [1972], Pless [1998], van Lint [1999], or Welsh [1988]. For good short treatments, see Dornhoff and Hohn [1978] or Fisher [1977].

The basic steps in information transmission are modeled in Figure 10.1. We imagine that we start with a "word," an English word or a word already in some code, for example a bit string. In step (a) we encode it, usually into a bit string. We then transmit the encoded word over a transmission channel in step (b). Finally, the received word is decoded in step (c). This model applies to transmission over physical communication paths such as telegraph lines or across space via radio waves. However, as MacWilliams and Sloane [1983] point out, a similar analysis applies, for example, to the situation when data are stored in a computer and later retrieved and to many other data transmission applications in the modern world

[1]Sections 10.1–10.3 form the basis for this chapter. The reader interested in a brief treatment may then go directly to Section 10.5 (which depends on Chapter 9).

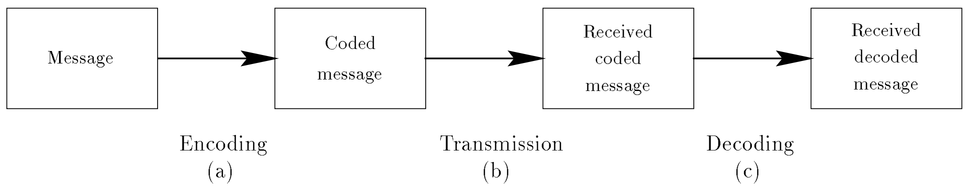

Figure 10.1: The basic steps in information transmission.

of telecommunications. As Fisher [1977] points out, the analysis also applies to communication of visual input into the retina (patterns of photons). The input is encoded into electrical impulses in some of the cells in the retina, and these impulses are transmitted through neurons to a visual area of the brain, where they are "decoded" as a visual pattern. There are many other applications as well.

We shall assume that the only errors which occur are in step (b) and are caused by the presence of noise or by weak signals. In Section 10.2 we discuss steps (a) and (c), the encoding and decoding steps, without paying attention to errors in transmission. In Section 10.3 we begin to see how to deal with such errors. In Section 10.4 we demonstrate how to use the encoding and decoding process to detect and correct errors in transmission. Finally, in Section 10.5 we discuss the use of block designs to obtain error-detecting and error-correcting codes. That section should be omitted by the reader who has skipped Chapter 9.

Much of the emphasis in this chapter is on the existence question: Is there a code of a certain kind? However, we also deal with the optimization question: What is the best or richest or largest code of a certain kind?

10.2 ENCODING AND DECODING

Sometimes, the encoding step (a) of Figure 10.1 must produce a coded message from a message containing "sensitive" information. In this case, we use the terms "encrypting" and "decrypting" to describe steps (a) and (c) of Figure 10.1. However, we shall use the terms "encode" and "decode" throughout. The field called *cryptography* is concerned with such encoding and decoding, and also with deciphering received coded messages if the code is not known. We discuss the encoding and decoding problems of cryptography here but not the deciphering problem.

A message to be encoded involves a sequence of symbols from some *message alphabet* A. The encoded message will be a sequence of symbols from a *code alphabet* B, possibly the same as the message alphabet. A simple encoding rule will encode each symbol a of A as a symbol $E(a)$ of B.

Example 10.1 Caesar Cyphers If A and B are both the 26 uppercase letters of the alphabet, a simple encoding rule $E(a)$ might take $E(a)$ to be the letter following a, with $E(Z) = A$. Thus, the message

$$\text{DEPOSIT SIX MILLION DOLLARS} \tag{10.1}$$

would be encoded as

EFQPTJU TJY NJMMJPO EPMMBST.

If A is as above but $B = \{0, 1, 2, \ldots, 25\}$, we could take $E(a)$ = the position of a in the alphabet + 2, where Z is assumed to have position 0. Here $24 + 2$ is interpreted as 0 and $25 + 2$ as 1. (Addition is modulo 26, to use the terminology of Section 9.3.1.) Thus, $E(\text{D}) = 6$. Similar encodings were used by Julius Caesar about 2000 years ago. They are now called *Caesar cyphers.* ■

Any function $E(a)$ from A into B will suffice for encoding provided that we can adequately decode, that is, find a from $E(a)$. To be able to do so unambiguously, we cannot have $E(a) = E(b)$ if $a \neq b$; that is, $E(a)$ must be one-to-one.

It is frequently useful to break a long message up into block of symbols rather than just one symbol, and encode the message in blocks. For instance, if we use blocks of length 2, the message (10.1) becomes

$$\text{DE} \quad \text{PO} \quad \text{SI} \quad \text{TS} \quad \text{IX} \quad \text{MI} \quad \text{LL} \quad \text{IO} \quad \text{ND} \quad \text{OL} \quad \text{LA} \quad \text{RS}. \tag{10.2}$$

We then encode each sequence of two symbols, that is, each block.[2] One way to encode blocks uses matrices, as the next example shows.

Example 10.2 Matrix Codes Let us replace each letter of the alphabet by a number representing its position in the alphabet (with Z having position 0). Then a block corresponds to a vector. For instance, the block DE above corresponds to the vector $(4, 5)$. Suppose that all blocks have length m. Let $\mathbf{M}$ be an $m \times m$ matrix, for instance

$$\mathbf{M} = \begin{pmatrix} 2 & 3 \\ 1 & 2 \end{pmatrix}.$$

Then we can encode a block a as a block $a\mathbf{M}$. In our case, we encode a block (i, j) as a block $E(i, j) = (i, j)\mathbf{M}$. Hence, DE or $(4, 5)$ gets encoded as

$$(4, 5)\begin{pmatrix} 2 & 3 \\ 1 & 2 \end{pmatrix} = (13, 22).$$

The reader can check that the message (10.1), when broken up as in (10.2), now gets encoded as

$$13, 22, 47, 78, 47, 75, 59, 98, 42, 75, 35, 57, 36, 60, 33, 57, 32, 50, 42, 69, 25, 38, 55, 92.$$

In general, the procedure we have defined is called *matrix encoding.* It is highly efficient. Moreover, decoding is easy. To decode, we break a coded message into blocks b, and find a so that $a\mathbf{M} = b$. We can unambiguously decode provided that $\mathbf{M}$ has an inverse. For then $a = b\mathbf{M}^{-1}$. In our example,

$$\mathbf{M}^{-1} = \begin{pmatrix} 2 & -3 \\ -1 & 2 \end{pmatrix}.$$

[2]To guarantee that a message will always break up into blocks of the same size, we can always lengthen the message by adding a recognizable "closing" string of copies of the same letter, for example, Z or ZZZ.

For instance, if we have the encoded block (6, 11), we find that it came from

$$(6,11)\mathbf{M}^{-1} = (6,11)\begin{pmatrix} 2 & -3 \\ -1 & 2 \end{pmatrix} = (1,4) = \text{AD}.$$

Note that if we have the encoded block (3, 4), decoding gives us

$$(3,4)\mathbf{M}^{-1} = (2,-1).$$

Since $(2,-1)$ does not correspond to any pair of letters, we can only conclude that an error was made in sending us the message that included the block (3, 4). Much of the emphasis in this chapter is on ways to detect errors, as we have done here. ■

In general, suppose that a message is broken up into blocks of length k. Let A^k consist of all blocks (sequences) of length k from the message alphabet A and B^n consist of all blocks (sequences) of length n from the code alphabet B. Let $\mathcal{A}$ be a subset of A^k called the set of *message blocks*. In most practical applications, $\mathcal{A} = A^k$, and we shall assume this unless stated explicitly otherwise. A *block code* or $k \to n$ *block code* is a one-to-one function $E: \mathcal{A} \to B^n$. The set C of all $E(a)$ for a in $\mathcal{A}$ is defined to be the set of *codewords*. Sometimes C alone is called the *code*. In Example 10.2, $k = n = 2$,

$$\begin{aligned} A &= \{\text{A}, \text{B}, \cdots, \text{Z}\}, \\ B &= \{0, 1, \cdots, 25\}, \end{aligned}$$

and $\mathcal{A} = A^k$. The blocks (13, 22) and (6, 11) are codewords. However, the block (3, 4) is not a codeword. In most practical examples, A and B will both be $\{0,1\}$, messages will be bit strings, and the encoding will take bit strings of length k into bit strings of length n. We shall see in Section 10.4 that taking $n > k$ will help us in error detection and correction.

Example 10.3 A Repetition Code Perhaps the simplest way to encode a message is to repeat the message. This type of encoding results in a *repetition code*. Suppose that we define $E: A^k \to A^{pk}$ by

$$E(a_1a_2\cdots a_k) = a_1a_2\cdots a_ka_1a_2\cdots a_k\cdots a_1a_2\cdots a_k,$$

where we have p copies of $a_1a_2\cdots a_k$. For instance, suppose that $k = 4$ and $p = 3$. Then $E(a_1a_2a_3a_4) = a_1a_2a_3a_4a_1a_2a_3a_4a_1a_2a_3a_4$. This is an example of a triple repetition code or $k \to 3k$ block code. In such a code, it is easy to detect errors by comparing the successive elements of A^k in the coded message received. It is even possible to use repetition codes to correct errors. We can simply use the majority rule of decoding. We pick for the ith digit in the message that letter from the message alphabet that appears most often in that place among the p copies. For instance, if $k = 4$, $p = 3$, and we receive the message *axybauybaxvb*, then since x appears a majority of times in the second position and y appears a majority of times in the third position, we "correct" the error and interpret the original message as *axyb*. We shall find more efficient ways to correct and detect errors later in this chapter. ■

We close this section by giving one more example of a block code called a *permutation code.*

Example 10.4 Permutation Codes Suppose that $A = B = \{0,1\}$ and π is a permutation of $\{1,2,\ldots,k\}$. Then we can define $E : A^k \to B^k$ by taking $E(a_1a_2\cdots a_k) = a_{\pi(1)}a_{\pi(2)}\cdots a_{\pi(k)}$. For instance, suppose that $k = 3$ and $\pi(1) = 2, \pi(2) = 3, \pi(3) = 1$. Then $E(a_1a_2a_3) = a_2a_3a_1$, so $E(011) = 110, E(101) = 011$, and so on. ■

It is easy to see (Exercises 15 and 16) that every such permutation code and repetition code is a matrix encoding.

EXERCISES FOR SECTION 10.2

1. Suppose that
$$\mathbf{M} = \begin{pmatrix} 1 & 0 \\ 2 & 4 \end{pmatrix}.$$
Encode the following expressions using the matrix code defined by $\mathbf{M}$.

   (a) AX (b) UV (c) BUNNZ HIL

   (d) SELL ALL SHARES OF IBM (e) INVEST TWO MILLION

   (f) ABORT THE MISSIONZ (*Note*: Z has been added at the end because there is an odd number of letters.)

2. Suppose that
$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}.$$
Using $\mathbf{M}$, find the codeword $x_1x_2\cdots x_n$ corresponding to each of the following message words $a_1a_2\cdots a_k$.

   (a) 11 (b) 10 (c) 01 (d) 00

3. Suppose that
$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}.$$
Repeat Exercise 2 for the following message words.

   (a) 111 (b) 101 (c) 000

4. Suppose that
$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}.$$
Repeat Exercise 2 for the following message words.

   (a) 1111 (b) 1000 (c) 0001

5. Suppose that

$$\mathbf{M} = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 1 & 2 & 1 \end{bmatrix}.$$

Encode the following expressions using the matrix code defined by $\mathbf{M}$.

(a) ABC (b) XAT (c) TTU

(d) BUY TWENTY SHARES (e) SEND THE MESSAGE AT EIGHT

(f) OBSERVE THE TRANSFERSZZ. (*Note*: Two Z's have been added to make the number of letters divisible by 3.)

6. Repeat Exercise 1 with

$$\mathbf{M} = \begin{pmatrix} 4 & 6 \\ 1 & 2 \end{pmatrix}.$$

7. Repeat Exercise 5 with

$$\mathbf{M} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 2 & 4 \\ 2 & 1 & 0 \end{bmatrix}.$$

8. If $\mathbf{M}$ of Exercise 1 is used in a matrix code, decode the following or show that an error was made; that is, this is not proper code.

(a) $(5,8)$ (b) $(8,12)$ (c) $(1,8)$ (d) $(7,12)$

(e) $16,48,44,60$ (f) $8,16,8,12$ (g) $6,10,51,69,20,20$

9. If $\mathbf{M}$ of Exercise 5 is used in a matrix code, decode the following or show that an error was made; that is, this is not proper code.

(a) $(8,14,14)$ (b) $(17,35,23)$ (c) $(24,47,30)$

(d) $52,95,102,46,72,88$ (e) $48,91,70,15,27,23$

10. Repeat Exercise 8 for $\mathbf{M}$ of Exercise 6.

11. Repeat Exercise 9 for $\mathbf{M}$ of Exercise 7.

12. For each of the following permutations of $\{1,2,\ldots,k\}$, find the values of E indicated in the corresponding permutation code.

(a) $k=3, \pi(1)=3, \pi(2)=2, \pi(3)=1$.

i. $E(110)$ ii. $E(010)$ iii. $E(011)$

(b) $k=4, \pi(1)=3, \pi(2)=4, \pi(3)=1, \pi(4)=2$.

i. $E(0110)$ ii. $E(1100)$ iii. $E(0111)$

(c) $k=5, \pi(1)=5, \pi(2)=1, \pi(3)=4, \pi(4)=2, \pi(5)=3$.

i. $E(00001)$ ii. $E(10101)$ iii. $E(11010)$

13. Find C if $\mathcal{A}$ consists of the blocks $010, 111, 001$, and 110 and $E(abc)$ is $a+b, a+c, b+c$, where addition is modulo 2.

14. Make up an example of a $1 \to 2$ block code.

15. Show that every permutation code is a matrix encoding.

16. Show that every repetition code is a matrix encoding.

10.3 ERROR-CORRECTING CODES

10.3.1 Error Correction and Hamming Distance

In this section we study the use of codes to detect and correct errors in transmission. In particular, we study step (b) of Figure 10.1. We assume that we start with an encoded message, which has been encoded using a block code, with blocks in the code alphabet having length n. For concreteness, we assume that the encoded message is a bit string, so all blocks are bit strings of length n. We speak of *binary codes* or *binary block codes* or *binary n-codes.* In Exercise 7 we modify the assumption and take encoded messages that are strings from an alphabet $\{0, 1, \ldots, q-1\}$. We then speak of *q-ary codes.* The message is sent by blocks. We assume that there are no errors in encoding, so the only blocks that are ever sent are codewords. Let us suppose that the only possible errors that can take place in transmission are interchanges of the digits 0 and 1. Other errors, for example deletion or addition of a digit, will be disregarded. They are easily detected since all blocks have the same length. Following common practice, we shall assume that the probability of changing 1 to 0 is the same as the probability of changing 0 to 1, and that the probability of an error is the same at each digit, independent of any previous errors that may have occurred.[3] In this case, we speak of a *binary symmetric channel.* For the implications of these assumptions, see Section 10.3.3.

Recall that we will be sending only codewords. If we ever receive a block that is not a codeword, we have *detected* an error in transmission. For instance, if the codeword 10010 is sent and the block 10110 is received, and if 10110 is not a codeword, we know there was an error. There are situations where we wish to *correct* the error, that is, guess what codeword was sent. This is especially the case when we cannot ask for retransmission, for instance in transmission of a photograph from a space probe (such as occurred on the Mariner 9 Mars probe, whose code we discuss below), or if the transmission is based on an old magnetic tape. Error-detecting and error-correcting codes have many applications. For instance, simple error-detecting codes called parity check codes (see Example 10.6) were used on the earliest computers: UNIVAC, Whirlwind I, and the IBM 650 (Wakerly [1978]). Indeed, it was the idea that coding methods could correct errors in the early computers used at Bell Laboratories that led Hamming [1950] to develop error-correcting codes. Such codes are now used extensively in designing fault-tolerant computers. (For references on the applications of error-detecting and error-correcting codes to computing, see MacWilliams and Sloane [1983], Pless [1998], Poli and Huguet [1992], Sellers, Hsiao, and Bearnson [1968], or Wakerly [1978], and the survey articles by Avizenius [1976] and Carter and Bouricius [1971].) Such codes were also fundamental to the design of the compact disc, which revolutionized the music industry in the 1980s (as we note in Section 10.5).

There are good ways of designing error-correcting codes. Suppose that we choose the code so that in the set of codewords, no two codewords are too "close" or too

[3] Errors that don't occur randomly sometimes occur in bursts, i.e., several errors close together. See Bossert, *et al.* [1997] for more on burst-error-correcting codes.

similar. For example, suppose that the only codewords are

$$000000, \quad 010101, \quad 101010, \quad \text{and} \quad 111111. \tag{10.3}$$

Then we would be very unlikely to confuse two of these, and a message that is received which is different from one of the codewords could readily be interpreted as the codeword to which it is closest.

Let us make this notion of closest more precise. The *Hamming distance* between two bit strings of the same length is the number of digits on which they differ.[4] Hence, if $d(\cdot,\cdot)$ denotes this distance, we have

$$d(000000, 010101) = 3.$$

Our aim is to find a set of codewords with no two words too close in terms of the Hamming distance.

Suppose that we have found a set of codewords (all of the same length, according to our running assumption). Suppose that the smallest distance between two of these codewords is d. Then we can *detect* all errors of $d-1$ or fewer digits. For if $d-1$ or fewer digits are interchanged, the resulting bit string will not be a codeword and we can recognize or detect that there was an error. For example, if the possible codewords are those of (10.3), then d is 3 and we can detect errors in up to two digits. Suppose we use the strategy that if we receive a word which is not a codeword, we interpret it as the codeword to which it is closest in terms of Hamming distance. (In case of a tie, choose arbitrarily.) This is called the *nearest-neighbor rule.* For example, if the codewords are those of (10.3), and we receive the word 010000, we would interpret it as 000000, since

$$d(010000, 000000) = 1$$

whereas for every other codeword α,

$$d(010000, \alpha) > 1.$$

Using the nearest-neighbor rule, we can *correct* all errors that involve fewer than $d/2$ digits. For if fewer than $d/2$ digits are interchanged, the resulting bit string is closest to the correct codeword (the one transmitted), and so is interpreted as that codeword by the nearest-neighbor rule. Hence, we have an *error-correcting code.* A code that can correct up to t errors is called a *t-error-correcting code.*

We summarize our results as follows.

Theorem 10.1 Suppose that d is the minimum (Hamming) distance between two codewords in the binary code C. Then the code C can detect up to $d-1$ errors and, using the nearest-neighbor rule, can correct up to $\lceil (d/2) - 1 \rceil$ errors.

Our next theorem says that no error-correcting rule can do better than the nearest-neighbor rule.

[4] The Hamming distance was named after R. W. Hamming, who wrote the pioneering paper (Hamming [1950]) on error-detecting and error-correcting codes.

Theorem 10.2 Suppose that d is the minimum (Hamming) distance between two codewords in the binary code C. Then no error-detecting rule can detect more than $d-1$ errors and no error-correcting rule can correct more than $t = \lceil (d/2) - 1 \rceil$ errors.

Proof. Since d is the minimum (Hamming) distance, there are two codewords α and β with $d(\alpha, \beta) = d$. If α is sent and d errors occur, then β could be received. Thus, no error would be detected. Next, note that an error-correcting rule assigns a codeword $R(\lambda)$ to each word λ (each bit string λ of length n). Let γ be a word different from α and β with $d(\alpha, \gamma) \leq t+1$ and $d(\beta, \gamma) \leq t+1$. Then $R(\gamma)$ must be some codeword. This cannot be both α and β. Say that it is not α without loss of generality. Then α could be sent, and with at most $t+1$ errors, γ could be received. This would be "corrected" as $R(\gamma)$, which is not α. Q.E.D.

Since the nearest-neighbor rule (and any other reasonable rule) does not allow us to correct as many errors as we could detect, we sometimes do not attempt to correct errors but only to detect them and ask for a retransmission if errors are detected. This is the case in such critical situations as sending a command to fire a missile or to change course in a space shot.

Suppose that we know how likely errors are to occur. If they are very likely to occur, we would like to have a code with minimum Hamming distance d fairly large. If errors are unlikely, this minimum distance d could be smaller, perhaps as small as 3 if no more than 1 error is likely to occur. In general, we would like to be able to construct sets of codewords of given length n with minimum Hamming distance a given number d. Such a set of codewords in a binary code will be called an (n, d)*-code.*

There always is an (n, d)-code if $d \leq n$. Let the set of codewords be

$$\underbrace{000\cdots0}_{n \text{ 0's}} \quad \text{and} \quad \underbrace{111\cdots1}_{d \text{ 1's}}\underbrace{00\cdots0}_{n-d \text{ 0's}}.$$

The trouble with this code is that there are very few codewords. We could only encode sets $\mathcal{A}$ of two different message blocks. We want ways of constructing richer codes, codes with more possible codewords.

In Section 10.4 we shall see how to use clever encoding methods $E : \mathcal{A} \to B^n$ to find richer (n, d)-codes. In Section 10.5 we shall see how to use block designs, and in particular their incidence matrices, to define very rich (n, d)-codes. In the next subsection, we obtain an upper bound on the size of an (n, d)-code.

In Example 10.3, we introduced the idea of a repetition code.

Example 10.5 A Repetition Code (Example 10.3 Revisited) As described in Example 10.3, a repetition code that repeats a block r times, a $k \to rk$ code, takes $E(a) = aa \cdots a$, where a is repeated r times. Using such a code, we can easily detect errors and correct errors. For example, consider the case where we have a triple repetition code. The set C of codewords consists of blocks of length $3k$ which have a block of length k repeated three times. To detect errors, we compare the ith digits in the three repetitions of the k-block. Thus, suppose that $k = 4$ and we receive 001101110110, which breaks into k-blocks as 0011/0111/0110. Then we know that there has been an error, since, for example, the second digits of the first two k-blocks differ. We can detect up to two transmission errors in this way. However, three errors might go undetected if they were all in the same digit of the original k-block. Of course, the minimum distance between two codewords aaa is $d = 3$, so this observation agrees with the result of Theorem 10.2. We could try to correct the error as follows. Note that the second digit of the k-blocks is 1 twice and 0 once. If we assume that errors are unlikely, we could use the majority rule and assume that the correct digit was 1. Using similar reasoning, we would decode 001101110110 as 0111. This error-correcting procedure can correct up to one error. However, two errors would be "corrected" incorrectly if they were both on the same digit. This observation again agrees with the results of Theorem 10.2, as we have a code with $d = 3$. If we want to correct more errors, we simply use more repetitions. For example, a five-repetitions code $E(a) = aaaaa$ has $d = 5$ and can detect up to four errors and correct up to two. Unfortunately, the repetition method of designing error-correcting codes is expensive in terms of both time and space. ■

Example 10.6 Parity Check Codes A simple way to detect a single error is to add one digit to a block, a digit that will always bring the number of 1's to an even number. The extra digit added is called a *parity check digit*. The corresponding code is a $k \to (k+1)$ code, and it lets $E(0011) = 00110$, $E(0010) = 00101$, and so on. We can represent E by

$$E(a_1 a_2 \cdots a_k) = a_1 a_2 \cdots a_k \sum_{i=1}^{k} a_i,$$

where $\sum_{i=1}^{k} a_i$ is interpreted as the summation modulo 2, as defined in Section 9.3.1. Throughout the rest of this chapter, addition will be interpreted this way. Thus, $1 + 1 = 0$, which is why E(0011) = 00110. In such a *parity check code* E, the minimum distance between two codewords is $d = 2$, so one error can be detected. No errors can be corrected. As we have pointed out before, such parity check codes were used on the earliest computers: UNIVAC, Whirlwind I, and the IBM 650. Large and small computers of the 1960s had memory parity check as an optional feature, and by the 1970s this was standard on a number of machines. ■

10.3.2 The Hamming Bound

Fix a given bit string s in an (n, d)-code C. Let the collection of bit strings of length n which have distance exactly t from string s be denoted $B_t(s)$. Then the number of bit strings in $B_t(s)$ is given by $\binom{n}{t}$, for we choose t positions out of n in which to change digits. Let the collection of bit strings of length n that have distance *at* *most* t from s be denoted by $B'_t(s)$. Thus, the number of bit strings in $B'_t(s)$ is given by

$$b = \binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{t}. \tag{10.4}$$

If $t = \lceil (d/2) - 1 \rceil$, then $B'_t(s) \cap B'_t(s^*) = \emptyset$ for every $s \neq s^*$ in C. Thus, with $t = \lceil (d/2) - 1 \rceil$, every bit string of length n is in at most one set $B'_t(s)$. Since there are 2^n bit strings of length n, we have

$$|C|\, b = \sum_{s \in C} |B'_t(s)| = |\cup_{s \in C} B'_t(s)| \leq 2^n.$$

Thus, we get the following theorem.

Theorem 10.3 (Hamming Bound) If C is an (n, d)-code and $t = \lceil (d/2) - 1 \rceil$, then

$$|C| \leq \frac{2^n}{\binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{t}}. \tag{10.5}$$

This result is due to Hamming [1950]. It is sometimes called the *sphere packing bound.*

To illustrate this theorem, let us take $n = d = 3$. Then $t = 1$. We find that any (n, d)-code has at most

$$\frac{2^3}{\binom{3}{0} + \binom{3}{1}} = 2$$

codewords. There is a $(3, 3)$-code of two codewords: Use the bit strings 111 and 000. We investigate an alternative upper bound on the size of (n, d)-codes in Section 10.5.3.

10.3.3 The Probability of Error

Recall that we are assuming that we have a binary symmetric channel: The probability of switching 1 to 0 is the same as the probability of switching 0 to 1, and this common probability p is the same at each digit, independent of any previous errors that may have occurred. Then we have the following theorem.

Theorem 10.4 In a binary symmetric channel, the probability that exactly r errors will be made in transmitting a bit string of length n is given by

$$\binom{n}{r} p^r (1 - p)^{n-r}.$$

Proof.[5] The student familiar with probability will recognize that we are in a situation of Bernoulli trials as described in Example 5.35. We seek the probability

[5] The proof should be omitted by the student who is not familiar with probability.

of r successes in n independent, repeated trials, if the probability of success on any one trial is p. The formula given in the theorem is the well-known formula for this probability (see Example 5.35). Q.E.D.

For instance, suppose that $p = .01$. Then in a 4-digit message, the probability of no errors is

$$(1 - .01)^4 = .960596,$$

the probability of one error is

$$\binom{4}{1}(.01)(1 - .01)^3 = .038812,$$

the probability of two errors is

$$\binom{4}{2}(.01)^2(1 - .01)^2 = .000588,$$

and the probability of more than two errors is

$$1 - .960596 - .038812 - .000588 = .000004.$$

It is useful to compare the two codes of Examples 10.5 and 10.6. Suppose that we wish to send a message of 1000 digits, with the probability of error equal to $p = .01$. If there is no encoding, the probability of making no errors is $(1 - .01)^{1000}$, which is approximately .000043. Suppose by way of contrast that we use the $k \to 3k$ triple repetition code of Example 10.5, with $k = 1$. Suppose that we want to send a digit a. We encode it as aaa. The probability of correctly sending the block aaa is $(.99)^3$, or approximately .970299. By Theorem 10.4, the probability of a single error is $\binom{3}{1}(.01)(.99)^2$, or approximately .029403. Thus, since we can correct single errors, the probability of correctly interpreting the message aaa and hence of correctly decoding the single digit a is $.970299 + .029403 = .999702$. There are 1000 digits in the original message, so the probability of decoding the whole message correctly is $(.999702)^{1000}$, or approximately .742268. This is much higher than .000043. Note that the greatly increased likelihood of error-free transmission is bought at a price: We need to send 3000 digits in order to receive 1000.

Let us compare the parity check code. Suppose that we break the 1000-digit message into blocks of length 10; there are 100 such in all. Each such block is encoded as an 11-digit block by adding a parity check digit. The probability of sending the 11-digit block without making any errors is $(.99)^{11}$, or approximately .895338. Also, by Theorem 10.4, the probability of sending the block with exactly one error is $\binom{11}{1}(.01)(.99)^{10}$, or approximately .099482. Now if a single error is made, we can detect it and ask for a retransmission. Thus, it is reasonable to assume that we can eliminate single errors. Hence, the probability of receiving the 11-digit block correctly is

$$.895338 + .099482 = .994820.$$

Now the original 1000-digit message has 100 blocks of 10 digits, so the probability of eventually decoding this entire message correctly is $.994820^{100}$, or approximately .594909. The probability of correct transmission is less than with a triple repetition code, but much greater than without a code. The price is much less than the triple repetition code: We need to send 1100 digits to receive 1000.

10.3.4 Consensus Decoding and Its Connection to Finding Patterns in Molecular Sequences[6]

Consider the situation of transmission over a "noisy" binary symmetric channel where there is a good chance of errors. We might request retransmission a number of times, as in repetition codes, receiving a collection of strings in B^n, say strings $x_1, x_2, \ldots, x_p$. Some of them are in the set C of codewords, others are not. We would like to determine which codeword in C was intended. The problem is to find a word in C that is in some sense a "consensus" of the words actually received. The idea of "majority rule decoding" is a special case of this. This problem of consensus is widely encountered in applications including voting and "group decisionmaking," selecting the closest match from a database of molecular sequences, and "metasearching" on the Internet (comparing the results of several search engines). The mathematics of the consensus problem has been studied widely in the literature of "mathematical social science." For an introduction to this topic, see Bock [1988], Johnson [1998], and Roberts [1976].

One consensus procedure widely in use is the *median procedure*: Find the codeword w in C for which $\sum_{i=1}^{p} d(w, x_i)$ is minimized, where d is the Hamming distance. An alternative, the *mean procedure*, is to find the codeword w in C for which $\sum_{i=1}^{p} d(w, x_i)^2$ is minimized. Consider, for example, the code C consisting of the codewords in (10.3). Suppose that we ask for three transmissions of our message and receive the three words:

$$x_1 = 100000, \quad x_2 = 110000, \quad x_3 = 111000.$$

We calculate

$$\sum_{i=1}^{3} d(000000, x_i) = 1 + 2 + 3 = 6, \qquad \sum_{i=1}^{3} d(010101, x_i) = 4 + 3 + 4 = 11,$$
$$\sum_{i=1}^{3} d(101010, x_i) = 2 + 3 + 2 = 7, \qquad \sum_{i=1}^{3} d(111111, x_i) = 5 + 4 + 3 = 12.$$

Since 000000 has the smallest sum, it is chosen by the median procedure. We call

[6] This subsection may be omitted.

it a *median*. Similarly,

$$\sum_{i=1}^{3} d\left(000000, x_i\right)^2 = 1 + 4 + 9 = 14, \qquad \sum_{i=1}^{3} d\left(010101, x_i\right)^2 = 16 + 9 + 16 = 41,$$
$$\sum_{i=1}^{3} d\left(101010, x_i\right)^2 = 4 + 9 + 4 = 17, \qquad \sum_{i=1}^{3} d\left(111111, x_i\right)^2 = 25 + 16 + 9 = 50,$$

so the mean procedure also chooses codeword 000000. We call it a *mean*.

The median procedure and mean procedure do not always agree and, moreover, they do not always give a unique decoding—there can be ambiguity. Consider, for example, the code

$$C = \{111111, 001110\}.$$

Suppose that words $x_1 = 001111$, $x_2 = 101011$ are received. Then

$$\sum_{i=1}^{2} d\left(111111, x_i\right) = 2 + 2 = 4, \qquad \sum_{i=1}^{2} d\left(001110, x_i\right) = 1 + 3 = 4,$$
$$\sum_{i=1}^{2} d\left(111111, x_i\right)^2 = 4 + 4 = 8, \qquad \sum_{i=1}^{2} d\left(001110, x_i\right)^2 = 1 + 9 = 10.$$

The median procedure leads to ambiguity, giving both codewords in C as medians. However, the mean procedure leads to the unique solution 111111 as the mean.

In many problems of the social and biological sciences, data are presented as a sequence or "word" from some alphabet Σ.[7] Given a set of sequences, we seek a *pattern* or *feature* that appears widely, and we think of this as a *consensus sequence* or set of sequences. A pattern is often thought of as a consecutive subsequence of short, fixed length. In Example 6.11 we noted that the discovery of such patterns or consensus sequences for molecular sequences has already led to such important discoveries as the fact that the sequence for platelet-derived factor, which causes growth in the body, is 87 percent identical to the sequence for *v-sis*, a cancer-causing gene. This led to the discovery that *v-sis* works by stimulating growth. To measure how closely a pattern fits into a sequence, we have to measure the distance between words of different lengths. If b is longer than a, then $d(a, b)$ could be the smallest number of mismatches in all possible *alignments* of a as a consecutive subsequence of b. We call this the *best-mismatch distance*. If the sequences are bit strings, this is

[7] The following discussion of pattern recognition in molecular sequences follows Mirkin and Roberts [1993]. It is an example from the field known as *bioconsensus*, which involves the use of (mostly) social-science-based methods in biological applications. In typical problems of bioconsensus, several alternatives (such as alternative phylogenetic trees or alternative molecular sequences) are produced using different methods or models, and then one needs to find a consensus solution. Day and McMorris [1992] surveyed the use of consensus methods in molecular biology and Day [2002] and Day and McMorris [1993] gave many references; there are literally hundreds of papers in the field of "alignment and consensus." For example, Kannan [1995] surveyed consensus methods for phylogenetic tree reconstruction. For more information, see Day and McMorris [2003] and Janowitz, *et al.* [2003].

the minimum Hamming distance between a and all consecutive subsequences of b of the same length as a. Consider, for example, the case where $a = 0011$, $b = 111010$. Then the possible alignments are:

```
111010    1111010    111010
0011       0011        0011
```

The best-mismatch distance is 2, which is achieved in the third alignment. An alternative way to measure $d(a, b)$ is to count the smallest number of mismatches between sequences obtained from a and b by inserting gaps in appropriate places (where a mismatch between a letter of Σ and a gap is counted as an ordinary mismatch). We won't use this alternative measure of distance, although it is used widely in molecular biology.

Waterman [1989] and others (Waterman, Arratia, and Galas [1984], Galas, Eggert, and Waterman [1985], and Waterman [1995]) study the situation where Σ is a finite alphabet, M is a fixed finite number (the pattern length), $\chi = \{x_1, x_2, \ldots, x_n\}$ is a set of words (sequences) of length L from Σ, where $L \geq M$, and we seek a set $F(\chi) = F(x_1, x_2, \ldots, x_n)$ of consensus words of length M from Σ. The set of consensus words could be any words in Σ^M or words from a fixed subset of allowable pattern words. For now, we shall disregard the latter possibility. Here is a small piece of data from Waterman [1989], in which he looks at 59 bacterial promoter sequences:

RRNABP1	ACTCCCTATAATGCGCCA
TNAA	GAGTGTAATAATGTAGCC
UVRBP2	TTATCCAGTATAATTTGT
SFC	AAGCGGTGTTATAATGCC.

Notice that if we are looking for patterns of length 4, each sequence has the pattern TAAT. However, suppose that we add another sequence:

M1RNA AACCCTCTATACTGCGCG.

The pattern TAAT does not appear here. However, it almost appears since the word TACT appears, and this has only one mismatch from the pattern TAAT. So, in some sense, the pattern TAAT is a pattern that is a good consensus pattern. We now make this idea more precise.

In practice, the problem is a bit more complicated than we have described it. We have long sequences and we consider "windows" of length L beginning at a fixed position, say the jth. Thus, we consider words of length L in a long sequence, beginning at the jth position. For each possible pattern of length M, we ask how closely it can be matched in each of the sequences in a window of length L starting at the jth position. To formalize this, let Σ be a finite alphabet of size at least 2 and χ be a finite collection of words of length L on Σ. Let $F(\chi)$ be the set of words of length $M \geq 2$ that are our *consensus patterns*. Let $\chi = \{x_1, x_2, \ldots, x_p\}$. One way to define $F(\chi)$ is as follows. Let $d(a, b)$ be the best-mismatch distance. Consider nonnegative parameters λ_d that are monotone decreasing with d, and let

$F(x_1, x_2, \ldots, x_p)$ be all those words w of length M from Σ^M (all M-tuples from Σ) that maximize

$$s_\chi(w) = \sum_{i=1}^{p} \lambda_{d(w,x_i)}.$$

We call such an F a *Waterman consensus.* In particular, Waterman and others use the parameters $\lambda_d = (M-d)/M$.

As an example, we note that a frequently used alphabet is the purine/pyrimidine alphabet {R, Y}, where R = A or G and Y = C or T. For simplicity, it is easier to use the digits 0, 1 rather than the letters R, Y. Thus, let $\Sigma = \{0, 1\}$, $M = 2$, and consider $F(x_1, x_2)$, where $x_1 = 111010$, $x_2 = 111111$. The possible pattern words are 00, 01, 10, 11. We have

$$\begin{array}{ll} d(00, x_1) = 1, & d(00, x_2) = 2, \\ d(01, x_1) = 0, & d(01, x_2) = 1, \\ d(10, x_1) = 0, & d(10, x_2) = 1, \\ d(11, x_1) = 0, & d(11, x_2) = 0. \end{array}$$

Thus,

$$\begin{aligned} s_\chi(00) &= \sum_{i=1}^{2} \lambda_{d(00,x_i)} = \lambda_1 + \lambda_2, \\ s_\chi(01) &= \sum_{i=1}^{2} \lambda_{d(01,x_i)} = \lambda_0 + \lambda_1, \\ s_\chi(10) &= \sum_{i=1}^{2} \lambda_{d(10,x_i)} = \lambda_0 + \lambda_1, \\ s_\chi(11) &= \sum_{i=1}^{2} \lambda_{d(11,x_i)} = \lambda_0 + \lambda_0. \end{aligned}$$

As long as $\lambda_0 > \lambda_1 > \lambda_2$, it follows that 11 is the consensus pattern, according to Waterman's consensus.

To give another example, let $\Sigma = \{0, 1\}$, $M = 3$, and consider $F(x_1, x_2, x_3)$ where $x_1 = 000000$, $x_2 = 100000$, $x_3 = 111110$. The possible pattern words are 000, 001, 010, 011, 100, 101, 110, 111. We have

$$s_\chi(000) = \lambda_2 + 2\lambda_0, \quad s_\chi(001) = \lambda_2 + 2\lambda_1, \quad s_\chi(100) = 2\lambda_1 + \lambda_0, \text{ etc.}$$

Now, $\lambda_0 > \lambda_1 > \lambda_2$ implies that $s_\chi(000) > s_\chi(001)$. Similarly, one can show that s_χ is maximized by $s_\chi(000)$ or $s_\chi(100)$. Monotonicity doesn't say which of these is greater.

An alternative consensus procedure is to use a variant of the median procedure that gives all words w of length M that minimize

$$\sigma_\chi(w) = \sum_{i=1}^{p} d(w, x_i);$$

or we can use a variant of the mean procedure that gives all words w of length M that minimize

$$\tau_\chi(w) = \sum_{i=1}^{p} d(w, x_i)^2.$$

For instance, suppose that $\Sigma = \{0, 1\}$, $M = 2$, $\chi = \{x_1, x_2, x_3, x_4\}$, $x_1 = 1111$, $x_2 = 0000$, $x_3 = 1000$, and $x_4 = 0001$. Then the possible pattern words are 00, 01, 10, 11. We have

$$\sum_{i=1}^{4} d(00, x_i) = 2, \quad \sum_{i=1}^{4} d(01, x_i) = 3,$$
$$\sum_{i=1}^{4} d(10, x_i) = 3, \quad \sum_{i=1}^{4} d(11, x_i) = 4.$$

Thus, 00 is the median. However,

$$\sum_{i=1}^{4} d(00, x_i)^2 = 4, \quad \sum_{i=1}^{4} d(01, x_i)^2 = 3,$$
$$\sum_{i=1}^{4} d(10, x_i)^2 = 3, \quad \sum_{i=1}^{4} d(11, x_i)^2 = 6,$$

so the mean procedure leads to the two words 01 and 10, neither of which is a median.

Let us now consider the Waterman consensus with the special case of parameter $\lambda_d = (M - d)/M$ that is generally used in practice. Recall that

$$\sigma_\chi(w) = \sum_{i=1}^{p} d(w, x_i) \text{ and } s_\chi(w) = \sum_{i=1}^{p} \lambda_{d(w, x_i)} = p - \frac{1}{M} \sum_{i=1}^{p} d(w, x_i).$$

Thus, for fixed $M \geq 2$, Σ of size at least 2, and any size set χ of x_i's of length $L \geq M$, for all words w, w' of length M:

$$\sigma_\chi(w) \geq \sigma_\chi(w') \leftrightarrow s_\chi(w) \leq s_\chi(w').$$

It follows that for fixed $M \geq 2$, Σ of size at least 2, and any size set χ of x_i's of length $L \geq M$, there is a choice of the parameter λ_d so that the Waterman consensus is the same as the median. (This also holds for $M = 1$ or $|\Sigma| = 1$, but these are uninteresting cases.)

Similarly, one can show (see Exercises 32 and 33) that for fixed $M \geq 2$, Σ of size at least 2, and any size set χ of x_i's of length $L \geq M$, there is a choice of parameter λ_d so that for all words w, w' of length M:

$$\tau_\chi(w) \geq \tau_\chi(w') \leftrightarrow s_\chi(w) \leq s_\chi(w').$$

For this choice of λ_d, a word is a Waterman consensus if and only if it is a mean. It is surprising that the widely used Waterman consensus can actually be the mean or median in disguise.

EXERCISES FOR SECTION 10.3

1. In each of the following cases, find the Hamming distance between the two codewords x and y.

   (a) $x = 1010001, y = 0101010$ (b) $x = 11110011000, y = 11001001001$

   (c) $x = 10011001, y = 10111101$ (d) $x = 111010111010, y = 101110111011$

2. In the parity check code, find $E(\mathbf{a})$ if $\mathbf{a}$ is:

   (a) 111111 (b) 1001011

   (c) 001001001 (d) 01010110111

3. For each of the following codes C, find the number of errors that could be detected and the number that could be corrected using the nearest-neighbor rule.

   (a) $C = \{00000000, 11111110, 10101000, 01010100\}$

   (b) $C = \{000000000, 111111111, 111110000, 000001111\}$

   (c) $C = \{000000000, 111111111, 111100000, 000011111, 101010101, 010101010\}$

4. For each of the following binary codes C and received strings $x_1, x_2, \ldots, x_p$, find all medians and means.

   (a) C: $\{00000000, 11111110, 10101000, 01010100\}$
   x_i's: 00011000, 01000010, 01100110

   (b) C: $\{000000000, 111111111, 111110000, 000001111\}$
   x_i's: 000101000, 010010010, 011010110

   (c) C: $\{000000000, 111111111, 111100000, 000011111, 101010101, 010101010\}$
   x_i's: 100011001, 011000110, 000110101, 110000011, 010001010

5. Find the best-mismatch distance $d(a, b)$ for the following:

   (a) $a = 01$, $b = 1010$ (b) $a = 001$, $b = 101010101010101$

   (c) $a = 00$, $b = 1101011011$ (d) $a = 101$, $b = 100110$

6. Show that there is a set C of codewords of length n and a set of messages of length n so that the median procedure gives a unique solution and the mean procedure does not.

7. A *q-ary block n-code* (*q-ary code*) is a collection of strings of length n chosen from the alphabet $\{0, 1, \ldots, q-1\}$. The *Hamming distance* between two strings from this alphabet is again defined to be the number of digits on which they differ. For instance, if $q = 4$, then $d(0123, 1111) = 3$.

   (a) Find the Hamming distance between the following strings x and y.

      i. $x = 0226215, y = 2026125$

      ii. $x = 000111222333, y = 001110223332$

      iii. $x = 01010101010, y = 01020304050$

   (b) If the minimum distance between two codewords in a q-ary block n-code is d, how many errors can the code detect?

   (c) How many errors can the code correct under the nearest-neighbor rule?

8. Find an upper bound on the number of codewords in a code where each codeword has length 10 and the minimum distance between any two codewords is 7.
9. Repeat Exercise 8 for length 11 and minimum distance 6.
10. Find an upper bound on the number of codewords in a code C whose codewords have length 7 and which corrects up to 2 errors.
11. If codewords are bit strings of length 10, we have a binary symmetric channel, and the probability of error is .1, find the probability that if a codeword is sent there will be:

    (a) No errors (b) Exactly one error

    (c) Exactly two errors (d) More than two errors
12. Repeat Exercise 11 if the length of codewords is 6 and the probability of error is .001.
13. Suppose we assume that under transmission of a q-ary codeword (Exercise 7), the probability p of error is the same at each digit, independent of any previous errors. Moreover, if there is an error at the ith digit, the digit is equally likely to be changed into any one of the remaining $q - 1$ symbols. We then refer to a *q-ary symmetric channel.*

    (a) In a q-ary symmetric channel, what is the probability of exactly t errors if a string of length n is sent?

    (b) If $q = 3$, what is the probability of receiving 22222 if 11111 is sent?

    (c) If $q = 4$, what is the probability of receiving 2013 if 1111 is sent?
14. Suppose that in a binary symmetric channel, codewords have length $n = 3$ and that the probability of error is .1. If we want a code which, with probability $\lambda \geq .95$, will correct a received message if it is in error, what must be the minimum distance between two codewords?
15. Repeat Exercise 14 if:

    (a) $n = 5$ and $\lambda \geq .50$ (b) $n = 3$ and $\lambda \geq .90$
16. Suppose that in a binary triple repetition code, the value of k is 5 and the set $\mathcal{A} = A^5$.

    (a) How many errors can be detected? (b) How many errors can be corrected?

    (c) Suppose that instead, $k = 6$. How do your answers to parts (a) and (b) change?
17. Exercises 17–22 deal with q-ary codes (Exercise 7). Find a bound on the number of codewords in a q-ary code that is analogous to Theorem 10.3.
18. Suppose that a code is built up of strings of symbols using 0's, 1's, and 2's, and that S is a set of codewords with the following properties:

    (i) Each word in S has length n.

    (ii) Each word in S has a 2 in the first and last places and nowhere else.

    (iii) $d(a, b) = d$ for each pair of words a and b in S, where $d(a, b)$ is the Hamming distance.

    Let T be defined from S by changing 0 to 1 and 1 to 0 but leaving 2 unchanged. For example, the word 2102 becomes the word 2012.

(a) What is the distance between two words of T?

(b) What is the distance between a word of S and a word of T?

(c) Suppose that R consists of the words of S plus the words of T. How many errors can R detect? Correct?

19. Let A be a $q \times q$ Latin square on the set of symbols $\{0, 1, 2, \ldots, q-1\}$. Consider the q-ary code C consisting of all strings of length 3 of the form i, j, a_{ij}.

(a) If 5 is replaced by 0, find C corresponding to the Latin square of Table 1.4.

(b) Given the code C for an arbitrary $q \times q$ Latin square, how many errors can the code detect?

(c) How many errors can it correct?

20. Suppose that we have an orthogonal family of Latin squares of order q, $A^{(1)}$, $A^{(2)}$, $\ldots$, $A^{(r)}$, each using the symbols $0, 1, \ldots, q-1$. Consider the q-ary code S consisting of all strings of length $r+2$ of the form

$$i, j, a_{ij}^{(1)}, a_{ij}^{(2)}, \ldots, a_{ij}^{(r)}.$$

(a) Find S corresponding to the orthogonal family of Table 9.23.

(b) For arbitrary q, how many errors can the code S detect?

(c) How many errors can it correct?

21. Consider codewords of length 4, with each digit chosen from the alphabet $\{0,\ 1,\ 2,\ 3,\ 4,\ 5\}$. Show that there cannot be a set of 36 codewords each one of which has Hamming distance at least 3 from each other one.

22. Let A be a Latin square of order q on the set of symbols $\{0, 1, \ldots, q-1\}$ which is horizontally complete in the sense of Section 9.3, Exercise 5. Consider the q-ary code T consisting of all strings of length 4 of the form

$$i, j, a_{ij}, a_{i\ j+1}$$

for $i = 0, 1, \ldots, q-1$, $j = 0, 1, \ldots, q-2$.

(a) Find T corresponding to the Latin square constructed in Exercise 5, Section 9.3, in the case $n = 4$.

(b) For arbitrary q, how many errors can the code T detect?

(c) How many errors can it correct?

23. Suppose that we have a message of 10 digits and that $p = .1$ is the probability of an error.

(a) What is the probability of no errors in transmission if we use no code?

(b) What is the probability of correctly transmitting the message if we use the $k \to 3k$ triple repetition code with $k = 1$?

(c) What is this probability if we use the parity check code with blocks of size 2?

24. Suppose that we have a message of 1000 digits and that $p = .01$ is the probability of error. Suppose that we use the $k \to 5k$ five repetitions code with $k = 1$. What is the probability of transmitting the message correctly? (*Hint:* How many errors can be corrected?)
25. Suppose that majority rule decoding is used with a $k \to 3k$ repetition code, $k = 5$.
    (a) What is the "corrected" message if we receive *auvwbcuvzbcuvwd*?
    (b) If a given digit in a message is changed with probability $p = .1$, independent of its position in the message, what is the probability that a message of length $3k$ will be received without any errors?
    (c) Continuing with part (b), what is the probability of decoding a message of length $k = 5$ correctly if majority rule decoding is used?
    (d) Continuing with part (b), what number of repetitions would be needed to achieve a probability of at least .9999 of decoding a message of length $k = 5$ correctly if majority rule decoding is used?
26. An (n, d)-code C is called a *perfect t-error-correcting code* if $t = \lceil (d/2) - 1 \rceil$ and inequality (10.5) is an equality. Show that such a C *never* corrects more than t errors.
27. Show that the $1 \to 2t + 1$ repetition codes (Example 10.5) are perfect t-error-correcting codes.
28. It turns out (Tietäväinen [1973], van Lint [1971]) that there is only one perfect t-error-correcting code other than the repetition codes and the Hamming codes (to be introduced in Section 10.4.3). This is a $12 \to 23$ perfect 3-error correcting code due to Golay [1949]. How many codewords does this code have?
29. Suppose that C is a q-ary block n-code of minimum distance d. Generalize the Hamming bound (Theorem 10.3) to find an upper bound on $|C|$.
30. Let $\Sigma = \{0, 1\}$, $L = 6$, $M = 2$, and $\chi = \{110100, 010101\}$.
    (a) Calculate $s_\chi(w)$ for all possible words w in Σ^M.
    (b) Find the Waterman consensus if $\lambda_d = (M - d)/M$.
    (c) Calculate $\sigma_\chi(w)$ for all w in Σ^M.
    (d) Calculate $\tau_\chi(w)$ for all w in Σ^M.
    (e) Find all medians.
    (f) Find all means.
31. Repeat Exercise 30 if $\Sigma = \{0, 1\}$, $L = 6$, $M = 3$, and $\chi = \{000111, 011001, 100101\}$.
32. (Mirkin and Roberts [1993]) Show that if $\lambda_d = Ad + B$, $A < 0$, the Waterman consensus is the same as that chosen by the median procedure.
33. (Mirkin and Roberts [1993]) Show that if $\lambda_d = Ad^2 + B$, $A < 0$, the Waterman consensus is the same as that chosen by the mean procedure.
34. (Mirkin and Roberts [1993])
    (a) Suppose that $M \geq 2$, $|\Sigma| \geq 2$, $\lambda_1 < \lambda_0$, and $\lambda_d = Ad + B$, $A < 0$. Show that $\sigma_\chi(w) \geq \sigma_\chi(w') \leftrightarrow s_\chi(w) \leq s_\chi(w')$ for all words w, w' from Σ^M and all finite sets χ of *distinct* words from Σ^L, $L \geq M$.

(b) Interpret the conclusion.

35. (Mirkin and Roberts [1993])

(a) Suppose that $M \geq 2$, $|\Sigma| \geq 2$, $\lambda_1 < \lambda_0$, and $\lambda_d = Ad^2 + B$, $A < 0$. Show that $\tau_\chi(w) \geq \tau_\chi(w') \leftrightarrow s_\chi(w) \leq s_\chi(w')$ for all words w, w' from Σ^M and all finite sets χ of *distinct* words from Σ^L, $L \geq M$.

(b) Interpret the conclusion.

10.4 LINEAR CODES[8]

10.4.1 Generator Matrices

In this section we shall see how to use the encoding step to build error-detecting and error-correcting codes. Moreover, the encoding we present will have associated with it very efficient encoding and decoding procedures. The approach we describe was greatly influenced by the work of R. W. Hamming and D. Slepian in the 1950s. See, for example, the papers of Hamming [1950] and Slepian [1956a,b, 1960].

We first generalize Example 10.6. We can consider a $k \to n$ code E as adding $n-k$ *parity check digits* to a block of length k. In general, a *message block* $a_1a_2\cdots a_k$ is encoded as $x_1x_2\cdots x_n$, where $x_1 = a_1, x_2 = a_2, \ldots, x_k = a_k$ and the parity check digits are determined from the k *message digits* a_1, a_2, …, a_k. The easiest way to obtain such an encoding is to generalize the matrix encoding method of Example 10.2. We let $\mathbf{M}$ be a $k \times n$ matrix, called the *generator matrix*, and we define $E(\mathbf{a})$ to be $\mathbf{aM}$.

Example 10.7 Suppose that $k = 3$, $n = 6$, and

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}.$$

Then the message word 110 is encoded as $110\mathbf{M} = 110101$. The fifth digit is 0 because $1 + 1 = 0$: Recall that we are using addition modulo 2. Note that $\mathbf{M}$ begins with a 3×3 identity matrix. Since we want $a_i = x_i$ for $i = 1, 2, \ldots, k$, every generator matrix will begin with the $k \times k$ identity matrix $\mathbf{I}_k$. ■

To give some other examples, the parity check codes of Example 10.6 are defined by taking

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 1 \\ 0 & 1 & 0 & \cdots & 0 & 1 \\ 0 & 0 & 1 & \cdots & 0 & 1 \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ 0 & 0 & 0 & \cdots & 1 & 1 \end{bmatrix}. \tag{10.6}$$

[8]This section may be omitted if the reader wants to go directly to Section 10.5.

We add a column of all 1's to the $k \times k$ identity matrix $\mathbf{I}_k$. The triple repetition code of Example 10.5 is obtained by taking $\mathbf{M}$ to be three copies of the $k \times k$ identity matrix.

If a code is defined by a generator matrix, decoding is trivial (when the transmission is correct). We simply decode $x_1 x_2 \cdots x_n$ as $x_1 x_2 \cdots x_k$; that is, we drop the $n - k$ parity check digits.

In general, codes definable by such generator matrices are called *linear codes*. This is because if $\mathbf{x} = x_1 x_2 \cdots x_n$ and $\mathbf{y} = y_1 y_2 \cdots y_n$ are codewords, so is the word $\mathbf{x} + \mathbf{y}$ whose ith digit is $x_i + y_i$, where addition is modulo 2. To see this, simply observe that if $\mathbf{aM} = \mathbf{x}$ and $\mathbf{bM} = \mathbf{y}$, then $(\mathbf{a} + \mathbf{b})\mathbf{M} = \mathbf{x} + \mathbf{y}$.[9]

It follows that $\mathbf{0} = 00 \cdots 0$ is a codeword of every (nonempty) linear code. For if $\mathbf{x}$ is any codeword, then since addition is modulo 2, $\mathbf{x} + \mathbf{x} = \mathbf{0}$. Indeed, it turns out that the set of all codewords under the operation $+$ defines a group, to use the language of Section 8.2 (see Exercise 26). Thus, linear codes are sometimes called *binary group codes.*

We next note that in a linear code, the minimum distance between two codewords is easy to find. Let the *Hamming weight* of a bit string $\mathbf{x}$, denoted $wt(\mathbf{x})$, be the number of nonzero digits of $\mathbf{x}$.

Theorem 10.5 In a linear code C, the minimum distance d between two codewords is equal to the minimum Hamming weight w of a codeword other than the codeword $\mathbf{0}$.

Proof. Note that if $d(\mathbf{x}, \mathbf{y})$ is Hamming distance, then since C is linear, $\mathbf{x} + \mathbf{y}$ is in C, so

$$d(\mathbf{x}, \mathbf{y}) = wt(\mathbf{x} + \mathbf{y}).$$

This conclusion uses the fact that addition is modulo 2. Suppose that the minimum distance d is given by $d(\mathbf{x}, \mathbf{y})$ for a particular $\mathbf{x}$, $\mathbf{y}$ in C. Then

$$d = d(\mathbf{x}, \mathbf{y}) = wt(\mathbf{x} + \mathbf{y}) \geq w.$$

Next, suppose that $\mathbf{u} \neq \mathbf{0}$ in C has minimum weight w. Then since $\mathbf{0} = \mathbf{u} + \mathbf{u}$ is in C, we have

$$w = wt(\mathbf{u}) = wt(\mathbf{u} + \mathbf{0}) = d(\mathbf{u}, \mathbf{0}) \geq d.$$ Q.E.D.

It follows by Theorem 10.5 that in the code of Example 10.6, the minimum distance between two codewords is 2, since the minimum weight of a codeword other than $\mathbf{0}$ is 2. This weight is attained, for example, in the string 11000. The minimum distance is attained, for example, as

$$d(11000, 00000).$$

[9] We use the assumption, which we shall make throughout this section, that $\mathcal{A} = A^k$.

10.4.2 Error Correction Using Linear Codes

If **M** is a generator matrix for a linear code, it can be represented schematically as $[\mathbf{I}_k\mathbf{G}]$, where **G** is a $k \times (n-k)$ matrix. Let $\mathbf{G}^T$ be the transpose of matrix **G** and let $\mathbf{H} = [\mathbf{G}^T\mathbf{I}_{n-k}]$ be an $(n-k) \times n$ matrix, called the *parity check matrix*. The matrix $\mathbf{G}^T$ in Example 10.6 is a row vector of k 1's, and the matrix **H** is a row vector of $(k+1)$ 1's. In Example 10.5, with three repetitions and message blocks of length $k = 4$, we have

$$\mathbf{G}^T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

and

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{10.7}$$

In Example 10.7,

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}. \tag{10.8}$$

Note that a linear code can be defined by giving either the matrix **G** from the generator matrix **M** or the parity check matrix **H**. For we can derive **H** from **G** and **G** from **H**. We shall see that parity check matrices will provide a way to detect and correct errors. The basic ideas of this approach are due to Slepian [1956a,b, 1960].

First, we note that the parity check matrix can be used to identify codewords.

Theorem 10.6 In a linear code, a block $\mathbf{a} = a_1a_2\cdots a_k$ is encoded as $\mathbf{x} = x_1x_2\cdots x_n$ if and only if $a_i = x_i$ for $i \leq k$ and $\mathbf{H}\mathbf{x}^T = \mathbf{0}$.

Proof.[10] By definition of the encoding, if block $\mathbf{a} = a_1a_2\cdots a_k$ is encoded as $\mathbf{x} = x_1x_2\cdots x_n$, then

$$(a_1a_2\cdots a_k)[\mathbf{I}_k\mathbf{G}] = (x_1x_2\cdots x_n).$$

[10]The proof may be omitted.

Now, clearly, $a_i = x_i, i \leq k$. Also,

$$\begin{aligned}
\mathbf{Hx}^T &= \mathbf{H}(\mathbf{a}[\mathbf{I}_k\mathbf{G}])^T \\
&= \mathbf{H}[\mathbf{I}_k\mathbf{G}]^T\mathbf{a}^T \\
&= [\mathbf{G}^T\mathbf{I}_{n-k}]\begin{bmatrix} \mathbf{I}_k \\ \mathbf{G}^T \end{bmatrix}\mathbf{a}^T \\
&= (\mathbf{G}^T + \mathbf{G}^T)\mathbf{a}^T \\
&= \mathbf{0a}^T \\
&= \mathbf{0},
\end{aligned}$$

where $\mathbf{G}^T + \mathbf{G}^T = \mathbf{0}$ since addition is modulo 2.

Conversely, suppose that $a_i = x_i, i \leq k$, and $\mathbf{Hx}^T = \mathbf{0}$. Now suppose that $\mathbf{a}$ is encoded as $\mathbf{y} = y_1y_2\cdots y_n$. Then $a_i = y_i, i \leq k$, so $x_i = y_i, i \leq k$. Also, by the first part of the proof, $\mathbf{Hy}^T = \mathbf{0}$. It follows that $x = y$. To see why, note that the equations $\mathbf{Hx}^T = \mathbf{0}$ and $\mathbf{Hy}^T = \mathbf{0}$ each give rise to a system of equations. The jth equation in the first case involves at most the variables $x_1, x_2, \ldots, x_k$ and x_{k+j}. Thus, since $x_1, x_2, \ldots, x_k$ are given by $a_1, a_2, \ldots, a_k$, the jth equation defines x_{k+j} uniquely in terms of the k message digits $a_1, a_2, \ldots, a_k$. The same is true of y_{k+j}. Thus, $x_{k+j} = y_{k+j}$. Q.E.D.

Corollary 10.6.1 A bit string $\mathbf{x} = x_1x_2\cdots x_n$ is a codeword if and only if $\mathbf{Hx}^T = \mathbf{0}$.

Corollary 10.6.2 There is a unique bit string $\mathbf{x}$ so that $x_i = a_i, i \leq k$, and $\mathbf{Hx}^T = \mathbf{0}$. This bit string $\mathbf{x}$ is the encoding of $a_1a_2\cdots a_k$. An expression for the entry x_{k+j} of $\mathbf{x}$ in terms of $a_1, a_2, \ldots, a_k$ is obtained by multiplying the jth row of $\mathbf{H}$ by $\mathbf{x}^T$.

Proof. This is a corollary of the proof. Q.E.D.

To illustrate these results, note that in the parity check codes, $H = (11\cdots1)$ with $n = k + 1$ 1's. Then $\mathbf{Hx}^T = \mathbf{0}$ if and only if

$$x_1 + x_2 + \cdots + x_{k+1} = 0. \tag{10.9}$$

But since addition is modulo 2, (10.9) is exactly equivalent to the condition that

$$x_{k+1} = x_1 + x_2 + \cdots + x_k,$$

which defines codewords. This equation is called the *parity check equation.* Similarly, in the triple repetition codes with $k = 4$, we see from (10.7) that $\mathbf{Hx}^T = \mathbf{0}$ if and only if

$$\begin{array}{llll}
x_1 + x_5 = 0, & x_2 + x_6 = 0, & x_3 + x_7 = 0, & x_4 + x_8 = 0, \\
x_1 + x_9 = 0, & x_2 + x_{10} = 0, & x_3 + x_{11} = 0, & x_4 + x_{12} = 0.
\end{array} \tag{10.10}$$

Since addition is modulo 2, Equations (10.10) are equivalent to

$$\begin{array}{llll}
x_1 = x_5, & x_2 = x_6, & x_3 = x_7, & x_4 = x_8, \\
x_1 = x_9, & x_2 = x_{10}, & x_3 = x_{11}, & x_4 = x_{12},
\end{array}$$

which define a codeword. These equations are called the *parity check equations.* Finally, in Example 10.7, by (10.8), $\mathbf{Hx}^T = \mathbf{0}$ if and only if

$$\begin{aligned} x_1 + x_3 + x_4 &= 0, \\ x_1 + x_2 + x_5 &= 0, \\ x_2 + x_3 + x_6 &= 0. \end{aligned}$$

Using the fact that addition is modulo 2, these equations are equivalent to the following parity check equations, which determine the parity check digits x_{k+j} in terms of the message digits $x_i, i \leq k$:

$$\begin{aligned} x_4 &= x_1 + x_3, \\ x_5 &= x_1 + x_2, \\ x_6 &= x_2 + x_3. \end{aligned}$$

Thus, as we have seen before, 110 is encoded as 110101, since $x_4 = 1 + 0 = 1, x_5 = 1 + 1 = 0$, and $x_6 = 1 + 0 = 1$.

In general, by Corollary 10.6.2, the equations $\mathbf{Hx}^T = \mathbf{0}$ define x_{k+j} in terms of $a_1, a_2, \ldots, a_k$. The equations giving x_{k+j} in these terms are the *parity check equations.*

Theorem 10.6 and Corollary 10.6.1 provide a way to detect errors. We simply note that if $\mathbf{x}$ is received and if $\mathbf{Hx}^T \neq \mathbf{0}$, then an error occurred. The parity check matrix allows us not only to detect errors, but to correct them, as the next theorem shows.

Theorem 10.7 Suppose that the columns of the parity check matrix $\mathbf{H}$ are all nonzero and all distinct. Suppose that a codeword $\mathbf{y}$ is transmitted and a word $\mathbf{x}$ is received. If $\mathbf{x}$ differs from $\mathbf{y}$ only on the ith digit, then $\mathbf{Hx}^T$ is the ith column of $\mathbf{H}$.

Proof. Note that by Corollary 10.6.1, $\mathbf{Hy}^T = \mathbf{0}$. Now $\mathbf{x}$ can be written as $\mathbf{y}+\mathbf{e}$, where $\mathbf{e}$ is an error string, a bit string with 1's in the digits that differ from $\mathbf{y}$. (Recall that addition is modulo 2.) We conclude that

$$\mathbf{Hx}^T = \mathbf{H}(\mathbf{y}+\mathbf{e})^T = \mathbf{H}(\mathbf{y}^T + \mathbf{e}^T) = \mathbf{Hy}^T + \mathbf{He}^T = \mathbf{He}^T.$$

Now if $\mathbf{e}$ is the vector with all 0's except a 1 in the ith digit, then $\mathbf{He}^T$ is the ith column of $\mathbf{H}$. Q.E.D.

To illustrate this result, suppose that we have a triple repetition code and $k = 4$. Then the parity check matrix $\mathbf{H}$ is given by (10.7). Suppose that a codeword $\mathbf{y} = 110111011101$ is sent and a codeword $\mathbf{x} = 110011011101$, which differs on the fourth digit, is received. Then note that

$$\mathbf{H}\mathbf{x}^T = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix},$$

which is the fourth column of **H**.

In general, error correction using the parity check matrix **H** will proceed as follows, assuming that **H** has distinct columns and all are nonzero. Having received a block **x**, compute $\mathbf{H}\mathbf{x}^T$. If this is **0**, and errors in transmission are unlikely, it is reasonable to assume that **x** is correct. If it is not **0**, but is the ith column of **H**, and if errors are unlikely, it is reasonable to assume that only one error was made, so the correct word differs from **x** on the ith digit. If $\mathbf{H}\mathbf{x}^T$ is not **0** and not a column of **H**, at least two errors occurred in transmission and error correction cannot be carried out this way.

10.4.3 Hamming Codes

Recall that for error correction, we want the $(n-k) \times n$ parity check matrix **H** to have columns that are nonzero and distinct. Now if $p = n - k$, the columns of **H** are bit strings of length p. There are 2^p such strings, $2^p - 1$ nonzero ones. The *Hamming code* $\mathcal{H}_p$ arises when we take **H** to be a matrix whose columns are all of these $2^p - 1$ nonzero bit strings of length p, arranged in any order. Technically, to conform with our definition, the last $n-k$ columns should form the identity matrix. The resulting code is a $k \to n$ code, where $n = 2^p - 1$ and $k = n - p = 2^p - 1 - p$. For instance, if $p = 2$, then $n = 2^2 - 1 = 3$ and a typical **H** is given by

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

It is easy to see that **H** defines a $1 \to 3$ triple repetition code. If $p = 3$, then $n = 2^3 - 1 = 7$, and a typical **H** is given by

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

We get a $4 \to 7$ code. The Hamming codes were introduced by Hamming [1950] and Golay [1949]. [Note that if we change the order of columns in **H**, we get (essentially) the same code or set of codewords. For any two such codes are seen to be equivalent (in the sense to be defined in Exercise 25) by changing the order of the parity check digits. That is why we speak of *the* Hamming code $\mathcal{H}_p$.]

Theorem 10.8 In the Hamming codes $\mathcal{H}_p, p \geq 2$, the minimum distance d between two codewords is 3.

Proof. Since the columns of the parity check matrix $\mathbf{H}$ are nonzero and distinct, single errors can be corrected (see the discussion following Theorem 10.7). Thus, by Theorem 10.2, $d \geq 3$. Now it is easy to show that for $p \geq 2$, there are always three nonzero bit strings of length p whose sum (under addition modulo 2) is zero (Exercise 23). If these occur as columns u, v, and w of the matrix $\mathbf{H}$, we take $\mathbf{x}$ to be the vector that is 1 in positions u, v, and w, and 0 otherwise. Then $\mathbf{Hx}^T$ is the sum of the uth, vth, and wth columns of $\mathbf{H}$, and so is $\mathbf{0}$. Thus, $\mathbf{x}$ is a codeword of weight 3. By Theorem 10.5, $d \leq 3$. Q.E.D.

It follows from Theorem 10.8 that the Hamming codes *always* detect up to two errors and correct up to one. In Exercise 27 we ask the reader to show that the Hamming codes can *never* correct more than one error.

EXERCISES FOR SECTION 10.4

1. Suppose that
$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}.$$
Using $\mathbf{M}$, find the codeword $x_1x_2\cdots x_n$ corresponding to each of the following message words $a_1a_2\cdots a_k$.

   (a) 11 (b) 10 (c) 01 (d) 00

2. Suppose that
$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}.$$
Repeat Exercise 1 for the following message words.

   (a) 111 (b) 101 (c) 000

3. Suppose that
$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}.$$
Repeat Exercise 1 for the following message words.

   (a) 1111 (b) 1000 (c) 0001

4. Find the linear code C generated by the matrices $\mathbf{M}$ of:

   (a) Exercise 1 (b) Exercise 2 (c) Exercise 3

5. Find a generator matrix for the code that translates 0 into 000000 and 1 into 111111.

6. Each of the following codes is linear. For each, find the minimum distance between two codewords.

   (a) 000000, 001001, 010010, 100100, 011011, 101101, 110110, 111111

   (b) 0000, 0011, 0101, 1001, 0110, 1010, 1100, 1111

   (c) 00000, 00011, 00101, 01001, 10001, 00110, 01010, 01100, 10010, 10100, 11000, 01111, 11011, 10111, 11101, 11110

   (d) 11111111, 10101010, 11001100, 10011001, 11110000, 10100101, 11000011, 10010110, 00000000, 01010101, 00110011, 01100110, 00001111, 01011010, 00111100, 01101001

7. Find the parity check matrix $\mathbf{H}$ corresponding to $\mathbf{M}$ of:

   (a) Exercise 1 (b) Exercise 2 (c) Exercise 3

8. Find the generator matrix $\mathbf{M}$ corresponding to the following parity check matrices:

$$\text{(a) } \mathbf{H} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \qquad \text{(b) } \mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

$$\text{(c) } \mathbf{H} = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

9. Find the parity check equations in each of the following:

   (a) Exercise 1 (b) Exercise 2 (c) Exercise 3
   (d) Exercise 8(a) (e) Exercise 8(b) (f) Exercise 8(c)

10. Suppose that we use the $k \to 5k$ five repetitions code with $k = 2$. Find the parity check matrix $\mathbf{H}$ and derive the parity check equations.

11. For the matrix $\mathbf{H}$ of Exercise 8(a), suppose that a word $\mathbf{x}$ is received over a noisy channel. Assume that errors are unlikely. For each of the following $\mathbf{x}$, determine if an error was made in transmission.

    (a) $\mathbf{x} = 111000$ (b) $\mathbf{x} = 111100$ (c) $\mathbf{x} = 000100$

12. Repeat Exercise 11 for $\mathbf{H}$ of Exercise 8(b) and

    (a) $\mathbf{x} = 11101$ (b) $\mathbf{x} = 01101$ (c) $\mathbf{x} = 00011$

13. Repeat Exercise 11 for $\mathbf{M}$ of Exercise 1 and

    (a) $\mathbf{x} = 1111$ (b) $\mathbf{x} = 1000$ (c) $\mathbf{x} = 0100$

14. In each part of Exercise 11, if possible, correct an error if there was one.

15. In each part of Exercise 12, if possible, correct an error if there was one.

16. Consider the parity check matrix

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

Find all the codewords in the corresponding code. Does the code correct all single errors?

17. In the Hamming code $\mathcal{H}_p$ with $p = 3$, encode the following words.

(a) 1001 (b) 0001 (c) 1110

18. For the Hamming code $\mathcal{H}_p$ with $p = 4$, find a parity check matrix and encode the word 10000000000.
19. Find all codewords in the Hamming code $\mathcal{H}_2$.
20. Find all codewords in the Hamming code $\mathcal{H}_3$.
21. Find a word of weight 3 in the Hamming code $\mathcal{H}_p$, for every $p \geq 2$.
22. Consider a codeword $x_1 x_2 \cdots x_n$ in the Hamming code $\mathcal{H}_p$, $p \geq 2$. Let $k = n - p$.

(a) Show that if $x_i = 0$, all $i \leq k$, then $x_i = 0$, all i.

(b) Show that it is impossible to have exactly one x_i equal to 1 and all other x_i equal to 0.

23. Show that if $p \geq 2$, there are always three bit strings of length p whose sum (under addition modulo 2) is zero.
24. Let C be a linear code with codewords of length n in which some codewords have odd weight. Form a new code C' by adding 0 at the end of each word of C of even weight and 1 at the end of each word of C of odd weight.

(a) If $\mathbf{M}$ is the parity check matrix for C, show that the parity check matrix for C' is given by

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ & & & 0 \\ & \mathbf{M} & & 0 \\ & & & \vdots \\ & & & 0 \end{bmatrix}.$$

(b) Show that the distance between any two codewords of C' is even.

(c) Show that if the minimum distance d between two codewords of C was odd, the minimum distance between two codewords of C' is $d + 1$.

(d) For all $p \geq 2$, find a linear code with codewords of length $n = 2^p$ and with minimum distance between two codewords equal to 4.

25. Two n-codes C and C' are *equivalent* if they differ only in the order of symbols, that is, if one can be obtained from the other by applying the same permutation to each element. For example, the following C and C' are equivalent:

C	C'
0000	0000
1111	1111
0011	0101
1100	1010

Show that if $\mathbf{M}$ is a generator matrix giving rise to a code C, and $\mathbf{M}'$ is obtained from $\mathbf{M}$ by interchanging rows in the matrix $\mathbf{G}$ corresponding to $\mathbf{M}$, and $\mathbf{M}'$ gives rise to a code C', then C and C' are equivalent.

26. Show that under a linear code, the set of codewords under the operation $+$ defines a group.

27. Show that the Hamming codes $\mathcal{H}_p$, $p \geq 2$, are perfect 1-error-correcting codes. (See Exercise 26, Section 10.3 for the definition of perfect t-error-correcting codes.)

28. We can extend the theory of linear codes from binary codes to q-ary codes (Exercises 7, 13, and 17–22, Section 10.3), where q is a power of a prime. We continue to define a codeword from a message word by adding parity check digits, and specifically use a generator matrix $\mathbf{M}$, but with addition in the finite field $GF(q)$, rather than modulo 2. Note that most of our results hold in this general setting, with the major exceptions being noted in Exercises 29 and 30. Suppose that $q = 5$.

    (a) If $\mathbf{M}$ is as in Exercise 1, find the codeword corresponding to the message words:

    i. 14 ii. 03 iii. 13

    (b) If $\mathbf{M}$ is as in Exercise 2, find the codeword corresponding to the message words:

    i. 124 ii. 102 iii. 432

29. Continuing with Exercise 28

    (a) Note that $d(\mathbf{x}, \mathbf{y})$ is not necessarily $wt(\mathbf{x} + \mathbf{y})$.

    (b) What is the relation between distance and weight?

30. Continuing with Exercise 28, show that Theorem 10.6 may not be true.

31. Continuing with Exercise 28, show that Theorem 10.5 still holds.

32. Suppose that C is a q-ary block n-code of minimum distance d. Generalize the Hamming bound (Theorem 10.3) to find an upper bound on $|C|$.

10.5 THE USE OF BLOCK DESIGNS TO FIND ERROR-CORRECTING CODES[11]

10.5.1 Hadamard Codes

One way to find error-correcting codes is to concentrate first on finding a rich set C of codewords and then perform an encoding of messages into C. This is the opposite of the approach we have taken so far, which first defined the encoding, and defined the set C to be the set of all the words arising from the encoding. Our goal will be to find a set C of codewords that corrects a given number of errors or, equivalently, has a given minimum distance d between codewords, and which has many codewords in it, thus allowing more possible message blocks to be encoded.

Recall that an (n, d)-code has codewords of length n and the minimum distance between two codewords is d. A useful way to build (n, d)-codes C is to use the

[11]This section depends on Section 9.4.

incidence matrix of a (v, k, λ)-design (see Section 9.4.2). Each row of this incidence matrix has $r = k$ 1's and the rest 0's. The rows have length $b = v$. Any two rows have 1's in common in a column exactly λ times. The rows can define codewords for an (n, d)-code, with $n = v$. What is d? Let us consider two rows, the ith and jth. There are λ columns where there is a 1 in both rows. There are k 1's in each row, and hence there are $k - \lambda$ columns where row i has 1 and row j has 0, and there are $k - \lambda$ columns where row i has 0 and row j has 1. All other columns have 0's in both rows. It follows that the two rows differ in $2(k - \lambda)$ places. This is true for every pair of rows, so

$$d = 2(k - \lambda).$$

Theorem 9.17 says that for arbitrarily large m, there are Hadamard designs of dimension m, that is, $(4m - 1, 2m - 1, m - 1)$-designs. It follows that we can find (v, k, λ)-designs with arbitrarily large $k - \lambda$, and hence (n, d)-codes for arbitrarily large d. For given a Hadamard design of dimension m, we have

$$d = 2\,(k - \lambda) = 2\,[(2m - 1) - (m - 1)] = 2m.$$

Hence, if there are Hadamard designs of dimension m for arbitrarily large m, there are error-correcting codes that will detect up to $d - 1 = 2m - 1$ errors and correct up to $\lceil (d/2) - 1 \rceil = m - 1$ errors for arbitrarily large m. These codes are $(4m - 1, 2m)$-codes, since each codeword has length $4m - 1$. We call them *Hadamard codes.* We shall first set out to prove the existence of Hadamard designs of arbitrarily large dimension m, that is, to prove Theorem 9.17. Then we shall ask how rich are the codes constructed from the incidence matrices of Hadamard designs; that is, how do these codes compare to the richest possible (n, d)-codes?

10.5.2 Constructing Hadamard Designs[12]

The basic idea in constructing Hadamard designs is that certain kinds of matrices will give rise to the incidence matrices of these designs. An $n \times n$ matrix $\mathbf{H} = (h_{ij})$ is called a *Hadamard matrix of order* n if h_{ij} is $+1$ or -1 for every i and j and if

$$\mathbf{H}\mathbf{H}^T = n\mathbf{I},$$

where $\mathbf{H}^T$ is the transpose of $\mathbf{H}$ and $\mathbf{I}$ is the $n \times n$ identity matrix.[13] The matrix $n\mathbf{I}$ has n's down the diagonal, and 0's otherwise. To give an example, suppose that

$$\mathbf{H} = \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}. \tag{10.11}$$

Then

$$\mathbf{H}^T = \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$$

[12]This subsection may be omitted.

[13]Hadamard matrices were introduced by Hadamard [1893] and, in an earlier use, by Sylvester [1867]. Plotkin [1960] and Bose and Shrikhande [1959] constructed binary codes from Hadamard matrices.

and

$$\mathbf{HH}^T = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}.$$

Hence, the matrix $\mathbf{H}$ of (10.11) is a Hadamard matrix of order 2. A Hadamard matrix of order 4 is given by

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & -1 & 1 \\ 1 & -1 & -1 & -1 \\ -1 & -1 & -1 & 1 \\ 1 & -1 & 1 & 1 \end{bmatrix}. \tag{10.12}$$

For it is easy to check that

$$\mathbf{HH}^T = \begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix}.$$

An equivalent definition of Hadamard matrix is the following. Recall that if

$$(x_1, x_2, \ldots, x_n) \text{ and } (y_1, y_2, \ldots, y_n)$$

are two vectors, their *inner product* is defined to be

$$\sum_i x_i y_i.$$

Then an $n \times n$ matrix of $+1$'s and -1's is a Hadamard matrix if for all i, the inner product of the ith row with itself is n, and for all $i \neq j$, the inner product of the ith row with the jth is 0. This is just a restatement of the definition. To illustrate, let us consider the matrix $\mathbf{H}$ of (10.11). The first row is the vector $(1, 1)$ and the second is the vector $(-1, 1)$. The inner product of the first row with itself is

$$1 \cdot 1 + 1 \cdot 1 = 2.$$

The inner product of the first row with the second is

$$1 \cdot (-1) + 1 \cdot 1 = 0,$$

and so on.

A Hadamard matrix is called *normalized* if the first row and first column consist of just $+1$'s. For example, the matrix

$$\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{10.13}$$

is a normalized Hadamard matrix. So is the matrix

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}. \tag{10.14}$$

Some of the most important properties of Hadamard matrices are summarized in the following theorem.

Theorem 10.9 If $\mathbf{H}$ is a normalized Hadamard matrix of order $n > 2$, then $n = 4m$ for some m. Moreover, each row (column) except the first has exactly $2m$ $+1$'s and $2m$ -1's, and for any two rows (columns) other than the first, there are exactly m columns (rows) in which both rows (columns) have $+1$.

We prove Theorem 10.9 later in this subsection. However, let us see how Theorem 10.9 gives rise to a proof of Theorem 9.17. Given a normalized Hadamard matrix, we can define a (v, k, λ)-design. We do so by deleting the first row and column and, in what is left, replacing every -1 by a 0. As we shall see, this gives an incidence matrix $\mathbf{A}$ of a (v, k, λ)-design. Performing this procedure on the normalized Hadamard matrix of (10.14) first gives

$$\begin{bmatrix} -1 & 1 & -1 \\ 1 & -1 & -1 \\ -1 & -1 & 1 \end{bmatrix}$$

and then the incidence matrix

$$\mathbf{A} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} B_1 & B_2 & B_3 \end{array} \\ \left(\begin{array}{ccc} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{array}\right) \end{array}.$$

This gives rise to the blocks

$$B_1 = \{2\}, \quad B_2 = \{1\}, \quad B_3 = \{3\},$$

and to a design with $v = 3$, $k = 1$, and $\lambda = 0$. (This is, technically, not a design, since we have required that $\lambda > 0$. However, it illustrates the procedure.)

To show that this procedure always gives rise to a (v, k, λ)-design, let us note that by Theorem 10.9, the incidence matrix $\mathbf{A}$ has $4m-1$ rows and $4m-1$ columns, so $b = v = 4m - 1$. Also, one 1 (the first one) has been removed from each row, so each row of $\mathbf{A}$ has $2m-1$ 1's and $r = 2m-1$. By a similar argument, each column of $\mathbf{A}$ has $2m-1$ 1's and $k = 2m-1$. Finally, any two rows had a pair of 1's in the first column in common, so now have one fewer pair, namely $m - 1$, in common. Hence, $\lambda = m-1$. Thus, we have a (v, k, λ)-design with $v = 4m-1, k = 2m-1, \lambda = m-1$, provided that $m \geq 2$.[14]

In fact, the procedure we have described can be reversed. We have the following theorem.

Theorem 10.10 There exists a Hadamard design of dimension m if and only if there exists a Hadamard matrix of order $4m$.

[14] If $m = 1$, λ turns out to be 0, as we have seen.

Proof. It remains to start with a Hadamard design of dimension m and construct a Hadamard matrix of order $4m$ from it. Let $\mathbf{A}$ be the incidence matrix of such a design. Construct $\mathbf{H}$ by changing 0 to -1, adding a row of 1's at the top, and adding a column of 1's at the front. It is easy to verify that $\mathbf{H}$ is a Hadamard matrix. The proof is left as an exercise (Exercise 14). Q.E.D.

Let us next show how to construct normalized Hadamard matrices of order $4m$ for arbitrarily large m. This will prove Theorem 9.17. Suppose that $\mathbf{H}$ is an $n \times n$ Hadamard matrix. Let $\mathbf{K}$ be the matrix

$$\mathbf{K} = \begin{bmatrix} \mathbf{H} & \mathbf{H} \\ \mathbf{H} & -\mathbf{H} \end{bmatrix},$$

where $-\mathbf{H}$ is the matrix obtained from $\mathbf{H}$ by multiplying each entry by -1. For example, if $\mathbf{H}$ is the matrix of (10.13), $\mathbf{K}$ is the matrix

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}.$$

$\mathbf{K}$ is always a Hadamard matrix of order $2n$, for each entry is $+1$ or -1. Moreover, it is easy to show that the inner product of a row with itself is always $2n$, and the inner product of two different rows is always 0. Finally, we observe that if $\mathbf{H}$ is normalized, so is $\mathbf{K}$. Thus, there are normalized Hadamard matrices of arbitrarily large orders, and in particular of all orders 2^p for $p \geq 1$. A Hadamard matrix of order $4m$ for $m \geq 2$ gives rise to a $(4m-1, 2m-1, m-1)$-design. Since $4m = 2^p$, $m \geq 2$ is equivalent to $m = 2^k, k \geq 1$, this completes the proof of Theorem 9.17.

It is interesting to go through the construction we have just gone through if the matrix $\mathbf{H}$ is the matrix shown in (10.14). Then the corresponding matrix $\mathbf{K}$ is given by

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix}. \qquad (10.15)$$

The corresponding incidence matrix for a (v, k, λ)-design is given by deleting the first row and column and changing the remaining -1's to 0's. We get

$$\begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix}. \qquad (10.16)$$

This is the incidence matrix for a symmetric BIBD with $v = 7, k = 3$, and $\lambda = 1$. It defines a design different from that of Example 9.10. Repeating the construction one more time gives a 16×16 Hadamard matrix and hence a $(15, 7, 3)$-design. It is this design whose blocks are shown in Table 9.27. A proof is left to the reader as Exercise 8.

The construction procedure we have outlined certainly leads to normalized Hadamard matrices of order $4m$ for arbitrarily large m, in particular for $m = 1, 2, 4, 8, \ldots$, and so of orders $4, 8, 16, 32, \ldots$. Notice that we do not claim to construct normalized Hadamard matrices of order $4m$ for every m. It is conjectured that for every m, a normalized Hadamard matrix of order $4m$ exists. Whether or not this conjecture is true is another open question of the mathematics of combinatorial design and coding. Note that if there is any Hadamard matrix of order $4m$, there is a normalized one (Exercise 6). (For other methods of constructing Hadamard matrices, see, for example, Hall [1967] or MacWilliams and Sloane [1983]. Craigen and Wallis [1993] is a survey article on Hadamard matrices marking the 100th anniversary of their introduction.)

We conclude this subsection by filling in the one missing step in the proof of Theorem 9.17, namely, by proving Theorem 10.9.[15] We need one preliminary result.

Theorem 10.11 If $\mathbf{H}$ is a Hadamard matrix, so is $\mathbf{H}^T$.

Proof. If $\mathbf{H}\mathbf{H}^T = n\mathbf{I}$, then

$$\frac{\mathbf{H}}{\sqrt{n}}\frac{\mathbf{H}^T}{\sqrt{n}} = \mathbf{I},$$

where $\mathbf{H}/\sqrt{n}$ is obtained from $\mathbf{H}$ by dividing each entry by $\sqrt{n}$, and similarly for $\mathbf{H}^T/\sqrt{n}$. Since $\mathbf{AB} = \mathbf{I}$ implies that $\mathbf{BA} = \mathbf{I}$ for square matrices $\mathbf{A}$ and $\mathbf{B}$, we have

$$\frac{\mathbf{H}^T}{\sqrt{n}}\frac{\mathbf{H}}{\sqrt{n}} = \mathbf{I}$$

or

$$\mathbf{H}^T\mathbf{H} = n\mathbf{I}.$$

Since $(\mathbf{H}^T)^T = \mathbf{H}$, it follows that $\mathbf{H}^T$ is Hadamard. Q.E.D.

Proof of Theorem 10.9. Let $\mathbf{H}$ be a normalized Hadamard matrix of order n. The results for columns follow from the results for rows by applying Theorem 10.11. Hence, we need only prove the row results. Since the first row of $\mathbf{H}$ is

$$1 \quad 1 \quad 1 \quad \cdots \quad 1$$

and since the inner product with any other row is 0, any other row must have an equal number of $+1$'s and -1's. Thus, n is even, and there are $(n/2)$ $+1$'s and $(n/2)$ -1's. Interchange columns so that the second row has $+1$'s coming first and then -1's. Thus, the first two rows look like this:

$$\begin{array}{cccccccc} 1 & 1 & \cdots & 1 & 1 & 1 & \cdots & 1 \\ 1 & 1 & \cdots & 1 & -1 & -1 & \cdots & -1 \end{array}$$

[15]The rest of this subsection may be omitted.

Second Row	1	1	-1	-1
ith row, $i \neq 1, 2$	1	-1	1	-1
	u digits	$\frac{n}{2} - u$ digits	v digits	$\frac{n}{2} - v$ digits

Figure 10.2: Schematic for computing the inner product of the second and ith rows ($i \neq 1, 2$) in a normalized Hadamard matrix.

This interchange does not affect the size of the matrix or the number of $+1$'s in each row or the number of columns where two rows share a $+1$. Consider the ith row, $i \neq 1, 2$. The ith row has u digits which are $+1$ and $n/2 - u$ digits which are -1 in the first half (the first $n/2$ entries), and v digits which are $+1$ and $n/2 - v$ digits which are -1 in the second half. The first row and ith row have inner product 0. Hence, the ith row has $(n/2)$ $+1$'s and

$$u + v = \frac{n}{2}. \tag{10.17}$$

The second and ith rows have inner product 0 if $i \neq 1, 2$. Hence,

$$u - \left(\frac{n}{2} - u\right) - v + \left(\frac{n}{2} - v\right) = 0 \tag{10.18}$$

(see Figure 10.2). It follows that

$$u - v = 0. \tag{10.19}$$

Equations (10.17) and (10.19) give us

$$2u = \frac{n}{2}$$

or

$$n = 4u.$$

Thus, n is a multiple of 4, which proves the first part of Theorem 10.9. Moreover, there are $(n/2 = 2u)$ $+1$'s in each row other than the first, proving another part of Theorem 10.9. Finally, the second and ith rows, $i \neq 1, 2$, have exactly u columns in which $+1$'s appear in common. The same can be shown for rows j and i, $j \neq i$, $j, i \neq 1$, by repeating the argument above, interchanging columns so that the $+1$'s in row j come first and then the -1's. This completes the proof of Theorem 10.9.
Q.E.D.

10.5.3 The Richest (n, d)-Codes

We consider next the question: How "rich" are the codes obtained from Hadamard matrices? That is, compared to other possible codes with the same n (length of codewords) and the same d (minimum distance between codewords), does such a code have many or few codewords?

Let $A(n, d)$ be the maximum number of codewords in an (n, d)-code. Equation (10.5) of Theorem 10.3 gives an upper bound for $A(n, d)$. Our next result gives a better bound in most cases.

Theorem 10.12 (Plotkin [1960]) Suppose that an (n, d)-code has N codewords. If $d > n/2$, then

$$N \leq \frac{2d}{2d-n}.$$

Proof.[16] Form the $N \times n$ matrix $\mathbf{M} = (m_{ij})$ whose rows are codewords. Consider

$$S = \sum d(u, v), \tag{10.20}$$

where the sum is taken over all (unordered) pairs of words u, v from the set of codewords. We know that $d(u, v) \geq d$ for all u, v, so

$$S \geq \binom{N}{2} d = \frac{N(N-1)}{2} d. \tag{10.21}$$

Let $t_0^{(i)}$ be the number of times 0 appears in the ith column of $\mathbf{M}$ and $t_1^{(i)}$ be the number of times 1 appears in the ith column. Note that

$$S = \sum_{\{i,k\}} \sum_j |m_{ij} - m_{kj}| = \sum_j \sum_{\{i,k\}} |m_{ij} - m_{kj}|. \tag{10.22}$$

Now

$$\sum_{\{i,k\}} |m_{ij} - m_{kj}|$$

is the number of times that we have a pair of rows i and k with one having a 1 in the jth column and the other a 0 in the jth column, and this is given by $t_0^{(j)} t_1^{(j)}$. Thus, by (10.22),

$$S = \sum_j t_0^{(j)} t_1^{(j)}. \tag{10.23}$$

Now

$$t_1^{(j)} = N - t_0^{(j)},$$

so

$$t_0^{(j)} t_1^{(j)} = t_0^{(j)} [N - t_0^{(j)}].$$

We seek an upper bound on $t_0^{(j)} t_1^{(j)}$. Let $f(x)$ be the function defined by

$$f(x) = x(N-x),$$

[16] The proof may be omitted.

$0 \leq x \leq N$. Note that $f(x)$ is maximized when $x = N/2$ and the maximum value of $f(x)$ is $(N/2)(N/2) = N^2/4$. Thus,

$$t_0^{(j)} t_1^{(j)} \leq \frac{N^2}{4},$$

so, by (10.23),

$$S \leq n\frac{N^2}{4}. \tag{10.24}$$

Then, by (10.21) and (10.24),

$$\begin{aligned} \frac{N(N-1)d}{2} &\leq \frac{nN^2}{4}, \\ (N-1)d &\leq \frac{nN}{2}, \\ N\left(d - \frac{n}{2}\right) &\leq d. \end{aligned}$$

Since $d > n/2$, we have

$$N \leq \frac{d}{d - n/2} = \frac{2d}{2d - n}.$$

Q.E.D.

Corollary 10.12.1 $A(n,d) \leq \dfrac{2d}{2d-n}$, if $d > \dfrac{n}{2}$.

A normalized Hadamard matrix of order $4m$ gives rise to an (n,d)-code with $n = 4m-1, d = 2m$, and $N = 4m-1$ codewords. Thus,

$$A(4m-1, 2m) \geq 4m - 1. \tag{10.25}$$

By Corollary 10.12.1,

$$A(4m-1, 2m) \leq \frac{2(2m)}{2(2m) - (4m-1)} = \frac{4m}{1} = 4m. \tag{10.26}$$

Thus, the code obtained from a normalized Hadamard matrix is close to the best possible. One gets the best possible code in terms of number of codewords by adding one codeword:

$$1 \quad 1 \quad \cdots \quad 1.$$

Adding this word is equivalent to modifying our earlier construction and not deleting the first row of the normalized Hadamard matrix of order $4m$. The distance of this word to any codeword obtained from the Hadamard matrix is $2m$, because any such codeword has $2m-1$ 1's and $2m$ 0's. It follows that we have a code with $4m$ words, each of length $4m-1$, and having minimum distance between two codewords equal to $2m$. This code will be called the $(4m-1)$-*Hadamard code.* It follows that

$$A(4m-1, 2m) \geq 4m. \tag{10.27}$$

Equations (10.26) and (10.27) give us the following theorem.

Theorem 10.13 For all m for which there is a (normalized)[17] Hadamard matrix of order $4m$,

$$A(4m-1,2m)=4m.$$

The bound is attained by using a $(4m-1)$-Hadamard code.

Let $B(n,d)$ be the maximum number of words of length n each of which has distance exactly d from the other. Clearly, $B(n,d) \leq A(n,d)$.

Corollary 10.13.1 For all m for which there is a (normalized) Hadamard matrix of order $4m$,

$$B(4m-1,2m)=4m.$$

Proof. The code we have constructed has all distances equal to $2m$. Q.E.D.

It is interesting to note that sometimes we can get much richer codes by increasing the length of codewords by 1. We shall prove the following theorem.

Theorem 10.14 For all m for which there is a (normalized) Hadamard matrix of order $4m$,

$$A(4m,2m)=8m.$$

This theorem shows that in cases where there is a (normalized) Hadamard matrix of order $4m$, adding one digit leads to a doubling in the number of possible codewords.

We first show that $A(4m,2m)\geq 8m$ by constructing a $(4m,2m)$-code with $8m$ codewords.

Starting with a $4m\times 4m$ normalized Hadamard matrix $\mathbf{H}$, we have constructed a $(4m-1,2m-1,m-1)$-design. Let $\mathbf{A}$ be the incidence matrix of this design. Use the rows of $\mathbf{A}$ and the rows of $\mathbf{B}$, the matrix obtained from $\mathbf{A}$ by interchanging 0's and 1's. This gives us $8m-2$ words in all. The distance between two words in $\mathbf{A}$ is $2m$, the distance between two words in $\mathbf{B}$ is $2m$, and the distance between a word in $\mathbf{A}$ and a word in $\mathbf{B}$ is $2m-1$ if one is the ith word in $\mathbf{A}$ and the second is the jth word in $\mathbf{B}$ with $i\neq j$, and it is $4m-1$ if one is the ith word in $\mathbf{A}$ and the second the ith word in $\mathbf{B}$. Now add the digit 1 before words of $\mathbf{A}$ and the digit 0 before words of $\mathbf{B}$, obtaining two sets of words, $\mathbf{A}'$ and $\mathbf{B}'$. The distance between two words of $\mathbf{A}'$ or of $\mathbf{B}'$ is now still $2m$, while the distance between the ith word of $\mathbf{A}'$ and the jth word of $\mathbf{B}'$ is $2m$ if $i\neq j$ and $4m$ if $i=j$. There are still only $8m-2$ words. Add the $4m$-digit words

$$0\quad 0\quad \cdots\quad 0$$

and

$$1\quad 1\quad \cdots\quad 1.$$

We now have $8m$ words. The distance between these two new words is $4m$. Any word in $\mathbf{A}'$ or $\mathbf{B}'$ has $2m$ 1's and $2m$ 0's [since words of $\mathbf{A}$ had $k=(2m-1)$ 1's]. Thus, the new words have distance $2m$ from any word of $\mathbf{A}'$ or of $\mathbf{B}'$. We now

[17]Recall that if there is any Hadamard matrix of order $4m$, there is a normalized one.

Table 10.1: The $8m = 16$ Codewords of the $4m$-Hadamard Code Derived from the Normalized Hadamard Matrix $\mathbf{H}$ of (10.15)

1	1	1	1	1	1	1	1
1	0	1	0	1	0	1	0
1	1	0	0	1	1	0	0
1	0	0	1	1	0	0	1
1	1	1	1	0	0	0	0
1	0	1	0	0	1	0	1
1	1	0	0	0	0	1	1
1	0	0	1	0	1	1	0

From $\mathbf{H}$

0	0	0	0	0	0	0	0
0	1	0	1	0	1	0	1
0	0	1	1	0	0	1	1
0	1	1	0	0	1	1	0
0	0	0	0	1	1	1	1
0	1	0	1	1	0	1	0
0	0	1	1	1	1	0	0
0	1	1	0	1	0	0	1

From $-\mathbf{H}$

have $8m$ codewords, each of which has length $4m$ and with the minimum distance between two such codewords equal to $2m$. This set of codewords defines the $4m$-*Hadamard code*. It is equivalent to the set of codewords obtained by taking the normalized Hadamard matrix $\mathbf{H}$ of order $4m$, not deleting the first row or column, considering the matrix

$$\begin{bmatrix} \mathbf{H} \\ -\mathbf{H} \end{bmatrix},$$

changing -1 to 0, and using the rows in the resulting matrix as codewords. The $4m$-Hadamard code with $m = 2$ derived from the normalized Hadamard matrix of (10.15) is given in Table 10.1. This completes the proof of the fact that if there is a (normalized) Hadamard matrix of order $4m$, then

$$A(4m, 2m) \geq 8m. \tag{10.28}$$

We now prove that for all $m > 0$,

$$A(4m, 2m) \leq 8m, \tag{10.29}$$

which will prove Theorem 10.14.

Theorem 10.15 If $0 < d < n$, then $A(n, d) < 2A(n-1, d)$.

Proof. Consider an (n, d)-code C of $A(n, d)$ words. Let C' be obtained from C by choosing all codewords beginning with 1 and deleting the first digit of these codewords. Then C' defines an $(n-1, d)$-code, so

$$|C'| \leq A(n-1, d).$$

Similarly, if C'' is obtained by choosing all codewords of C beginning with 0 and deleting the first digit, then

$$|C''| \leq A(n-1, d).$$

Thus,

$$A(n, d) = |C| = |C'| + |C''| \leq 2A(n-1, d).$$

Q.E.D.

Now (10.29) follows immediately from Theorem 10.15 and (10.26) for we have

$$A(4m, 2m) \leq 2A(4m-1, 2m) \leq 2(4m) = 8m.$$

(Note that Theorem 10.13 could not be used directly to give this result. Why?)

The $4m$-Hadamard codes we have constructed are also called *Reed-Muller codes* (of the first kind), after Reed [1954] and Muller [1954]. The results in Theorems 10.13 and 10.14 are due to Levenshtein [1964], who obtains more general results as well.

In closing this section we note that the $(4m-1)$-Hadamard codes and the $4m$-Hadamard codes are in fact linear codes if 0's and 1's are interchanged and $m = 2^p$ for some p, as in our construction. However, the 24-Hadamard code is an example of one that is not linear. We shall not present a proof of these facts here, but instead refer the reader to the coding theory literature: for example, to MacWilliams and Sloane [1983]. (See also Exercises 25–28.)

10.5.4 Some Applications

Example 10.8 The Mariner 9 Space Probe Error-correcting codes are widely used to send messages back to Earth from deep-space missions such as the 1997 NASA Pathfinder mission to Mars. The Reed-Muller codes were used as early as the 1972 Mariner 9 space probe, which returned photographs of Mars. The specific code used was the $4m$-Hadamard code for $m = 8$, that is, the code based on a 32×32 normalized Hadamard matrix. This code has 64 codewords of 32 bits each. A photograph of Mars was broken into very small dots, and each dot was assigned 1 of 64 levels of grayness, which was encoded in one of the 32-bit codewords. (See Posner [1968], van Lint [1982], or MacWilliams and Sloane [1983] for more details.) ■

Example 10.9 The Role of Error-Correcting Codes in the Development of Compact Discs The compact disc (CD) encodes a signal digitally with high "information density." It is designed to have powerful error-correction properties, which explains the tremendous improvement in sound that CD recordings introduced. Errors can result from damage in use, dust, manufacturing defects, and so on. Such errors lead to "bursts" or consecutive sequences of errors. A coding technique known as CIRC (Cross-Interleaved Reed-Solomon Code) leads to the ability to correct up to 4000 consecutive errors through the use of two interleaved codes. For more details about the use of error-correcting codes in CDs, see Bossert, *et al.* [1997], Immink [1991], Peek [1985], Pinch [2001], or Vries and Odaka [1982]. ■

Example 10.10 "Reading" DNA to Produce Protein Golomb [1962] speculated that error-correcting codes might be at work in genetic coding, specifically in the process by which DNA strands are "read" in order to produce proteins. Error correction would be used whenever a sequence of three bases does not correspond to a code for an amino acid (see Example 2.2). Golomb speculated that a $4m$-Hadamard code is used. The smallest such code that would encode for the 20 different amino acids has 24 codewords and results from a 12×12 normalized Hadamard matrix. (That such a matrix exists does not follow from our theorems.) Codewords have length 12. A DNA chain would be encoded into a string of 0's and 1's in two steps. First, one of the letters A, C, G, and T would be encoded as 00, one as 01, one as 10, and one as 11. This represents a DNA chain of length m as a bit string of length $2m$, the message word. Every message word would be broken into blocks of length $k = 2$ and encoded by a $2 \to 12$ code. Every six bits of the message word (every three letters of the chain) or every 36 bits of the code would correspond to an amino acid. ■

EXERCISES FOR SECTION 10.5

1. A Hadamard design includes the following blocks. Find the missing blocks.

$$\{1,2,4\}, \{2,3,5\}, \{1,3,6\}, \{3,4,7\}.$$

2. Suppose that $\mathbf{A}$ is the incidence matrix of a (b, v, r, k, λ)-design and that a code is made up of the rows of this matrix.
   (a) What is the distance between two words in this code?
   (b) How many errors will the code detect?
   (c) How many errors will the code correct?
   (d) Suppose that $\mathbf{B}$ is the incidence matrix of the complementary design (Exercise 20, Section 9.4). What is the distance between the ith row of $\mathbf{A}$ and the jth row of $\mathbf{B}$ if $i \neq j$?
   (e) If rows of $\mathbf{B}$ form a code, how many errors will the code detect?
   (f) How many will it correct?
3. Given a Steiner triple system of nine varieties, build a code by using the rows of the incidence matrix. Find the minimum distance between two codewords.
4. (a) Could there be a 3×3 Hadamard matrix?
   (b) A 6×6?
5. If $\mathbf{H}$ is a Hadamard matrix and some row is multiplied by -1, is the resulting matrix still Hadamard? Why?
6. Show that if there is a Hadamard matrix of order $4m$, there is a normalized Hadamard matrix of order $4m$.
7. Suppose that

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}.$$

   Use $\mathbf{H}$ to find an 8×8 Hadamard matrix and a $(7, 3, 1)$-design.
8. Derive the $(15, 7, 3)$-design of Table 9.27 from a 16×16 normalized Hadamard matrix.

9. (a) Find a $(4m-1)$-Hadamard code for $m=4$.
   (b) How many errors can this code detect?
   (c) How many errors can it correct?
10. (a) Find a $4m$-Hadamard code for $m=4$.
   (b) How many errors can this code detect?
   (c) How many errors can it correct?
11. Repeat Exercise 9 for $m=8$.
12. Repeat Exercise 10 for $m=8$.
13. The following is an $(11,5,2)$-design. Use it to find a code of 12 words that can detect up to five errors and a code of 24 words that can detect up to five errors.

$$\begin{array}{llll} \{1,2,3,4,9\}, & \{1,2,5,6,10\}, & \{2,3,6,7,11\}, & \{1,4,6,7,8\}, \\ \{2,5,7,8,9\}, & \{3,6,8,9,10\} & \{1,7,9,10,11\}, & \{2,4,8,10,11\}, \\ \{4,5,6,9,11\}, & \{3,4,5,7,10\}, & \{1,3,5,8,11\}. & \end{array}$$

14. Complete the proof that the matrix $\mathbf{H}$ constructed in the proof of Theorem 10.10 defines a Hadamard matrix.
15. Illustrate the construction in the proof of Theorem 10.10 by starting with the incidence matrix of a $(7,3,1)$-design and constructing the corresponding Hadamard matrix.
16. Suppose that $\mathbf{H}$ is the incidence matrix of a $(4m-1, 2m-1, m-1)$-design and $\mathbf{K}$ is the incidence matrix of the complementary design (Exercise 20, Section 9.4).
   (a) What is the distance between the ith row of $\mathbf{H}$ and the jth row of $\mathbf{H}$?
   (b) What is the distance between the ith row of $\mathbf{H}$ and the jth row of $\mathbf{K}$?
17. Suppose that S is a set of binary codewords of length n with Hamming distance $d(a,b)$ equal to d for each pair of words a and b in S. Let T be defined from S by taking complements of words in S (interchanging 0 and 1).
   (a) What is the distance between two words of T?
   (b) What is the distance between a word of S and a word of T?
   (c) If n is 12 and d is 5, how many errors will the code T detect? How many will it correct?
   (d) Suppose that a code is defined from the words in S and the words in T. If n is 12 and d is 5, how many errors will this code detect? How many will it correct?
18. Suppose that the $m \times n$ matrix $\mathbf{M}$ is the incidence matrix of a block design, and that the ith row and the jth row have 1's in common in u columns.
   (a) What is the inner product of these two rows?
   (b) What is the inner product in the complementary design (Exercise 20, Section 9.4)?
19. Find a 2×4 matrix of 1's and -1's such that the inner product of rows i and j is 0 if $i \neq j$ and 4 if $i = j$.

20. For what values of k does there exist a $2 \times k$ matrix of 1's and -1's such that the inner product of rows i and j is 0 if $i \neq j$ and k if $i = j$?
21. Show that $A(n, n-1) = 2$ for n sufficiently large.
22. (a) Show that if $n > m$, then $A(n,d) \geq A(m,d)$.

    (b) If $n > m$, is it necessarily the case that $A(n,d) > A(m,d)$? Why?
23. (a) Find an upper bound for $A(4m-2, 2m)$.

    (b) Find $A(4m-2, 2m)$ exactly for arbitrarily large values of m.
24. Using the results of Exercise 17, show that

$$A(2d,d) \geq 2B(2d,d).$$

25. Show that the $(4m-1)$-Hadamard code as we have defined it is not linear. (*Hint:* Consider sums of codewords.)
26. If $m = 2^p$ and if 0's and 1's are interchanged in the $(4m-1)$-Hadamard code as we have defined it, the code becomes linear. Show this for the case $m = 1$ by finding a generator matrix.
27. Continuing with Exercise 26, show linearity for the case $m = 2$ as follows:

    (a) Using the generator matrix obtained by taking

$$\mathbf{G} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix},$$

    generate the corresponding code.

    (b) Show that this code is equivalent (Exercise 25, Section 10.4) to the $(4m-1)$-Hadamard code with $m = 2$ and 0's and 1's interchanged.
28. If $m \neq 2^p$, show that even if 0's and 1's are interchanged in the $(4m-1)$-Hadamard code, the code is not linear. (*Hint:* How many codewords does a linear code have?)
29. Recall that the *Hamming weight* of a bit string $\mathbf{x}$, denoted $wt(\mathbf{x})$, is the number of nonzero digits of $\mathbf{x}$. Let $A(n,d,w)$ be the maximum number of codewords in an (n,d)-code in which each word has the same Hamming weight w. Show that:

    (a) $A(n, 2d-1, w) = A(n, 2d, w)$ (b) $A(n, 2d, w) = A(n, 2d, n-w)$

    (c) $A(n, 2d, w) = 1$ if $w < d$ (d) $A(n, 2d, d) = \lfloor n/d \rfloor$

REFERENCES FOR CHAPTER 10

AVIZENIUS, A., "Fault-Tolerant Systems," *IEEE Trans. Comput., C-25* (1976), 1304–1312.

BERLEKAMP, E. R., *Algebraic Coding Theory*, McGraw-Hill, New York, 1968.

BLAKE, I. F., and MULLIN, R. C., *The Mathematical Theory of Coding*, Academic Press, New York, 1975. (Abridged as *An Introduction to Algebraic and Combinatorial Coding Theory*, Academic Press, New York, 1976.)

BOCK, H. H. (ed.), *Classification and Related Methods of Data Analysis*, North-Holland, Amsterdam, 1988.

Bose, R. C., and Shrikhande, S. S., "A Note on a Result in the Theory of Code Construction," *Inf. Control, 2* (1959), 183–194.

Bossert, M., Braĭtbakh, M., Zyablov, V. V., and Sidorenko, V. R., "Codes That Correct Multiple Burst Errors or Erasures (Russian)" *Problemy Peredachi Informatsii, 33* (1997), 15–25; translation in *Problems Inform. Transmission, 33* (1997), 297–306.

Cameron, P. J., and van Lint, J. H., *Designs, Graphs, Codes, and Their Links*, London Mathematical Society Student Texts 22, Cambridge University Press, Cambridge, 1991.

Carter, W. C., and Bouricius, W. G., "A Survey of Fault-Tolerant Architecture and Its Evaluation," *Computer, 4* (1971), 9–16.

Craigen, R., and Wallis, W. D., "Hadamard Matrices: 1893 – 1993," *Congr. Numer., 97* (1993), 99–129.

Day, W. H. E., "The Sequence Analysis and Comparison Bibliography," at http://www.classification-society.org/sequence.html, Oct. 12, 2002.

Day, W. H. E., and McMorris, F. R., "Critical Comparison of Consensus Methods for Molecular Sequences," *Nucleic Acids Research, 20* (1992), 1093–1099.

Day, W. H. E., and McMorris, F. R., "Discovering Consensus Molecular Sequences," in O. Opitz, B. Lausen, and R. Klar (eds.), *Information and Classification: Proceedings of the 16th Annual Conference of the Gesellsshaft fuer Klassifikation e.V.*, Springer-Verlag, Heidelberg, 1993, 393–402.

Day, W. H. E., and McMorris, F. R., *Axiomatic Consensus Theory in Group Choice and Biomathematics*, SIAM Publications, Philadelphia, 2003.

Dornhoff, L. L., and Hohn, F. E., *Applied Modern Algebra*, Macmillan, New York, 1978.

Fisher, J. L., *Application-Oriented Algebra: An Introduction to Discrete Mathematics*, Harper & Row, New York, 1977.

Galas, D. J., Eggert, M., and Waterman, M. S., "Rigorous Pattern-Recognition Methods for DNA Sequences. Analysis of Promoter Sequences from *Escherichia Coli*," *J. Molecular Biol., 186* (1985), 117–128.

Golay, M. J. E., "Notes on Digital Coding," *Proc. IEEE, 37* (1949), 657.

Goldie, C. M., and Pinch, R. G. E., *Communication Theory*, London Mathematical Society Student Texts 20, Cambridge University Press, Cambridge, 1991.

Golomb, S. W., "Efficient Coding for the Deoxyribonucleic Channel," in *Mathematical Problems in the Biological Sciences*, Proceedings of Symposia in Applied Mathematics, Vol. 14, American Mathematical Society, Providence, RI, 1962, 87–100.

Hadamard, J., "Résolution d'une Question Relative aux Déterminants," *Bull. Sci. Math., 17* (1893), 240–248.

Hall, M., *Combinatorial Theory*, Ginn (Blaisdell), Boston, 1967. (Second printing, Wiley, New York, 1980.)

Hamming, R. W., "Error Detecting and Error Correcting Codes," *Bell Syst. Tech. J., 29* (1950), 147–160.

Hill, R., *A First Course in Coding Theory*, Oxford University Press, New York, 1986.

Immink, K. A. S., *Coding Techniques for Digital Recorders*, Prentice Hall International, Hertfordshire, UK, 1991.

Janowitz, M., LaPointe, F.-J., McMorris, F. R., Mirkin, B., and Roberts, F. S., *Bioconsensus*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Vol. 61, American Mathematical Society, Providence, RI, 2003.

Johnson, P. E., *Social Choice: Theory and Research*, Sage Publications, Thousand Oaks, CA, 1998.

Kannan, S., "A Survey of Tree Consensus Criteria and Methods," in S. Tavare (ed.), *Proceedings of Phylogeny Workshop*, DIMACS Tech. Rep. 95–48, DIMACS Center, Rutgers University, Piscataway, NJ, 1995.

Levenshtein, V. I., "The Application of Hadamard Matrices to a Problem in Coding," *Probl. Kibern., 5* (1961), 123–136. [English transl.: *Probl. Cybern., 5* (1964), 166–184.]

MacWilliams, F. J., and Sloane, N. J. A., *The Theory of Error-Correcting Codes*, Vols. 1 and 2, North-Holland, Amsterdam, 1983.

Mirkin, B., and Roberts, F. S., "Consensus Functions and Patterns in Molecular Sequences," *Bull. Math. Biol., 55* (1993), 695–713.

Muller, D. E., "Application of Boolean Algebra to Switching Circuit Design and to Error Detection," *IEEE Trans. Comput., 3* (1954), 6–12.

Peek, J. B. H., "Communications Aspects of the Compact Disc Digital Audio System," *IEEE Commun. Magazine, 23* (1985), 7–15.

Peterson, W. W., *Error Correcting Codes*, MIT Press, Cambridge, MA, 1961.

Peterson, W. W., and Weldon, E. J., *Error Correcting Codes*, MIT Press, Cambridge, MA, 1972. (Second edition of Peterson [1961].)

Pinch, R. G. E., "Coding Theory: The First 50 Years," at http://pass.maths.org/issue3/codes/, Feb. 21, 2001.

Pless, V., *Introduction to the Theory of Error-Correcting Codes*, Wiley, New York, 1998.

Plotkin, M., "Binary Codes with Specified Minimum Distances," *IEEE Trans. Inf. Theory, 6* (1960), 445–450.

Poli, A., and Huguet, L., *Error Correcting Codes: Theory and Applications*, Prentice Hall International, Hemel Hempstead/Masson, Paris, 1992.

Posner, E. C., "Combinatorial Structures in Planetary Reconnaissance," in H. B. Mann (ed.), *Error Correcting Codes*, Wiley, New York, 1968.

Reed, I. S., "A Class of Multiple-Error-Correcting Codes and the Decoding Scheme," *IEEE Trans. Inf. Theory, 4* (1954), 38–49.

Roberts, F. S., *Discrete Mathematical Models, with Applications to Social, Biological, and Environmental Problems*, Prentice Hall, Englewood Cliffs, NJ, 1976.

Sellers, F. F., Hsiao, M. Y., and Bearnson, L. W., *Error Detecting Logic for Digital Computers*, McGraw-Hill, New York, 1968.

Slepian, D., "A Class of Binary Signaling Alphabets," *Bell Syst. Tech. J., 35* (1956), 203–234. (a)

Slepian, D., "A Note on Two Binary Signaling Alphabets," *IEEE Trans. Inf. Theory, 2* (1956), 84–86. (b)

Slepian, D., "Some Further Theory of Group Codes," *Bell Syst. Tech. J., 39* (1960), 1219–1252.

Sylvester, J. J., "Thoughts on Inverse Orthogonal Matrices, Simultaneous Sign Successions, and Tesselated Pavements in Two or More Colors, with Applications to Newton's Rule, Ornamental Tile-Work, and the Theory of Numbers," *Philos. Mag., 34* (1867), 461–475.

Tietäväinen, A., "On the Nonexistence of Perfect Codes over Finite Fields," *SIAM J. Appl. Math., 24* (1973), 88–96.

van Lint, J. H., "Nonexistence Theorems for Perfect Error-Correcting Codes," in *Computers in Algebra and Number Theory*, Vol. 4 (SIAM-AMS Proceedings), 1971.

van Lint, J. H., "Coding, Decoding and Combinatorics," in R. J. Wilson (ed.), *Applications of Combinatorics*, Shiva, Nantwich, UK, 1982, 67–74.

van Lint, J. H., *Introduction to Coding Theory*, 3rd ed., Springer-Verlag, New York, 1999.

Vries, L. B., and Odaka, K., "CIRC – The Error-Correcting Code for the Compact Disc Digital Audio System," *Collected Papers from the AES Premiere Conference, Rye, New York*, Audio Engineering Society, New York, 1982, 178–188.

Wakerly, J., *Error Detecting Codes, Self-Checking Circuits, and Applications*, Elsevier North-Holland, New York, 1978.

Waterman, M. S., "Consensus Patterns in Sequences," in M. S. Waterman (ed.), *Mathematical Methods for DNA Sequences*, CRC Press, Boca Raton, FL, 1989, 93–115.

Waterman, M. S., *Introduction to Computational Biology; Maps, Sequences and Genomes*, CRC Press, Boca Raton, FL, 1995.

Waterman, M. S., Arratia, R., and Galas, D. J., "Pattern Recognition in Several Sequences: Consensus and Alignment," *Bull. Math. Biol., 46* (1984), 515–527.

Welsh, D., *Codes and Cryptography*, Oxford University Press, New York, 1988.