Chapter 1

Introduction

Lecture 1, January 11, 2011

1.1 Examples

In the modern era, digital information has become a valuable commodity. For example, governments, corporations, and universities all exchange enormous quantities of digitized information every day. Since the transmission lines that we use for sending and receiving data are not perfect, it has become necessary to develop ways of detecting, or even correcting, errors. The theory of error-correcting codes originated with Claude Shannon’s famous 1948 paper “A Mathematical Theory of Communication”. Usually, coding is defined as source coding and channel coding.

Source coding involves changing the message source to a suitable code to be transmitted through the channel. Two examples of source coding are the ASCII code (The American Standard Code for Information Interchange), which converts each character to a byte of 8 bits; the ISBN code (International Standard Book Number), a 10-digit codeword, by the publisher. The ISBN code will be discussed at the end of this chapter.

A simple communication model can be represented by Fig. 1.1.

Example 1.1. Consider the source encoding of four Letter Grades: A, B, C, D, as follows:

\[ A \rightarrow 00, \quad B \rightarrow 01, \quad C \rightarrow 10, \quad D \rightarrow 11. \]

Suppose the message ‘A’, which is encoded as 00, is transmitted over a noisy channel. The
message may become distorted and may be received as 01 (that there is only one error introduced) (see Fig. 1.2). The receiver may not realize that the message was corrupted. This communication fails.

Example 1.2. In Example 1.1, we perform the channel encoding by introducing a redundancy of one bit as follows:

00 → 000, 01 → 011, 10 → 101, 11 → 110.
Suppose that the message ‘A’, which is encoded as 000 after the source and channel encoding, is transmitted over a noisy channel, and that there is only one error introduced. Then the received word must be one of the following three: 100, 010 or 001. In this way, we can detect the error, as none of 100, 010 or 001 is among our encoded messages.

**Example 1.3.** Note that the above encoding scheme allows us to detect errors at the cost of reducing transmission speed as we have to transmit three bits for a message of two bits. The above channel encoding scheme does not allow us to correct errors. For instance, if 100 is received, then we do not know whether 100 comes from 000, 110 or 101. However, if more redundancy is introduced, we are able to correct errors. For instance, we can design the following channel coding scheme:

\[
\begin{align*}
00 & \rightarrow 00000, \quad 01 \rightarrow 01111, \quad 10 \rightarrow 10110, \quad 11 \rightarrow 11001.
\end{align*}
\]

Suppose that the message ‘A’ is transmitted over a noisy channel, and that there is only one error introduced. Then the received word must be one of the following five: 10000, 01000, 00100, 00010 or 00001. Assume that 10000 is received. Then we can be sure that 10000 comes from 00000 because there are at least two errors between 10000 and each of the other three encoded messages 01111, 10110 and 11001. Note that we lose even more in terms of information transmission speed in this case. See Fig. 1.4 for this example.

![Diagram of Example 1.3](image)

**Fig. 1.4: Example 1.3.**

**Example 1.4.** Here is a simple and general method of adding redundancy for the purpose of error correction. Assume that source coding has already been done and that the information consists of bit strings of fixed length \(k\). Encoding is carried out by taking a bit string and repeating it \(2r + 1\) times, where \(r \geq 1\) is a fixed integer. For instance, \(01 \rightarrow 01010101\) if \(k = 2\) and \(r = 2\). In this special case, decoding is done by first
considering the positions 1, 3, 5, 7, 9 of the received string and taking the first decoded bit as the one which appears more frequently at these positions; we deal similarly with the positions 2, 4, 6, 8, 10 to obtain the second decoded bit. For instance, the received string 1100100010 is decoded to 10. It is clear that, in this special case, we can decode up to two errors correctly. In the general case, we can decode up to \( r \) errors correctly. Since \( r \) is arbitrary, there are thus encoders which allow us to correct as many errors as we want. For obvious reasons, this method is called a repetition code. The only problem with this method is that it involves a serious loss of information transmission speed. Thus, we will look for more efficient methods.

The goal of channel coding is to construct encoders and decoders in such a way as to effect:

1). fast encoding of messages;
2). easy transmission of encoded messages;
3). fast decoding of received messages;
4). maximum transfer of information per unit time;
5). maximal detection or correction capability.

From the mathematical point of view, the primary goals are 4) and 5). However, 5) is, in general, not compatible with 4). Therefore, any solution is necessarily a trade-off among the five objectives.

Throughout this book, we are primarily concerned with channel coding. Channel coding is also called algebraic coding as algebraic tools are extensively involved in the study of channel coding.

1.2 Error Detection and Correction

We saw in Section 1 that the purpose of channel coding is to introduce redundancy to information messages so that errors that occur in the transmission can be detected or even corrected. In this chapter, we formalize and discuss the notions of error-detection and error-correction. We also introduce some well known decoding rules, i.e., methods
that retrieve the original message sent by detecting and correcting the errors that have occurred in the transmission.

The sequences “00”, “01” and so on in the previous example are known as binary codewords. Together they comprise a binary code. More generally, we introduce the following definitions.

**Definition 1.1 (code).** Let \( q \in \mathbb{N} \) and \( \Sigma_q = \{a_1, a_2, \ldots, a_q\} \) be a set of size \( q \), which we refer to as a code alphabet and whose elements are called code symbols.

1. A \( q \)-ary word of length \( n \) over \( \Sigma_q \) is a sequence \( w = w_1 w_2 \cdots w_n \) with each \( w_i \in \Sigma_q \) for all \( i \). Equivalently, \( w \) may also be regarded as the vector \((w_1, \ldots, w_n)\) in the space \( \Sigma_q^n = \Sigma_q \times \Sigma_q \times \cdots \times \Sigma_q \).

2. A \( q \)-ary code \( C \) is a set of \( q \)-ary words.

3. An element of \( C \) is called a codeword in \( C \).

4. The number of codewords in \( C \), denoted by \( |C| \), is called the size of the code \( C \).

5. A code of length \( n \) (all the codewords in it are of length \( n \)) and size \( M \) is called an \((n, M)\)-code.

**Remark.** Typically, we will take \( \Sigma_q \) to be the set \( \mathbb{Z}_q = \{0, 1, 2, \ldots, q - 1\} \).

**Example 1.5.**

1. A binary codeword, corresponding to the case \( q = 2 \), is just a finite sequence of 0s and 1s, i.e., the code symbols for a binary code are 0 and 1.

2. A ternary codeword, corresponding to the case \( q = 3 \), is just a finite sequence of 0s, 1s, and 2s.

3. The set of all words in the English language is a code over \( \Sigma_{26} \) or the 26-letter alphabet \( \{A, B, \ldots, Z\} \) (case-insensitive, never mind the letter uppercase or lowercase).

4. The set of all 10-digit telephone numbers in the United Kingdom is a 10-ary code of length 10. It is possible to use a code of over 82 million 10-digit telephone numbers (enough to meet the needs of the U.K.) such that if just one digit of any phone number is misdialed, the correct connection can still be made. Unfortunately, little thought was given to this, and as a result, frequently misdialed numbers do occur in the U.K. (as well as in North America)!
Definition 1.2 (Decoding rule). In a communication channel with coding, only codewords are transmitted. Suppose that a word $w$ is received. If $w$ is a valid codeword, we may conclude that there is no error in the transmission. Otherwise, we know that some errors have occurred. In this case, we need a rule for finding the most likely codeword sent. Such a rule is known as a decoding rule.

Definition 1.3 (Hamming distance). Let $x$ and $y$ be words of length $n$ over $\Sigma_q$. The (Hamming) distance between $x$ and $y$, denoted by $d(x, y)$, is defined to be the number of places in which $x$ and $y$ differ.

Remark. Notice that $d(x, y) \leq n$ and $d(x, y)$ is a metric on $\Sigma_q^n$ since it is always non-negative and satisfies

1. $d(x, y) = 0 \iff x = y$,

2. $d(x, y) = d(y, x)$ for all $x, y \in \Sigma_q^n$,

3. $d(x, y) \leq d(x, z) + d(z, y)$ for all $x, y, z \in \Sigma_q^n$.

The first two properties are immediate consequences of the definition, while the third property, known as the triangle inequality, follows from the simple observation that $d(x, y)$ is the minimum number of digit changes required to change $x$ to $y$, whereas if we were to change $x$ to $y$ by first changing $x$ to $z$ and then changing $z$ to $y$, we would require $d(x, z) + d(z, y)$ changes. Thus $d(x, y) \leq d(x, z) + d(z, y)$. Indeed, we can look at the third in another way, If $x = x_1 \ldots x_n$ and $y = y_1 \ldots y_n$, then $d(x, y) = d(x_1, y_1) + \ldots + d(x_n, y_n)$. So it is enough to prove 3) when $n = 1$, which we now assume. If $x = y$, then 3) is obviously true since $d(x, y) = 0$. If $x \neq y$, then either $z \neq x$ or $z \neq y$, so 3) is again true.

Lecture 2, January 13, 2011

Review:

Definition (Code). Let $q \in \mathbb{N}$ and $\Sigma_q = \{a_1, a_2, \ldots, a_q\}$ be a set of size $q$, which we refer to as a code alphabet and whose elements are called code symbols.

1. A $q$-ary word of length $n$ over $\Sigma_q$ is a sequence $w = w_1 w_2 \cdots w_n$ with each $w_i \in \Sigma_q$ for all $i$. Equivalently, $w$ may also be regarded as the vector $(w_1, \ldots, w_n)$ in the space $\Sigma_q^n = \Sigma_q \times \Sigma_q \times \cdots \times \Sigma_q$. 
2). A \( q \)-ary code \( C \) is a set of \( q \)-ary words.

3). An element of \( C \) is called a codeword in \( C \).

4). The number of codewords in \( C \), denoted by \( |C| \), is called the size of the code \( C \).

5). A code of length \( n \) (all the codewords in it are of length \( n \)) and size \( M \) is called an \( (n, M) \)-code.

Remark. An \( (n, M) \)-code \( C \) can be displayed as an \( M \times n \) matrix whose rows are the codewords and the number of rows is the size of code.

Examples. binary codeword; ternary codeword and so on.

Definition (Decoding rule). A rule for finding the most likely codeword sent.

Definition (Hamming distance). Let \( x \) and \( y \) be words of length \( n \) over \( \Sigma_q \). The (Hamming) distance between \( x \) and \( y \), denoted by \( d(x, y) \), is the number of positions at which the corresponding symbols of \( x \) and \( y \) are different.

\[
d(x, y) = \#\{1 \leq i \leq n \mid x_i \neq y_i\} \leq n.
\]

Remark. Notice that \( d(x, y) \) is a metric on \( \Sigma_q^n \) since it is always non-negative and satisfies

1). \( d(x, y) = 0 \iff x = y \),

2). \( d(x, y) = d(y, x) \) for all \( x, y \in \Sigma_q^n \),

3). (Triangle inequality) \( d(x, y) \leq d(x, z) + d(z, y) \) for all \( x, y, z \in \Sigma_q^n \).

New definitions:

Definition 1.4 (Nearest neighbour decoding). Suppose that codewords from a code \( C \) are being sent over a communication channel. If a word \( x \) is received, the nearest neighbour decoding rule (or minimum distance decoding rule) will decode \( x \) to \( c_x \) if \( d(x, c_x) \) is minimal among all the codewords in \( C \), i.e.,

\[
d(x, c_x) = \min_{c \in C} d(x, c).
\]

Remark (Detection and Correction). See the figure, if the receiving word \( y \) can not be a codeword of \( C \), then \( C \) can detect the error; moreover if \( c_y \) is unique and equal to \( x \), then \( C \) can correct the error.
Definition 1.5 (Weight). Let $x$ be a word in $\mathbb{Z}_q^n$. The (Hamming) weight of $x$, denoted by $\text{wt}(x)$, is defined to be the number of nonzero digits in $x$; i.e., $\text{wt}(x) = d(x, 0)$, where 0 is the zero word.

Remark. 1). Let $x$ and $y$ be binary words of length $n$ in $\mathbb{Z}_2^n$. Then

$$d(x, y) = \text{wt}(x - y) = \text{wt}(x) + \text{wt}(y) - 2\text{wt}(x \circ y).$$

Here, $x - y$ and $x \circ y$ are computed mod 2, digit by digit.

2). Let $x$ and $y$ be words in $\mathbb{Z}_q^n$. Then $d(x, y) = \text{wt}(x - y)$. Here, $x - y$ is computed mod $q$, digit by digit.

Definition 1.6 (Distance of Code). For a code $C$ containing at least two words, the (minimum) distance of $C$, denoted by $d(C)$, is

$$d(C) = \min\{d(x, y) : x, y \in C, x \neq y\}.$$ 

Remark. It is clear that in code $C$ there are codewords $x$ and $y$ such that $d(x, y) = d(C)$.

Definition 1.7 ($(n, M, d)$-Code). An $(n, M, d)$-code is a code of length $n$, size $M$ and distance $d$. The numbers $n$, $M$ and $d$ are called the parameters of the code.

Example 1.6. 1). Let $C = \{00000, 00111, 11111\}$ be a binary code. Then $d(C) = 2$ since

$$d(00000, 00111) = 3,$$
$$d(00000, 11111) = 5,$$
$$d(00111, 11111) = 2.$$

Hence, $C$ is a binary $(5, 3, 2)$-code.

2). Here is a $(5, 4, 3)$-code, consisting of four codewords from $\mathbb{Z}_2^5$, which are at least a distance 3 from each other:

$$C_3 = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 1 \\
1 & 0 & 1 & 1 & 0 \\
1 & 1 & 0 & 1 & 1
\end{pmatrix}$$
Upon considering each of the \( \binom{4}{2} = \frac{4 \times 3}{2} = 6 \) pairs of distinct codewords (rows), we see that the minimum distance of \( C_3 \) is indeed 3. With this code, we can either

a). detect up to two errors (since the members of each pair of distinct codewords are more than a distance 2 apart).

b). detect and correct a single error (since, if only a single error has occurred, the received vector will still be closer to the transmitted codeword than to any other).

The following theorem (cf. Fig. 1.6) shows how this works in general.

**Fig. 1.6**: Detection of up to \( t \) errors in a transmitted codeword \( x \) requires that all other codewords \( y \) lie outside a sphere \( S \) of radius \( t \) centered on \( x \). Correction of up to \( t \) errors requires that no sphere of radius \( t \) centered about any other codeword \( y \) overlaps with \( S \).

**Theorem 1.1 (Error Detection and Correction).** In a communication system,

1). A code \( C \) can detect up to \( t \) errors in every codeword \( \iff \) \( d(C) \geq t + 1 \); (usually \( C \) is call a \( t \)-error-detecting code)

2). A code \( C \) can correct up to \( t \) errors in every codeword \( \iff \) \( d(C) \geq 2t + 1 \). (usually \( C \) is call a \( t \)-error-correcting code)

**Proof.** 1). “\( \Leftarrow \)” Suppose \( d(C) \geq t + 1 \). Let a codeword \( x \) be transmitted such that \( t \) or fewer errors are introduced, resulting in a new vector \( y \in \Sigma_q^n \). Then \( d(x, y) = w(x - y) \leq t < t + 1 \leq d(C) \), so \( y \notin C \), i.e., the received vector cannot be another codeword. Hence the error can be detected.

“\( \Rightarrow \)” Suppose \( C \) can detect up to \( t \) errors. We prove this direction by contradiction. If \( d(C) < t + 1 \), i.e., \( d(C) \leq t \), then there is some pair of codewords \( x \) and
\( y \) with \( 1 \leq d(x, y) = d(C) \leq t \). Since it is possible to send the codeword \( x \) and receive another codeword \( y \) by the introduction of \( t \) or fewer errors, we conclude that \( C \) cannot detect \( t \) errors, contradicting our premise. Hence \( d(C) \geq t + 1 \).

2). “⇒” Suppose \( d(C) \geq 2t + 1 \). Let a codeword \( x \) be transmitted such that \( t \) or fewer errors are introduced, resulting in a new vector \( y \in \Sigma_q^n \) satisfying \( d(x, y) \leq t \). Since it is possible to send the codeword \( x \) and receive another codeword \( y \) by the introduction of \( t \) or fewer errors, we conclude that \( C \) cannot detect \( t \) errors, contradicting our premise. Hence \( d(C) \geq t + 1 \).

“⇐” Suppose \( C \) can correct up to \( t \) errors. We prove this direction by contradiction. If \( d(C) < 2t + 1 \), i.e., \( d(C) \leq 2t \), then there is some pair of distinct codewords \( x \) and \( x' \) with distance \( d(x, x') = d(C) \leq 2t \). If \( d(x, x') \leq t \), let \( y = x' \), so that \( 0 = d(y, x') < d(y, x) \leq t \). Since it is possible to send the codeword \( x \) and receive another codeword \( y = x' \) by the introduction of \( t \) or fewer errors, we conclude that \( C \) cannot correct \( t \) errors, contradicting our premise. Therefore, \( t + 1 \leq d(x, x') \leq 2t \), Without loss of generality, we may hence assume that \( x \) and \( x' \) differ in exactly the first \( d = d(x, x') \) positions. If the word

\[
Y = y_1 \cdots y_{t+1} \cdots y_d \quad y_{d+1} \cdots y_n
\]

is received (it is possible, since \( d(y, x) = t \)), then we have \( d(y, x') = d - t \leq 2t - t = t = d(y, x) \). But since \( d(y, x') \leq d(y, x) \), the received vector \( y \) cannot not be unambiguously decoded as \( x \) using nearest-neighbour decoding. This contradicts our premise. Hence \( d(C) \geq 2t + 1 \).

\[ \square \]

Lecture 3, January 18, 2011

Review:

Definitions. Until now, we have known the following definitions:

- \((q\text{-ary}) \) Code (Codewords, Length of the code, Size of the code), Decoding rule, Hamming distance between words, Nearest neighbour decoding rule, Hamming weight of word, Distance of code, \((n, M, d)\)-code, Error detection and correction.

Theorem (Error Detection and Correction). In a communication system,
1). A code \( C \) can detect up to \( t \) errors in every codeword \( \iff d(C) \geq t + 1; \)
(usually \( C \) is call a \( t \)-error-detecting code)

2). A code \( C \) can correct up to \( t \) errors in every codeword \( \iff d(C) \geq 2t + 1. \)
(usually \( C \) is call a \( t \)-error-correcting code)

New staffs:

Corollary 1.8. If a code \( C \) has minimum distance \( d \), then \( C \) can be used either (i) to
detect up to \( d - 1 \) errors or (ii) to correct up to \( \lfloor \frac{d-1}{2} \rfloor \) errors in any codeword. Here \( \lfloor x \rfloor \)
is the greatest integer less than or equal to \( x \) (Greatest integer function).

Remark. 1). From Example 1.6 (2), we know

\[
C_3 = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 1 \\
1 & 0 & 1 & 1 & 0 \\
1 & 1 & 0 & 1 & 1 \\
\end{pmatrix}
\]

is a \((5,4,3)\)-code. So by the corollary we have \( C_3 \) can detect up to \( d(C_3) - 1 = 2 \)
errors and can detect and correct up to \( \lfloor \frac{d(C_3)-1}{2} \rfloor = 1 \) error.

2). A good \((n,M,d)\)-code has small \( n \) (for rapid message transmission), large \( M \)
(to maximize the amount of information transmitted), and large \( d \) (to be able to correct many errors). A primary goal of coding theory is to find codes
that optimize \( M \) for fixed values of \( n \) and \( d \).

Definition 1.9 \((A_q(n,d))\). Let \( A_q(n,d) \) be the largest value of \( M \) such that there exists
a \( q \)-ary \((n,M,d)\)-code.

Remark. Since we have already constructed a binary \((5,4,3)\)-code, we know that \( A_2(5,3) \geq 4. \) We will soon see that \( 4 \) is in fact the maximum possible value of \( M \); i.e. \( A_2(5,3) = 4. \)

To help us tabulate \( A_q(n,d) \), let us first consider the following special cases:

Theorem 1.2 (Special Cases). For any values of \( q \) and \( n \), we have

1). \( A_q(n,1) = q^n; \)

2). \( A_q(n,n) = q. \)
Proof. 1). When the minimum distance \( d = 1 \), we require only that the codewords be distinct. The largest code with this property is the whole of \( \Sigma_q^n \), which has \( M = q^n \) codewords.

2). When the minimum distance \( d = n \), we require that any two distinct codewords differ in all \( n \) positions. In particular, this means that the symbols appearing in the first position must be distinct, so there can be no more than \( q \) codewords. A \( q \)-ary repetition code of length \( n \) is an example of an \( (n, q, n) \)-code, so the bound \( A_q(n, n) = q \) can actually be realized.

\[ \square \]

Lemma 1.10 (Reduction Lemma). If a \( q \)-ary \( (n, M, d) \)-code exists, with \( d \geq 2 \), then there also exists an \( (n - 1, M, d - 1) \) \( q \)-ary code. Moreover, we have \( A_q(n, d) \leq A_q(n - 1, d - 1) \).

Proof. Given an \( (n, M, d) \)-code \( C \), let \( x, y \in C \) be codewords such that \( d(x, y) = d \) and choose any position where \( x \) and \( y \) differ. Delete this column from all codewords of \( C \). Since \( d \geq 2 \), the codewords that result are distinct and form a \( (n - 1, M, d - 1) \)-code. \( \square \)

Remark. If \( d \geq 2 \), we have \( A_q(n, d) \leq A_q(n - 1, d - 1) \leq \cdots \leq A_q(n - d + 1, 1) = q^{n-d+1} \).

Theorem 1.3 (Even Values of \( d \)). Suppose \( d \) is even. Then a binary \( (n, M, d) \)-code exists. \( \iff \) a binary \( (n - 1, M, d - 1) \)-code exists.

Proof. “\( \Rightarrow \)” This follows from the above Lemma.

“\( \Leftarrow \)” Suppose \( C \) is a binary \( (n - 1, M, d - 1) \)-code. Let \( \hat{C} \) be the code of length \( n \) obtained by extending each codeword \( x \) of \( C \) by adding a parity bit \( \text{wt}(x) \mod 2 \), denote by \( \hat{x} \), i.e., if \( \text{wt}(x) \) even then extend \( x \) by \( 0 (\hat{x} = x 0) \); if \( \text{wt}(x) \) odd then extend \( x \) by \( 1 (\hat{x} = x 1) \). This makes the weight \( \text{wt}(\hat{x}) \) of every codeword \( \hat{x} \) of \( \hat{C} \) even, since \( \text{wt}(\hat{x}) = \text{wt}(x) + 0 \) if \( \text{wt}(x) \) even and \( \text{wt}(\hat{x}) = \text{wt}(x) + 1 \) if \( \text{wt}(x) \) odd. Then \( d(\hat{x}, \hat{y}) = \text{wt}(\hat{x}) + \text{wt}(\hat{y}) - 2\text{wt}(\hat{x} \circ \hat{y}) \) must be even for every pair of codewords \( \hat{x} \) and \( \hat{y} \) in \( \hat{C} \), so \( d(\hat{C}) \) is even. Note that \( d - 1 = d(C) \leq d(\hat{C}) \leq d \). But \( d - 1 \) is odd, so in fact \( d(\hat{C}) = d \). Thus \( \hat{C} \) is a \( (n, M, d) \)-code. \( \square \)

Corollary 1.11 (Maximum Code Size for Even \( d \)). If \( d \) is even, then \( A_2(n, d) = A_2(n - 1, d - 1) \).

Lecture 4, January 20, 2011
Remark. This result means that we only need to calculate $A_2(n,d)$ for odd $d$. In fact, in view of Theorem 1.1, there is little advantage in considering codes with even $d$ if the goal is error correction. In Table 1.1, we present values of $A_2(n,d)$ for $n \leq 16$ and odd values of $d \leq 7$.

<table>
<thead>
<tr>
<th>$n$</th>
<th>$d=3$</th>
<th>$d=5$</th>
<th>$d=7$</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>16</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>20</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>9</td>
<td>40</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>72</td>
<td>12</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>144</td>
<td>24</td>
<td>4</td>
</tr>
<tr>
<td>12</td>
<td>256</td>
<td>32</td>
<td>4</td>
</tr>
<tr>
<td>13</td>
<td>512</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td>14</td>
<td>1024</td>
<td>128</td>
<td>16</td>
</tr>
<tr>
<td>15</td>
<td>2048</td>
<td>256</td>
<td>32</td>
</tr>
<tr>
<td>16</td>
<td>2720-3276</td>
<td>256-340</td>
<td>36-37</td>
</tr>
</tbody>
</table>

Table 1.1: $A_2(n,d)$ for $n \leq 16$ and odd $d \leq 7$.

As an example, we now compute the value $A_2(5,3)$ entered in Table 1.1, after establishing a useful simplification, beginning with the following definition

**Definition 1.12 (Equivalent).** Two $q$-ary codes are equivalent if one can be obtained from the other by a combination of

(A) permutation of the positions of the code;

(B) permutation of the symbols appearing in a fixed position.

Remark. 1). If an $(n,M)$-code is displayed as an $M \times n$ matrix, then an operation of type (A) corresponds to a permutation, or rearrangement, of the columns of the matrix; while an operation of type (B) corresponds to a relabeling of the symbols appearing in a given column.

2). Note that the distances between codewords are unchanged by each of these operations. That is, equivalent codes have the same $(n, M, d)$ parameters and can correct the same number of errors.
Remark. The binary code
\[
\begin{pmatrix}
0 & 1 & 0 & 1 & 0 \\
1 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 1
\end{pmatrix}
\]
is seen to be equivalent to our previous \((5, 4, 3)\)-code \(C_3\) by interchanging the first two columns and then relabeling \(0 \leftrightarrow 1\) in the first and fourth columns of the resulting matrix.

\[
C_3 = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 1 \\
1 & 0 & 1 & 1 & 0 \\
1 & 1 & 0 & 1 & 1
\end{pmatrix}
\]

**Lemma 1.13 (Zero Vector).** Any code over an alphabet containing the symbol 0 is equivalent to a code containing the zero vector \(0\), i.e., if \(C\) is a code over \(\Sigma_q\) and \(0 \in \Sigma_q\), then there is another code \(\tilde{C}\) such that \(C\) is equivalent to \(\tilde{C}\) and \(0 \in \tilde{C}\).

**Proof.** Given a code of length \(n\), choose any codeword \(x = x_1 x_2 \cdots x_n\). For each \(i\) such that \(x_i \neq 0\), apply the relabeling \(0 \leftrightarrow x_i\) to the symbols in the \(i\)-th column.

Remark. 1). For a binary word \(x\), we have \(\text{wt}(x) = d(x, 0) = \text{number of 1s in } x\).

2). Armed with the above lemma and the concept of equivalence, it is now easy to prove that \(A_2(5, 3) = 4\). Let \(C\) be a \((5, M, 3)\)-code with \(M \geq 4\). Without loss of generality, we may assume that \(C\) contains the zero vector (if necessary, by replacing \(C\) with an equivalent code). Then there can be no nonzero codewords with just one or two 1s since \(d = 3\). Also, there can be at most one codeword with four or more 1s; otherwise there would be two codewords with at least three 1s in common positions and less than a distance 3 apart. Since \(M \geq 4\), there must be at least two codewords containing exactly three 1s. By rearranging columns, if necessary, we see that the code contains the codewords

\[
\begin{pmatrix}
0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 & 1
\end{pmatrix}
\]

There is no way to add any more codewords containing exactly three 1s and we can also now rule out the possibility of five 1s. This means that there can be at most four codewords, that is, \(A_2(5, 3) \leq 4\). Since we have previously shown that \(A_2(5, 3) \geq 4\), we deduce that \(A_2(5, 3) = 4\).
3). A fourth codeword, if present in the above code, must have exactly four 1s. The only possible position for the 0 symbol is in the middle position, so the fourth codeword must be 11011. We then see that the resulting code is equivalent to $C_3$ and hence the binary $(5, 4, 3)$ code is unique, up to equivalence.

The above trial-and-error approach becomes impractical for large codes. In some of these cases, an important bound, known as the sphere-packing or Hamming bound, can be used to establish that a code is as large as possible for given values of $n$ and $d$.

**Lemma 1.14 (Counting).** A sphere of radius $t$ centered on $x \in \Sigma_q^n$, with $0 \leq t \leq n$, contains exactly

$$\sum_{k=0}^{t} \binom{n}{k} (q-1)^k$$

vectors.

*Proof.* The number of vectors that are a distance $k$ from a fixed vector in $\Sigma_q^n$ is $\binom{n}{k} (q-1)^k$, because there are $\binom{n}{k}$ choices for the $k$ positions that differ from those of the fixed vector and there are $q - 1$ values that can be assigned independently to each of these $k$ positions. Summing over the possible values of $k$, we obtain the desired result. \qed

**Proof.** Let

$$\Theta = \{y \in \Sigma_q^n \mid d(y, x) \leq t\} \text{ and } \Theta_k = \{y \in \Sigma_q^n \mid d(y, x) = k\}.$$ 

Then $\Theta = \bigcup_{k=0}^{t} \Theta_k$ and it is a disjoint union. On the other hand, the number of vectors in $\Theta_k$ is $\binom{n}{k} (q-1)^k$, because there are $\binom{n}{k}$ choices for the $k$ positions that differ from those of the fixed vector and there are $q - 1$ values that can be assigned independently to each of these $k$ positions. The desired result follows immediately. \qed

**Theorem 1.4 (Sphere-Packing Bound).** If there is a $q$-ary $(n, M, 2t + 1)$-code, then we have

$$M \sum_{k=0}^{t} \binom{n}{k} (q-1)^k \leq q^n.$$  \hspace{1cm} (1.1)

In particular, we know

$$A_q(n, 2t + 1) \leq \frac{q^n}{\sum_{k=0}^{t} \binom{n}{k} (q-1)^k}.$$
Proof. By the triangle inequality, any two spheres of radius \( t \) that are centered on distinct codewords will have no vectors in common. The total number of vectors in the \( M \) spheres of radius \( t \) centered on the \( M \) codewords is thus given by the left-hand side of the above inequality; this number can be no more than the total number \( q^n \) of vectors in \( \Sigma_q^n \). \( \square \)

Proof. Let \( C \) be an \((n, M, 2t + 1)\)-code. For any codeword \( x \) in \( C \), let \( S_x \) be set of vectors contained in the sphere of radius \( t \) centered on \( x \), i.e.,
\[
S_x = \{ z \in \Sigma_q^n | d(y, x) \leq t \}.
\]
Then for any two codewords \( x \) and \( y \), we have \( S_x \cap S_y = \emptyset \). Because if \( z \in S_x \cap S_y \) then by triangle inequality we have \( d(x, y) \leq d(x, z) + d(y, z) \leq t + t = 2t < 2t + 1 \). It is a contradiction. Now we have the disjoint union \( \bigcup_{x \in C} S_x \) is a subset of \( \Sigma_q^n \), hence
\[
q^n \geq \sum_{x \in C} |S_x| \cdot \sum_{k=0}^{t} \binom{n}{k} (q - 1)^k = M \cdot \sum_{k=0}^{t} \binom{n}{k} (q - 1)^k.
\]
That is what we need. \( \square \)

Lecture 5, January 25, 2011

Remark. For our binary \((5, 4, 3)\)-code, Eq. (1.1) gives the bound \( M(1 + 5) \leq 2^5 = 32 \), which implies that \( A_2(5, 3) \leq 5 \). We have already seen that \( A_2(5, 3) = 4 \). This emphasizes, that just because some set of numbers \( n, M, t \) satisfy Eq. (1.1), there is no guarantee that such a code actually exists.

Definition 1.15 (Perfect code). A perfect code is an \((n, A_q(n, 2t + 1), 2t + 1)\) code for which equality occurs in 1.1, i.e., \( A_q(n, 2t + 1) \sum_{k=0}^{t} \binom{n}{k} (q - 1)^k = q^n \). For such a code, the \( M \) spheres of radius \( t \) centered on the codewords fill the whole space \( \Sigma_q^n \) completely, without overlapping.

Remark. 1). The codes that consist of a single codeword (taking \( t = n \) and \( M = 1 \)), codes that contain all vectors of \( \Sigma_q^n \) (with \( t = 0 \) and \( M = q^n \)), and the binary repetition code (with \( t = \frac{n-1}{2} \) and \( M = 2 \)) of odd length \( n \) are trivially perfect codes.

2). Let \( q = 2, n = 7 \) and \( t = 1 \) in the theorem, then we have
\[
A_2(7, 3) \leq \frac{2^7}{1 + 7} = 16.
\]
In next section, we will use the balanced block design to construct a binary \((7, 16, 3)\)-code, then we conclude that \( A_2(7, 3) = 16 \).
1.3 Balanced Block Designs

**Definition 1.16.** Let $S$ be a set of $v$ elements. A **balanced block design** consists of a collection of $b$ subsets of $S$, such that

1). each element of $S$ lies in exactly $r$ subsets;

2). each subset contains exactly $k$ elements;

3). each pair of elements occurs together in exactly $\lambda$ subsets.

Usually, the elements of $S$ are called points and the subsets are called blocks. Such a design is called a $(b,v,r,k,\lambda)$ design.

**Example 1.7.** Let $S = \{1, 2, 3, 4, 5, 6, 7\}$ and consider the subsets $\{1, 2, 4\}$, $\{2, 3, 5\}$, $\{3, 4, 6\}$, $\{4, 5, 7\}$, $\{5, 6, 1\}$, $\{6, 7, 2\}$, $\{7, 1, 3\}$ of $S$. Each number lies in exactly 3 blocks, each block contains 3 numbers, and each pair of numbers occurs together in exactly 1 block. The six lines and circle in Fig. 1.7 represent the blocks. Hence these subsets form a $(7, 7, 3, 3, 1)$ design.

![Fig. 1.7: Seven-point plane.](image)

**Remark.** 1). The parameters $(b,v,r,k,\lambda)$ are not independent. Consider the set of ordered pairs

\[ T = \{(x,B) : x \text{ is a point, } B \text{ is a block, } x \in B\}. \]

Since each of the $v$ points lie in $r$ blocks, there must be a total of $vr$ ordered pairs in $T$. Alternatively, we know that since there are $b$ blocks and $k$ points in each block, we can form exactly $bk$ such pairs. Thus $bk = vr$. Similarly, by considering the set

\[ U = \{(x,y,B) : x, y \text{ are distinct points, } B \text{ is a block, } x, y \in B\}, \]
we deduce
\[ bk(k - 1) = \lambda v(v - 1), \]
which, using \( bk = vr \), simplifies to \( r(k - 1) = \lambda(v - 1) \).

2). Each of the \( b \) blocks contains \( k \) elements and the total number of elements in the array of blocks is \( bk \). Since each element appears \( r \) times (the number of replications of the element) in the array and there are \( v \) elements, we have \( bk = vr \).

Considering the \( r \) blocks containing a given element, we can form \( r(k - 1) \) pairs containing this element. Since there are \( v - 1 \) possible sets of unordered pairs of elements containing the given element and each such pair occurs \( \lambda \) times in the array, we have \( r(k - 1) = \lambda(v - 1) \).

**Definition 1.17 (Symmetric design).** A block design is **symmetric** if \( v = b \) (and hence \( k = r \)); that is, the number of points and blocks is identical. For brevity, this is called a \( (v, k, \lambda) \) design.

**Definition 1.18 (Incidence matrix).** The **incidence matrix** of a block design is a \( v \times b \) matrix with entries
\[
a_{ij} = \begin{cases} 1 & \text{if } x_i \in B_j, \\ 0 & \text{if } x_i \notin B_j, \end{cases}
\]
where \( x_i, i = 1, \ldots, v \) are the design points and \( B_j, j = 1, \ldots, b \) are the design blocks.

**Remark.** 1). For our above \( (7, 3, 1) \) symmetric design, the incidence matrix \( A \) is
\[
\begin{pmatrix}
1 & 0 & 0 & 0 & 1 & 0 & 1 \\
1 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 1 & 0 & 0 & 0 & 1 \\
1 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 1 & 1 
\end{pmatrix}
\]

2). We now construct a \( (7, 16, 3) \) binary code \( C \) consisting of the zero vector \( \mathbf{0} \), the unit vector \( \mathbf{1} \), the 7 rows of \( A \), and the 7 rows of the matrix \( B \) obtained from \( A \) by the relabeling \( 0 \leftrightarrow 1 \), i.e.,
\[
C = \begin{pmatrix} 0 \\ 1 \\ A \\ B \end{pmatrix}
\]
i.e.,

\[
C = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 1 & 1 \\
x_1 & 1 & 0 & 0 & 1 & 0 \\
x_2 & 1 & 1 & 0 & 0 & 1 & 0 \\
x_3 & 0 & 0 & 0 & 0 & 1 & 1 \\
x_4 & 1 & 0 & 1 & 1 & 0 & 0 \\
x_5 & 0 & 1 & 0 & 1 & 1 & 0 \\
x_6 & 0 & 0 & 1 & 0 & 1 & 1 \\
x_7 & 0 & 0 & 0 & 1 & 0 & 1 \\
y_1 & 0 & 1 & 1 & 1 & 0 & 1 \\
y_2 & 0 & 0 & 1 & 1 & 1 & 0 \\
y_3 & 1 & 0 & 0 & 1 & 1 & 1 \\
y_4 & 0 & 1 & 0 & 0 & 1 & 1 \\
y_5 & 1 & 0 & 1 & 0 & 0 & 1 \\
y_6 & 1 & 1 & 0 & 1 & 0 & 0 \\
y_7 & 1 & 1 & 1 & 0 & 1 & 0 \\
\end{pmatrix}
\]

To find the minimum distance of this code, note that each row of \(A\) has exactly three 1s (since \(r = 3\)) and any two distinct rows of \(A\) have exactly one 1 in common (since \(\lambda = 1\)). Hence \(d(x_i, x_j) = 3 + 3 - 2 \cdot 1 = 4\) for \(1 \leq i \neq j \leq 7\). Likewise, \(d(y_i, y_j) = 4\) (they are equivalent). Furthermore,

\[
d(0, x_i) = 3, \quad d(0, y_i) = 4, \quad d(1, x_i) = 4, \quad d(1, y_i) = 3, \quad d(x_i, y_i) = d(0, 1) = 7,
\]

for \(i = 1, \ldots, 7\). Finally, \(x_i\) and \(y_j\) disagree in precisely those places where \(x_i\) and \(x_j\) agree, so

\[
d(x_i, y_j) = \text{wt}(x_i - y_j) = \text{wt}(1 - (x_i + x_j)) = 7 - \text{wt}(x_i - x_j) = 7 - d(x_i, x_j) = 7 - 4 = 3,
\]

for \(1 \leq i \neq j \leq 7\). Thus \(C\) is a \((7, 16, 3)\) code, which in fact is perfect, since the equality in Eq. (1.1) is satisfied.

The existence of a perfect binary \((7, 16, 3)\) code establishes \(A_2(7, 3) = 16\), so we have now established another entry of Table 1.1.
1.4 The ISBN Code

Modern books are assigned an International Standard Book Number (ISBN) by the publisher. The ISBN is a 10-digit number that uniquely identifies books and book-like products published internationally. The ten-digit number is divided into four parts of variable length, each part separated by a hyphen. The four parts of an ISBN are as follows:

**Group or country identifier** which identifies a national or geographic grouping of publishers;

**Publisher identifier** which identifies a particular publisher within a group;

**Title identifier** which identifies a particular title or edition of a title;

**Check digit** is the single digit at the end of the ISBN which validates the ISBN.

Every ISBN will consist of thirteen digits in 2007. The thirteen digit number is divided into five parts: the above four parts and a GS1 prefix: 978 or 979. In this section we just discuss the 10-digit ISBNs.

For example, Hill [1997] has the ISBN number 0-19-853803-0. The three hyphens separate the codeword into four fields. The first field specifies the language (0 means English), the second field indicates the publisher (19 means Oxford University Press), the third field (853803) is the book number assigned by the publisher, and the final digit (0) is a check digit. If the digits of the ISBN number are denoted $x_1 \ldots x_{10}$, then the check digit $x_{10}$ is chosen as

$$x_{10} \equiv \sum_{k=1}^{9} kx_k \pmod{11}. $$

If $x_{10}$ turns out to be 10, an $X$ is printed in place of the final digit. The tenth digit serves to make the weighted check sum

$$\sum_{k=1}^{10} kx_k = \sum_{k=1}^{9} kx_k + 10x_{10} \equiv \sum_{k=1}^{9} kx_k + 10 \sum_{k=1}^{9} kx_k = 11 \sum_{k=1}^{9} kx_k \equiv 0 \pmod{11}. $$

So, if $\sum_{k=1}^{10} kx_k \not\equiv 0 \pmod{11}$, we know that some errors have occurred. In fact, the ISBN code is designed to (i) detect a single error or (ii) detect a transposition error that results in two digits (not necessarily adjacent) being interchanged. For a received vector $y = y_1 \ldots y_{10}$ calculate its weighted check sum $Y = \sum_{k=1}^{10} ky_k$. If $Y \not\equiv 0 \pmod{10}$, then we have detected errors. Let us verify that this works for cases (i) and (ii) above. Suppose $x = x_1 \ldots x_{10}$ is the codeword sent.

(i) Suppose the received vector $y = y_1 \ldots y_{10}$ is the same as $x$ except that digit $x_j$ is
received as $x_j + a$ with $a \neq 0$. Then

$$Y = \sum_{k=1}^{10} ky_k = \sum_{k=1}^{10} kx_k + ja \equiv ja \not\equiv 0 \pmod{11},$$

since $j$ and $a$ are non-zero.

(ii) Suppose $y$ is the same as $x$ except that digit $x_j$ and $x_k$ have been transposed. Then

$$Y = \sum_{k=1}^{10} ky_k = \sum_{k=1}^{10} kx_k + (k-j)x_j + (j-k)x_k \equiv (k-j)(x_j - x_k) \not\equiv 0 \pmod{11},$$

if $k \neq j$ and $x_j \neq x_k$.

Lecture 6, January 27, 2011

Recall: The ISBN is a 10-digit number that uniquely identifies books and book-like products published internationally. We call the set of all ISBNs an ISBN code. The first 9 digits of an ISBN are number from 0 to 9 and the last digit is called the check digit could be 0, 1, \ldots, 9 and X, so we can look at ISBN code is a 11-ary code. If $x$ is an ISBN codeword with $x = x_1 \ldots x_{10}$ and we already know the first 9 digits, then the check digit $x_{10}$ is chosen as

$$x_{10} \equiv \sum_{k=1}^{9} kx_k \pmod{11}.$$ 

If $x_{10}$ turns out to be 10, an $X$ is printed in place of the final digit. The tenth digit serves to make the weighted check sum

$$\sum_{k=1}^{10} kx_k = \sum_{k=1}^{9} kx_k + 10x_{10} \equiv \sum_{k=1}^{9} kx_k + 10 \sum_{k=1}^{9} kx_k = 11 \sum_{k=1}^{9} kx_k \equiv 0 \pmod{11}.$$ 

So, if $\sum_{k=1}^{10} kx_k \not\equiv 0 \pmod{11}$, we know that some errors have occurred.

In fact, the ISBN code is designed to

(i) detect a single error or

(ii) detect a transposition error that results in two digits (not necessarily adjacent) being interchanged.

Suppose $x = x_1 \ldots x_{10}$ is the ISBN codeword sent. For a received vector $y = y_1 \ldots y_{10}$ calculate its weighted check sum $Y = \sum_{k=1}^{10} ky_k$. If $Y \not\equiv 0 \pmod{10}$, then we have detected errors. Let us verify that this works for cases (i) and (ii) above.
(i) Suppose the received vector $y = y_1 \ldots y_{10}$ is the same as $x$ except that digit $x_j$ is received as $x_j + a$ with $a \neq 0$. Then

$$Y = \sum_{k=1}^{10} ky_k = \sum_{k=1}^{10} kx_k + ja \equiv ja \not\equiv 0 \pmod{11},$$

since $j$ and $a$ are non-zero.

(ii) Suppose $y$ is the same as $x$ except that digit $x_j$ and $x_k$ have been transposed, i.e., $y_j = x_k$ and $y_k = x_j$. Then

$$Y = \sum_{k=1}^{10} ky_k = \sum_{k=1}^{10} kx_k + (k - j)x_j + (j - k)x_k \equiv (k - j)(x_j - x_k) \not\equiv 0 \pmod{11},$$

if $k \neq j$ and $x_j \neq x_k$.

Remark. In the above arguments we have used the property of the field $\mathbb{Z}_{11}$ (the integers modulo 11) that the product of two nonzero elements is always nonzero.

Let $m$ be a fixed positive integer. Now recall some properties of $\mathbb{Z}_m$:

The following result is known as the Division Algorithm: If $a \in \mathbb{Z}$, then there exist unique $q, r \in \mathbb{Z}$ such that $a = qm + r$, $0 \leq r < m$. Here $q$ is called quotient of the integer division of $a$ by $m$, and $r$ is called principal remainder. The principal remainder equal to one of the integers in the set $\mathbb{Z}_m = \{0, 1, \ldots, m-1\}$. We know $a \equiv b$ means $a - b$ is divisible by $m$, i.e., $m|(a - b)$ or $a = km + b$ for some integer $k$.

A positive integer $d$ is called a common divisor of the integers $a$ and $b$, if $d$ divides $a$ and $b$. The greatest possible such $d$ is called the greatest common divisor of $a$ and $b$, denoted $gcd(a, b)$ or just $(a, b)$. If $gcd(a, b) = 1$ then $a, b$ are called relatively prime. For the greatest common divisor we have the following theorem:

Fact. If $a$ and $b$ are nonzero integers with greatest common divisor $d$, then there exist integers $x$ and $y$ (called Bzout numbers or Bzout coefficients) such that

$$ax + by = d.$$ 

Definition 1.19 (Ring). A commutative ring $(R, +, \cdot)$ is a non-empty set $R$ together with two binary operations: addition ($+$) and multiplication ($\cdot$) such that:

1). $(R, +)$ is an abelian group,
2). \((R, \cdot)\) is associative and commutative, i.e., \(x \cdot (y \cdot z) = (x \cdot y) \cdot z\) and \(xy = yx\) for all \(x, y, x \in R\),

3). distributive laws over the addition: \(x \cdot (y + z) = x \cdot y + x \cdot z\) and \((y + z) \cdot x = y \cdot x + z \cdot x\) \(\forall x, y, z \in R\).

We will assume that the ring \(R\) has a multiplicative identity element, denoted by \(1_R\), such that \(1_R \cdot x = x = x \cdot 1_R, \forall x \in R\). We also denote the additive identity by \(0\), i.e., the identity element of abelian group \((R, +)\).

Now we can give \(Z_m\) the structure of a ring. We define addition and multiplication in \(Z_m\) by

\[
a + b (\text{ or } a \cdot b) = \text{ the principal remainder when } a + b (\text{ or } a \cdot b) \text{ is divided by } m.
\]

For example, in \(Z_{11}\) we have

\[
8 + 4 = 1, \quad 7 + 9 = 5, \quad 3 \cdot 4 = 1, \quad 7 \cdot 6 = 9.
\]

**Definition 1.20 (Invertible).** An element \(x\) in a ring \(R\) is invertible if \(\exists y \in R\) such that \(x \cdot y = 1\).

**Definition 1.21 (Field).** A field is a ring \(R\) in which \(1 \neq 0\) and every non-zero element is invertible.

Now we have the following theorem:

**Theorem 1.5.** \(Z_m\) is a field \(\iff \) \(m\) is a prime number.

**Proof.** \((\Rightarrow)\) Let \(Z_m\) be a field. If \(n = ab\), with \(1 < a, b \leq m - 1\), then \(a, b\) should be non-zero elements in field \(Z_m\), i.e., we have \(0 \neq a^{-1} \in Z_m\). Hence \(b \equiv a^{-1}ab \equiv a^{-1}m \equiv 0 \pmod{m}\), a contradiction. Hence \(m\) must be prime.

\((\Leftarrow)\) Let \(m\) be prime. Since \(Z_m\) contains the identity and is commutative, we need only verify that each non zero element \(a \neq 0\) has an inverse. Consider the elements \(ia\), for \(i = 1, 2, \ldots, m - 1\). Each of these elements must be nonzero since neither \(i\) nor \(a\) is divisible by the prime number \(m\). These \(m - 1\) elements are distinct from each other since, for \(1 \leq i, j \leq m - 1\),

\[
ia \equiv ja \iff (i - j)a \equiv 0 \iff m|(i - j)a \iff m|(i - j) \iff i = j.
\]
Thus, the \( m - 1 \) elements \( a, 2a, \ldots, (m - 1)a \) must be equal to the \( m - 1 \) elements \( 1, 2, \ldots, m - 1 \) in some order. One of them, say \( ia \), must be equal to 1. That is, \( a \) has inverse \( i \).

\[ \square \]

For this reason, the ISBN code is calculated in \( \mathbb{Z}_{11} \) not \( \mathbb{Z}_{10} \), where \( 2 \cdot 5 = 0 \pmod{10} \).

The ISBN code cannot be used to correct errors unless we know a priori which digit is in error. To do this, we first need to construct a table of inverses modulo 11 using the **Euclidean division algorithm**. For example, let \( y \) be the inverse of 2 modulo 11. Then \( 2y \equiv 1 \pmod{11} \) implies \( 2y = 11q + 1 \) or \( 1 = 2y - 11q \) for some integers \( y \) and \( q \). On dividing 11 by 2 as we would to show that \( \gcd(11, 2) = 1 \), we find \( 11 = 5 \cdot 2 + 1 \) so that \( 1 = 11 - 5 \cdot 2 \), from which we see that \( q = -1 \) and \( y = -5 \equiv 6 \pmod{11} \) is the inverse of 2. Similarly, \( 7^{-1} \equiv 8 \pmod{11} \) since \( 11 = 1 \cdot 7 + 4 \) and \( 7 = 1 \cdot 4 + 3 \) and \( 4 = 1 \cdot 3 + 1 \), so \( 1 = 4 - 1 \cdot 3 = 4 - 1(7 - 1 \cdot 4) = 2 \cdot 4 - 1 \cdot 7 = 2 \cdot (11 - 1 \cdot 7) - 1 \cdot 7 = 2 \cdot 11 - 3 \cdot 7 \). Thus \( -3 \cdot 7 = -2 \cdot 11 + 1 \); that is, \( -3 \equiv 8 \pmod{11} \) is the inverse of 7 mod 11. The complete table of inverses modulo 11 is shown in Table 1.2.

\[
\begin{array}{cccccccccc}
  x & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\
  x^{-1} & 1 & 6 & 4 & 3 & 9 & 2 & 8 & 7 & 5 & 10 \\
\end{array}
\]

Table 1.2: Inverses modulo 11.

Suppose that we detect an error and we also know that it is the digit \( x_j \) that is in error (and hence unknown). Then we can use our table of inverses to solve for the value of \( x_j \), assuming all of the other digits are correct. Since

\[
j x_j + \sum_{k=1, k \neq j}^{10} k x_k = 0 \pmod{11},
\]

we know that

\[
x_j \equiv -j^{-1} \sum_{k=1, k \neq j}^{10} k x_k \pmod{11}.
\]

For example, if we did not know the fourth digit \( x \) of the ISBN 0-19-x 53803-0, we would calculate

\[
x \equiv -4^{-1}(1 \cdot 0 + 2 \cdot 1 + 3 \cdot 9 + 5 \cdot 5 + 6 \cdot 3 + 7 \cdot 8 + 8 \cdot 0 + 9 \cdot 3 + 10 \cdot 0) \pmod{11} \\
\equiv -3(0 + 2 + 5 + 3 + 7 + 1 + 0 + 5 + 0) \equiv -3(1) \equiv 8 \pmod{11},
\]

which is indeed correct.

**Problems:** We also showed \( A_q(3, 2) = q^2 \) for all \( q \geq 2 \); \( d(\text{ISBN}) = 2 \).