Coding and Cryptography

T. W. Körner

January 29, 2018

Transmitting messages is an important practical problem. Coding theory includes the study of compression codes which enable us to send messages cheaply and error correcting codes which ensure that messages remain legible even in the presence of errors. Cryptography on the other hand, makes sure that messages remain unreadable — except to the intended recipient. These techniques turn out to have much in common.

Many Part II courses go deeply into one topic so that you need to understand the whole course before you understand any part of it. They often require a firm grasp of some preceding course. Although this course has an underlying theme, it splits into parts which can be understood separately and, although it does require knowledge from various earlier courses, it does not require mastery of that knowledge. All that is needed is a little probability, a little algebra and a fair amount of common sense. On the other hand, the variety of techniques and ideas probably makes it harder to understand everything *in the course than in a more monolithic course.*

Small print The syllabus for the course is defined by the Faculty Board Schedules (which are minimal for lecturing and maximal for examining). I should **very much** appreciate being told of any corrections or possible improvements **however minor**. This document is written in LaTeX2e and should be available from my home page

http://www.dpmms.cam.ac.uk/~twk

in latex, dvi, ps and pdf formats. Supervisors can obtain comments on the exercises at the end of these notes from the secretaries in DPMMS or by e-mail from me.

My e-mail address is `twk@dpmms`.

These notes are based on notes taken in the course of a previous lecturer Dr Pinch, on the excellent set of notes available from Dr Carne's home page and on Dr Fisher's collection of examples. Dr Parker and Dr Lawther produced two very useful list of corrections. Any credit for these notes belongs to them, any discredit to me. This is a course outline. A few proofs are included or sketched in these notes but most are omitted. Please note that vectors are *row* vectors unless otherwise stated.

Contents

1 Codes and alphabets

Originally, a code was a device for making messages hard to read. The study of such codes and their successors is called cryptography and will form the subject of the last quarter of these notes. However, in the 19th century the optical[1] and then the electrical telegraph made it possible to send messages speedily, but only after they had been translated from ordinary written English or French into a string of symbols.

The best known of the early codes is the Morse code used in electronic telegraphy. We think of it as consisting of dots and dashes but, in fact, it had three symbols dot, dash and pause which we write as $\bullet$, $-$ and $*$. Morse assigned a *code word* consisting of a sequence of symbols to each of the letters of the alphabet and each digit. Here are some typical examples.

$$\begin{array}{rrr} A \mapsto \bullet - * & B \mapsto - \bullet\bullet\bullet * & C \mapsto - \bullet - \bullet * \\ D \mapsto - \bullet\bullet * & E \mapsto \bullet * & F \mapsto \bullet\bullet - \bullet * \\ O \mapsto - - - * & S \mapsto \bullet\bullet\bullet * & 7 \mapsto - - \bullet\bullet\bullet * \end{array}$$

The symbols of the original message would be *encoded* and the code words sent in sequence, as in

$$SOS \mapsto \bullet\bullet\bullet * - - - * \bullet\bullet\bullet *,$$

and then decoded in sequence at the other end to recreate the original message.

Exercise 1.1. *Decode* $- \bullet - \bullet * - - - * - \bullet\bullet * \bullet *$.

[1]See *The Count of Monte Cristo* and various Napoleonic sea stories. A statue to the inventor of the optical telegraph (semaphore) was put up in Paris in 1893 but melted down during World War II and not replaced (http://hamradio.nikhef.nl/tech/rtty/chappe/). In the parallel universe of Disc World the *clacks* is one of the wonders of the Century of the Anchovy.

Morse's system was intended for human beings. Once machines took over the business of encoding, other systems developed. A very influential one called ASCII was developed in the 1960s. This uses two symbols 0 and 1 and all code words have seven symbols. In principle, this would give 128 possibilities, but 0000000 and 1111111 are not used, so there are 126 code words allowing the original message to contain a greater variety of symbols than Morse code. Here are some typical examples

$$\begin{array}{rcr} A \mapsto 1000001 & B \mapsto 1000010 & C \mapsto 1000011 \\ a \mapsto 1100001 & b \mapsto 1100010 & c \mapsto 1100011 \\ + \mapsto 0101011 & ! \mapsto 0100001 & 7 \mapsto 0110111 \end{array}$$

Exercise 1.2. *Encode* $b7!$. *Decode* 110001111000011100010.

More generally, we have two alphabets $\mathcal{A}$ and $\mathcal{B}$ and a coding function $c : \mathcal{A} \to \mathcal{B}^*$ where $\mathcal{B}^*$ consists of all finite sequences of elements of $\mathcal{B}$. If $\mathcal{A}^*$ consists of all finite sequences of elements of $\mathcal{A}$, then the encoding function $c^* : \mathcal{A}^* \to \mathcal{B}^*$ is given by

$$c^*(a_1a_2 \dots a_n) = c(a_1)c(a_2)\dots c(a_n).$$

We demand that c^* is injective, since otherwise it is possible to produce two messages which become indistinguishable once encoded.

We call codes for which c^* is injective *decodable*.

For many purposes, we are more interested in the collection of code words $\mathcal{C} = c(\mathcal{A})$ than the coding function c. If we look at the code words of Morse code and the ASCII code, we observe a very important difference. All the code words in ASCII have the same length (so we have a *fixed length* code), but this is not true for the Morse code (so we have a *variable length* code).

Exercise 1.3. *Explain why (if c is injective) any fixed length code is decodable.*

A variable length code need not be decodable even if c is injective.

Exercise 1.4. *(i) Let $\mathcal{A} = \mathcal{B} = \{0, 1\}$. If $c(0) = 0$, $c(1) = 00$ show that c is injective but c^* is not.*

(ii) Let $\mathcal{A} = \{1, 2, 3, 4, 5, 6\}$ and $\mathcal{B} = \{0, 1\}$. Show that there is a variable length coding c such that c is injective and all code words have length 2 or less. Show that there is no decodable coding c such that all code words have length 2 or less.

However, there is a family of variable length codes which are decodable in a natural way.

Definition 1.5. *Let $\mathcal{B}$ be an alphabet. We say that a finite subset $\mathcal{C}$ of $\mathcal{B}^*$ is* prefix-free *if, whenever $w \in \mathcal{C}$ is an initial sequence of $w' \in \mathcal{C}$, then $w = w'$. If $c : \mathcal{A} \to \mathcal{B}^*$ is a coding function, we say that c is prefix-free if c is injective and $c(\mathcal{A})$ is prefix-free.*

If c is prefix-free, then, not only is c^* injective, but we can decode messages on the fly. Suppose that we receive a sequence b_1, b_2, ... The moment we have received some $c(a_1)$, we know that the first message was a_1 and we can proceed to look for the second message. (For this reason prefix-free codes are sometimes called instantaneous codes or self punctuation codes.)

Exercise 1.6. *Let $\mathcal{A} = \{0, 1, 2, 3\}$, $\mathcal{B} = \{0, 1\}$. If $c, \tilde{c} : \mathcal{A} \to \mathcal{B}^*$ are given by*

$$\begin{aligned} c(0) &= 0 & \tilde{c}(0) &= 0 \\ c(1) &= 10 & \tilde{c}(1) &= 01 \\ c(2) &= 110 & \tilde{c}(2) &= 011 \\ c(3) &= 111 & \tilde{c}(3) &= 111 \end{aligned}$$

show that c is prefix-free, but $\tilde{c}$ is not. By thinking about the way $\tilde{c}$ is obtained from c, or otherwise, show that $\tilde{c}^*$ is injective.

Exercise 1.7. *Why is every injective fixed length code automatically prefix-free?*

From now on, unless explicitly stated otherwise, c will be injective and the codes used will be prefix-free. In section 3 we show that we lose nothing by confining ourselves to prefix-free codes.

2 Huffman's algorithm

An electric telegraph is expensive to build and maintain. However good a telegraphist was, he could only send or receive a limited number of dots and dashes each minute. (This is why Morse chose a variable length code. The telegraphist would need to send the letter E far more often than the letter Q so Morse gave E the short code $\bullet *$ and Q the long code $- - \bullet - *$.) It is possible to increase the rate at which symbols are sent and received by using machines, but the laws of physics (backed up by results in Fourier analysis) place limits on the number of symbols that can be correctly transmitted over a given line. (The slowest rates were associated with undersea cables.)

Customers were therefore charged so much a letter or, more usually, so much a word[2] (with a limit on the permitted word length). Obviously it made

[2]Leading to a prose style known as telegraphese. 'Arrived Venice. Streets flooded. Advise.'

sense to have books of 'telegraph codes' in which one five letter combination, say, 'FTCGI' meant 'are you willing to split the difference?' and another 'FTCSU' meant 'cannot see any difference'[3].

Today messages are usually sent in as binary sequences like $01110010\ldots$, but the transmission of each digit still costs money. If we know that there are n possible messages that can be sent and that $n \leq 2^m$, then we can assign each message a different string of m zeros and ones (usually called *bits*) and each message will cost mK cents where K is the cost of sending one bit.

However, this may not be the best way of saving money. If, as often happens, one message (such as 'nothing to report') is much more frequent than any other then it may be cheaper *on average* to assign it a shorter code word even at the cost of lengthening the other code words.

Problem 2.1. *Given n messages M_1, M_2, $\ldots$, M_n such that the probability that M_j will be chosen is p_j, find distinct code words C_j consisting of l_j bits so that the expected cost*

$$K\sum_{j=1}^{n} p_j l_j$$

of sending the code word corresponding to the chosen message is minimised.

Of course, we suppose $K > 0$.

The problem is interesting as it stands, but we have not taken into account the fact that a variable length code may not be decodable. To deal with this problem we add an extra constraint.

Problem 2.2. *Given n messages M_1, M_2, $\ldots$, M_n such that the probability that M_j will be chosen is p_j, find a prefix-free collection of code words C_j consisting of l_j bits so that the expected cost*

$$K\sum_{j=1}^{n} p_j l_j$$

of sending the code word corresponding to the chosen message is minimised.

In 1951 Huffman was asked to write an essay on this problem as an end of term university exam. Instead of writing about the problem, he solved it completely.

[3]If the telegraph company insisted on ordinary words you got codes like 'FLIRT' for 'quality of crop good'. Google 'telegraphic codes and message practice, 1870-1945' for lots of examples.

Theorem 2.3. **[Huffman's algorithm]** *The following algorithm solves Problem 2.2 with n messages. Order the messages so that $p_1 \geq p_2 \geq \cdots \geq p_n$. Solve the problem with $n-1$ messages M'_1, M'_2, ..., M'_{n-1} such that M'_j has probability p_j for $1 \leq j \leq n-2$, but M'_{n-1} has probability $p_{n-1}+p_n$. If C'_j is the code word corresponding to M'_j, the original problem is solved by assigning M_j the code word C'_j for $1 \leq j \leq n-2$ and M_{n-1} the code word consisting of C'_{n-1} followed by 0 and M_n the code word consisting of C'_{n-1} followed by 1.*

Since the problem is trivial when $n = 2$ (give M_1 the code word 0 and M_2 the code word 1) this gives us what computer programmers and logicians call a *recursive solution.*

Recursive programs are often better adapted to machines than human beings, but it is very easy to follow the steps of Huffman's algorithm 'by hand'. (Note that the algorithm is very specific about the labelling of the code words so that, for example, message 1

Example 2.4. *Suppose $n = 4$, M_j has probability $j/10$ for $1 \leq j \leq 4$. Apply Huffman's algorithm.*

Solution. (Note that we do not bother to reorder messages.) Combining messages in the suggested way, we get

$$\begin{gathered} 1, 2, 3, 4 \\ [1, 2], 3, 4 \\ [[1, 2], 3], 4. \end{gathered}$$

Working backwards, we get

$$\begin{gathered} C_{[[1,2],3]} = 0\ldots,\ C_4 = 1 \\ C_{[1,2]} = 01\ldots,\ C_3 = 00 \\ C_1 = 011,\ C_2 = 010. \end{gathered}$$

□

The reader is strongly advised to do a slightly more complicated example like the next.

Exercise 2.5. *Suppose M_j has probability $j/45$ for $1 \leq j \leq 9$. Apply Huffman's algorithm.*

As we indicated earlier, the effects of Huffman's algorithm will be most marked when a few messages are highly probable.

Exercise 2.6. *Suppose* $n = 64$*,* M_1 *has probability* $1/2$*,* M_2 *has probability* $1/4$ *and* M_j *has probability* $1/248$ *for* $3 \leq j \leq 64$*. Explain why, if we use code words of equal length then the length of a code word must be at least* 6*. By using the ideas of Huffman's algorithm (you should not need to go through all the steps) obtain a set of code words such that the* expected *length of a code word sent is not more than* 3*.*

Whilst doing the exercises the reader must already have been struck by the fact that minor variations in the algorithm produce different codes. (Note, for example that, if we have a Huffman code, then interchanging the role of 0 and 1 will produce another Huffman type code.) In fact, although the Huffman algorithm will always produce a best code (in the sense of Problem 2.2), there may be other equally good codes which could not be obtained in this manner.

Exercise 2.7. *Suppose* $n = 4$*,* M_1 *has probability* $.23$*,* M_2 *has probability* $.24$*,* M_3 *has probability* $.26$ *and* M_4 *has probability* $.27$*. Show that any assignment of the code words* 00*,* 01*,* 10 *and* 11 *produces a best code in the sense of Problem 2.2.*

The fact that the Huffman code may not be the unique best solution means that we need to approach the proof of Theorem 2.3 with caution. We observe that reading a code word from a prefix-free code is like climbing a tree with 0 telling us to take the left branch and 1 the right branch. The fact that the code is prefix-free tells us that each code word may be represented by a leaf at the end of a final branch. Thus, for example, the code word 00101 is represented by the leaf found by following left branch, left branch, right branch, left branch, right branch. The next lemma contains the essence of our proof of Theorem 2.3.

Lemma 2.8. *(i) If we have a best code then it will split into a left branch and right branch at every stage.*

(ii) If we label every branch by the sum of the probabilities of all the leaves that spring from it then, if we have a best code, every branch belonging to a particular stage of growth will have at least as large a number associated with it as any branch belonging to a later stage.

(iii) If we have a best code then interchanging the probabilities of leaves belonging to the last stage (ie the longest code words) still gives a best code.

(iv) If we have a best code then two of the leaves with the lowest probabilities will appear at the last stage.

(v) There is a best code in which two of the leaves with the lowest probabilities are neighbours (have code words differing only in the last place).

In order to use the Huffman algorithm we need to know the probabilities of the n possible messages. Suppose we do not. After we have sent k messages we will know that message M_j has been sent k_j times and so will the recipient of the message. If we decide to use a Huffman code for the next message, it is not unreasonable (lifting our hat in the direction of the Reverend Thomas Bayes) to take

$$p_j = \frac{k_j + 1}{k + n}.$$

Provided the recipient knows the exact version of the Huffman algorithm that we use, she can reconstruct our Huffman code and decode our next message. Variants of this idea are known as 'Huffman-on-the-fly' and form the basis of the kind of compression programs used in your computer. Notice however, that whilst Theorem 2.3 is an examinable theorem, the contents of this paragraph form a non-examinable plausible statement.

3 More on prefix-free codes

It might be thought that Huffman's algorithm says all that is to be said on the problem it addresses. However, there are two important points that need to be considered. The first is whether we could get better results by using codes which are not prefix-free. The object of this section is to show that this is not the case.

As in section 1, we consider two alphabets $\mathcal{A}$ and $\mathcal{B}$ and a coding function $c : \mathcal{A} \to \mathcal{B}^*$ (where, as we said earlier, $\mathcal{B}^*$ consists of all finite sequences of elements of $\mathcal{B}$). For most of this course $\mathcal{B} = \{0, 1\}$, but in this section we allow $\mathcal{B}$ to have D elements. The elements of $\mathcal{B}^*$ are called *words*.

Lemma 3.1. [Kraft's inequality 1] *If a prefix-free code $\mathcal{C}$ consists of n words C_j of length l_j, then*

$$\sum_{j=1}^{n} D^{-l_j} \leq 1.$$

Lemma 3.2. [Kraft's inequality 2] *Given strictly positive integers l_j satisfying*

$$\sum_{j=1}^{n} D^{-l_j} \leq 1,$$

we can find a prefix-free code $\mathcal{C}$ consisting of n words C_j of length l_j.

Proof. Take $l_1 \leq l_2 \leq \cdots \leq l_n$. We give an inductive construction for an appropriate prefix-free code. Start by choosing C_1 to be any code word of length l_1.

Suppose that we have found a collection of r prefix-free code words C_k of length l_k $[1 \leq k \leq r]$. If $r = n$ we are done. If not, consider all possible code words of length l_{r+1}. Of these $D^{l_{r+1}-l_k}$ will have prefix C_k so at most (in fact, exactly)

$$\sum_{k=1}^{r} D^{l_{r+1}-l_k}$$

will have one of the code words already selected as prefix. By hypothesis

$$\sum_{k=1}^{r} D^{l_{r+1}-l_k} = D^{l_{r+1}} \sum_{k=1}^{r} D^{-l_k} < D^{l_{r+1}}.$$

Since there are $D^{l_{r+1}}$ possible code words of length l_{r+1} there is at least one 'good code word' which does not have one of the code words already selected as prefix. Choose one of the good code words as C_{r+1} and restart the induction. □

The method used in the proof is called a 'greedy algorithm' because we just try to do the best we can at each stage without considering future consequences.

Lemma 3.1 is pretty but not deep. MacMillan showed that the same inequality applies to all decodable codes. The proof is extremely elegant and (after one has thought about it long enough) natural.

Theorem 3.3. **[The MacMillan inequality]** *If a decodable code $\mathcal{C}$ consists of n words C_j of length l_j, then*

$$\sum_{j=1}^{n} D^{-l_j} \leq 1.$$

Using Lemma 3.2 we get the immediate corollary.

Lemma 3.4. *If there exists a decodable code $\mathcal{C}$ consisting of n words C_j of length l_j, then there exists a prefix-free code $\mathcal{C}'$ consisting of n words C'_j of length l_j.*

Thus if we are only concerned with the length of code words we need only consider prefix-free codes.

4 Shannon's noiseless coding theorem

In the previous section we indicated that there was a second question we should ask about Huffman's algorithm. We know that Huffman's algorithm is best possible, but we have not discussed how good the best possible should be.

Let us restate our problem. (In this section we allow the coding alphabet $\mathcal{B}$ to have D elements.)

Problem 4.1. *Given n messages M_1, M_2, …, M_n such that the probability that M_j will be chosen is p_j, find a decodable code $\mathcal{C}$ whose code words C_j consist of l_j bits so that the expected cost*

$$K\sum_{j=1}^{n} p_j l_j$$

of sending the code word corresponding to the chosen message is minimised.

In view of Lemma 3.2 (any system of lengths satisfying Kraft's inequality is associated with a prefix-free and so decodable code) and Theorem 3.3 (any decodable code satisfies Kraft's inequality), Problem 4.1 reduces to an abstract minimising problem.

Problem 4.2. *Suppose $p_j \geq 0$ for $1 \leq j \leq n$ and $\sum_{j=1}^{n} p_j = 1$. Find strictly positive integers l_j minimising*

$$\sum_{j=1}^{n} p_j l_j \text{ subject to } \sum_{j=1}^{n} D^{-l_j} \leq 1.$$

Problem 4.2 is hard because we restrict the l_j to be integers. If we drop the restriction we end up with a problem in Part IB variational calculus.

Problem 4.3. *Suppose $p_j \geq 0$ for $1 \leq j \leq n$ and $\sum_{j=1}^{n} p_j = 1$. Find strictly positive real numbers x_j minimising*

$$\sum_{j=1}^{n} p_j x_j \text{ subject to } \sum_{j=1}^{n} D^{-x_j} \leq 1.$$

Calculus solution. Observe that decreasing any x_k decreases $\sum_{j=1}^{n} p_j x_j$ and increases $\sum_{j=1}^{n} D^{-x_j}$. Thus we may demand

$$\sum_{j=1}^{n} D^{-x_j} = 1.$$

The Lagrangian is

$$L(\mathbf{x}, \lambda) = \sum_{j=1}^{n} p_j x_j - \lambda \sum_{j=1}^{n} D^{-x_j}.$$

Since

$$\frac{\partial L}{\partial x_j} = p_j + (\lambda \log D) D^{-x_j}$$

we know that, at any stationary point,

$$D^{-x_j} = K_0(\lambda) p_j$$

for some $K_0(\lambda) > 0$. Since $\sum_{j=1}^{n} D^{-x_j} = 1$, our original problem will have a *stationarising* solution when

$$D^{-x_j} = p_j, \text{ that is to say } x_j = -\frac{\log p_j}{\log D}$$

and

$$\sum_{j=1}^{n} p_j x_j = -\sum_{j=1}^{n} p_j \frac{\log p_j}{\log D}.$$

□

It is not hard to convince oneself that the stationarising solution just found is, in fact, maximising, but it is an unfortunate fact that IB variational calculus is suggestive rather than conclusive.

The next two exercises (which will be done in lectures and form part of the course) provide a rigorous proof.

Exercise 4.4. *(i) Show that*

$$\log t \leq t - 1$$

for $t > 0$ with equality if and only if $t = 1$.

(ii) **[Gibbs' inequality]** *Suppose that $p_j, q_j > 0$ and*

$$\sum_{j=1}^{n} p_j = \sum_{j=1}^{n} q_j = 1.$$

By applying (i) with $t = q_j/p_j$, show that

$$\sum_{j=1}^{n} p_j \log p_j \geq \sum_{j=1}^{n} p_j \log q_j$$

with equality if and only if $p_j = q_j$.

Exercise 4.5. *We use the notation of Problem 4.3.*

(i) Show that, if $x_j^* = -\log p_j/\log D$*, then* $x_j^* > 0$ *and*

$$\sum_{j=1}^{n} D^{-x_j^*} = 1.$$

(ii) Suppose that $y_j > 0$ *and*

$$\sum_{j=1}^{n} D^{-y_j} = 1.$$

Set $q_j = D^{-y_j}$*. By using Gibbs' inequality from Exercise 4.4 (ii), show that*

$$\sum_{j=1}^{n} p_j x_j^* \leq \sum_{j=1}^{n} p_j y_j$$

with equality if and only if $y_j = x_j^*$ *for all* j*.*

Analysts use logarithms to the base e, but the importance of two-symbol alphabets means that communication theorists often use logarithms to the base 2.

Exercise 4.6. *(Memory jogger.) Let* a, $b > 0$*. Show that*

$$\log_a b = \frac{\log b}{\log a}.$$

The result of Problem 4.3 is so important that it gives rise to a definition.

Definition 4.7. *Let* $\mathcal{A}$ *be a non-empty finite set and* A *a random variable taking values in* $\mathcal{A}$*. If* A *takes the value* a *with probability* p_a *we say that the system has* Shannon entropy[4] *(or* information entropy*)*

$$H(A) = -\sum_{a\in\mathcal{A}} p_a \log_2 p_a.$$

Theorem 4.8. *Let* $\mathcal{A}$ *and* $\mathcal{B}$ *be finite alphabets and let* $\mathcal{B}$ *have* D *symbols. If* A *is an* $\mathcal{A}$*-valued random variable, then any decodable code* $c : \mathcal{A} \to \mathcal{B}$ *must satisfy*

$$\mathbb{E}|c(A)| \geq \frac{H(A)}{\log_2 D}.$$

[4] It is unwise for the beginner and may or may not be fruitless for the expert to seek a link with entropy in physics.

Here $|c(A)|$ denotes the length of $c(A)$. Notice that the result takes a particularly simple form when $D = 2$.

In Problem 4.3 the x_j are just positive real numbers but in Problem 4.2 the d_j are integers. Choosing d_j as close as possible to the best x_j may not give the best d_j, but it is certainly worth a try.

Theorem 4.9. **[Shannon–Fano encoding]** *Let $\mathcal{A}$ and $\mathcal{B}$ be finite alphabets and let $\mathcal{B}$ have D symbols. If A is an $\mathcal{A}$-valued random variable, then there exists a prefix-free (so decodable) code $c : \mathcal{A} \to \mathcal{B}$ which satisfies*

$$\mathbb{E}|c(A)| \leq 1 + \frac{H(A)}{\log_2 D}.$$

Proof. By Lemma 3.2 (which states that given lengths satisfying Kraft's inequality we can construct an associated prefix-free code), it suffices to find strictly positive integers l_a such that

$$\sum_{a \in \mathcal{A}} D^{-l_a} \leq 1, \text{ but } \sum_{a \in \mathcal{A}} p_a l_a \leq 1 + \frac{H(A)}{\log_2 D}.$$

If we take

$$l_a = \lceil -\log_D p_a \rceil,$$

that is to say, we take l_a to be the smallest integer no smaller than $-\log_D p_a$, then these conditions are satisfied and we are done. □

It is very easy to use the method just indicated to find an appropriate code. (Such codes are called Shannon–Fano codes[5]. Fano was the professor who set the homework for Huffman. The point of view adopted here means that for some problems there may be more than one Shannon–Fano code.)

Exercise 4.10. *(i) Let $\mathcal{A} = \{1, 2, 3, 4\}$. Suppose that the probability that letter k is chosen is $k/10$. Observing that $\log 2^{-n} = -n$ write down an appropriate Shannon–Fano code c.*

(ii) We found a Huffman code c_h for the system in Example 2.4. Show[6] that the entropy is approximately 1.85, that $\mathbb{E}|c(A)| = 2.4$ and that $\mathbb{E}|c_h(A)| = 1.9$. Check that these results are consistent with our previous theorems.

Putting Theorems 4.8 and Theorem 4.9 together, we get the following remarkable result.

[5] Wikipedia and several other sources give a definition of Shannon–Fano codes which is definitely *inconsistent* with that given here. Within a Cambridge examination context you may assume that Shannon–Fano codes are those considered here.

[6] Unless you are on a desert island in which case the calculations are rather tedious.

Theorem 4.11. [Shannon's noiseless coding theorem] *Let $\mathcal{A}$ and $\mathcal{B}$ be finite alphabets and let $\mathcal{B}$ have D symbols. If A is a $\mathcal{A}$-valued random variable, then any decodable code c which minimises $\mathbb{E}|c(A)|$ satisfies*

$$\frac{H(A)}{\log_2 D} \leq \mathbb{E}|c(A)| \leq 1 + \frac{H(A)}{\log_2 D}.$$

In particular, Huffman's code c_h for two symbols satisfies

$$H(A) \leq \mathbb{E}|c_h(A)| \leq 1 + H(A).$$

Exercise 4.12. *(i) Sketch $h(t) = -t \log t$ for $0 \leq t \leq 1$. (We define $h(0) = 0$.)*

(ii) Let

$$\Gamma = \left\{ \mathbf{p} \in \mathbb{R}^n \ : \ p_j \geq 0, \ \sum_{j=1}^{n} p_j = 1 \right\}$$

and let $H : \Gamma \to \mathbb{R}$ be defined by

$$H(\mathbf{p}) = \sum_{j=1}^{n} h(p_j).$$

Find the maximum and minimum of H and describe the points where these values are attained.

(iii) If $n = 2^r + s$ with $0 \leq s < 2^r$ and $p_j = 1/n$, describe the Huffman code c_h for two symbols and verify directly that (with notation of Theorem 4.11)

$$H(A) \leq \mathbb{E}|c_h(A)| \leq 1 + H(A).$$

Waving our hands about wildly, we may say that 'A system with low Shannon entropy is highly organised and, knowing the system, it is usually quite easy to identify an individual from the system'.

Exercise 4.13. *The notorious Trinity gang has just been rounded up and Trubshaw of the Yard wishes to identify the leader (or Master, as he is called). Sam the Snitch makes the following offer. Presented with any collection of members of the gang he will (by a slight twitch of his left ear) indicate if the Master is among them. However, in view of the danger involved, he demands ten pounds for each such encounter. Trubshaw believes that the probability of the jth member of the gang being the Master is p_j $[1 \leq j \leq n]$ and wishes to minimise the expected drain on the public purse. Advise him.*

5 Non-independence

(This section is non-examinable.)

In the previous sections we discussed codes $c : \mathcal{A} \to \mathcal{B}^*$ such that, if a letter $A \in \mathcal{A}$ was chosen according to some random law, $\mathbb{E}|c(A)|$ was about as small as possible. If we choose A_1, A_2, ... independently according to the same law, then it is not hard to convince oneself that

$$\mathbb{E}|c^*(A_1A_2A_3 \dots A_n)| = n\mathbb{E}|c(A)|$$

will be as small as possible.

However, in re*l lif* th* let*ers a*e often no* i*d*p***ent. It is sometimes possible to send messages more efficiently using this fact.

Exercise 5.1. *Suppose that we have a sequence X_j of random variables taking the values 0 and 1. Suppose that $X_1 = 1$ with probability $1/2$ and $X_{j+1} = X_j$ with probability $.99$ independent of what has gone before.*

(i) Suppose we wish to send ten successive bits $X_jX_{j+1} \dots X_{j+9}$. Show that if we associate the sequence of ten zeros with 0, the sequence of ten ones with 10 and any other sequence $a_0a_1 \dots a_9$ with $11a_0a_1 \dots a_9$ we have a decodable code which on average requires about $5/2$ bits to transmit the sequence.

(ii) Suppose we wish to send the bits $X_jX_{j+10^6}X_{j+2\times 10^6} \dots X_{j+9\times 10^6}$. Explain why any decodable code will require on average at least 10 bits to transmit the sequence. (You need not do detailed computations.)

If we transmit sequences of letters by forming them into longer words and coding the words, we say we have a block code. It is plausible that the longer the blocks, the less important the effects of non-independence. In more advanced courses it is shown how to define entropy for systems like the one discussed in Exercise 5.1 (that is to say Markov chains) and that, provided we take long enough blocks, we can recover an analogue of Theorem 4.11 (the noiseless coding theorem).

In the real world, the problem lies deeper. Presented with a photograph, we can instantly see that it represents Lena wearing a hat. If a machine reads the image pixel by pixel, it will have great difficulty recognising much, apart from the fact that the distribution of pixels is 'non-random' or has 'low entropy' (to use the appropriate hand-waving expressions). Clearly, it ought to be possible to describe the photograph with many fewer bits than are required to describe each pixel separately, but, equally clearly, a method that works well on black and white photographs may fail on colour photographs and a method that works well on photographs of faces may work badly when applied to photographs of trees.

Engineers have a clever way of dealing with this problem. Suppose we have a sequence x_j of zeros and ones produced by some random process. Someone who believes that they partially understand the nature of the process builds us a prediction machine which, given the sequence x_1, x_2, $\dots x_j$ so far, predicts the next term will be x'_{j+1}. Now set

$$y_{j+1} \equiv x_{j+1} - x'_{j+1} \mod 2.$$

If we are given the sequence y_1, y_2, $\dots$ we can recover the x_j inductively using the prediction machine and the formula

$$x_{j+1} \equiv y_{j+1} + x'_{j+1} \mod 2.$$

If the prediction machine is good, then the sequence of y_j will consist mainly of zeros and there will be many ways of encoding the sequence as (on average) a much shorter code word. (For example, if we arrange the sequence in blocks of fixed length, many of the possible blocks will have very low probability, so Huffman's algorithm will be very effective.)

Build a better mousetrap, and the world will beat a path to your door. Build a better prediction machine and the world will beat your door down.

There is a further real world complication. Engineers distinguish between irreversible 'lossy compression' and reversible 'lossless compression'. For compact discs, where bits are cheap, the sound recorded can be reconstructed exactly. For digital sound broadcasting, where bits are expensive, the engineers make use of knowledge of the human auditory system (for example, the fact that we can not make out very soft noise in the presence of loud noises) to produce a result that might sound perfect (or nearly so) to us, but which is, in fact, not. For mobile phones, there can be greater loss of data because users do not demand anywhere close to perfection. For digital TV, the situation is still more striking with reduction in data content from film to TV of anything up to a factor of 60. However, medical and satellite pictures must be transmitted with no loss of data. Notice that lossless coding can be judged by absolute criteria, but the merits of lossy coding can only be judged subjectively.

Ideally, lossless compression should lead to a signal indistinguishable (from a statistical point of view) from a random signal in which the value of each bit is independent of the value of all the others. In practice, this is only possible in certain applications. As an indication of the kind of problem involved, consider TV pictures. If we know that what is going to be transmitted is 'head and shoulders' or 'tennis matches' or 'cartoons' it is possible to obtain extraordinary compression ratios by 'tuning' the compression method to the expected pictures, but then changes from what is expected can be disastrous. At present, digital TV encoders merely expect the picture to consist of blocks which move at nearly constant velocity remaining more or less unchanged from frame to frame[7]. In this, as in other applications, we know that after compression the signal still has non-trivial statistical properties, but we do not know enough about them to exploit them.

6 What is an error correcting code?

In the introductory Section 1, we discussed 'telegraph codes' in which one five letter combination 'QWADR', say, meant 'please book quiet room for two' and another 'QWNDR' meant 'please book cheapest room for one'. Obviously, also, an error of one letter in this code could have unpleasant consequences[8].

Today, we transmit and store long strings of binary sequences, but face the same problem that some digits may not be transmitted or stored correctly. We suppose that the string is the result of data compression and so, as we said at the end of the last section, *although the string may have non-trivial statistical properties, we do not know enough to exploit this fact.* (If we knew how to exploit any statistical regularity, we could build a prediction device and compress the data still further.) Because of this, we shall assume that we are asked to consider a collection of m messages *each of which is equally likely.*

Our model is the following. When the 'source' produces one of the m possible messages μ_i say, it is fed into a 'coder' which outputs a string $\mathbf{c}_i$ of n binary digits. The string is then transmitted one digit at a time along a 'communication channel'. Each digit has probability p of being mistransmitted (so that 0 becomes 1 or 1 becomes 0) independently of what happens to the other digits $[0 \leq p < 1/2]$. The transmitted message is then passed through a 'decoder' which either produces a message μ_j (where we hope that $j = i$) or an error message and passes it on to the 'receiver'. The technical term for our model is the *binary symmetric channel* (binary because we use two symbols, symmetric because the probability of error is the same whichever symbol we use).

Exercise 6.1. *Why do we not consider the case* $1 \geq p > 1/2$*? What if* $p = 1/2$*?*

[7] Watch what happens when things go wrong.

[8] This is a made up example, since compilers of such codes understood the problem.

For most of the time we shall concentrate our attention on a *code* $C \subseteq \{0,1\}^n$ consisting of the *codewords* $\mathbf{c}_i$. (Thus we use a fixed length code.) We say that C has *size* $m = |C|$. If m is large then we can send a large number of possible messages (that is to say, we can send more information) but, as m increases, it becomes harder to distinguish between different messages when errors occur. At one extreme, if $m = 1$, errors cause us no problems (since there is only one message) but no information is transmitted (since there is only one message). At the other extreme, if $m = 2^n$, we can transmit lots of messages but any error moves us from one codeword to another. We are led to the following rather natural definition.

Definition 6.2. *The information rate of C is* $\dfrac{\log_2 m}{n}$.

Note that, since $m \leq 2^n$ the information rate is never greater than 1. Notice also that the values of the information rate when $m = 1$ and $m = 2^n$ agree with what we might expect.

How should our decoder work? We have assumed that all messages are equally likely and that errors are independent (this would not be true if, for example, errors occurred in bursts[9]).

Under these assumptions, a reasonable strategy for our decoder is to guess that the codeword sent is one which differs in the fewest places from the string of n binary digits received. Here and elsewhere the discussion can be illuminated by the simple notion of a Hamming distance.

Definition 6.3. *If* $\mathbf{x},\ \mathbf{y} \in \{0,1\}^n$, *we write*

$$d(\mathbf{x},\mathbf{y}) = \sum_{j=1}^{n} |x_j - y_j|$$

and call $d(\mathbf{x},\mathbf{y})$ *the Hamming distance between* $\mathbf{x}$ *and* $\mathbf{y}$.

Lemma 6.4. *The Hamming distance is a metric.*

[9]For the purposes of this course we note that this problem could be tackled by permuting the 'bits' of the message so that 'bursts are spread out'. In theory, we could do better than this by using the statistical properties of such bursts to build a prediction machine. In practice, this is rarely possible. In the paradigm case of mobile phones, the properties of the transmission channel are constantly changing and are not well understood. (Here the main restriction on the use of permutation is that it introduces time delays. One way round this is 'frequency hopping' in which several users constantly swap transmission channels 'dividing bursts among users'.) One desirable property of codes for mobile phone users is that they should 'fail gracefully', so that as the error rate for the channel rises the error rate for the receiver should not suddenly explode.

We now do some very simple IA probability.

Lemma 6.5. *We work with the coding and transmission scheme described above. Let* $\mathbf{c} \in C$ *and* $\mathbf{x} \in \{0,1\}^n$.

(i) If $d(\mathbf{c}, \mathbf{x}) = r$*, then*

$$\Pr(\mathbf{x} \textit{ received given } \mathbf{c} \textit{ sent}) = p^r(1-p)^{n-r}.$$

(ii) If $d(\mathbf{c}, \mathbf{x}) = r$*, then*

$$\Pr(\mathbf{c} \textit{ sent given } \mathbf{x} \textit{ received}) = A(\mathbf{x})p^r(1-p)^{n-r},$$

where $A(\mathbf{x})$ *does not depend on* r *or* $\mathbf{c}$.

(iii) If $\mathbf{c}' \in C$ *and* $d(\mathbf{c}', \mathbf{x}) \geq d(\mathbf{c}, \mathbf{x})$*, then*

$$\Pr(\mathbf{c} \textit{ sent given } \mathbf{x} \textit{ received}) \geq \Pr(\mathbf{c}' \textit{ sent given } \mathbf{x} \textit{ received}),$$

with equality if and only if $d(\mathbf{c}', \mathbf{x}) = d(\mathbf{c}, \mathbf{x})$.

This lemma justifies our use, both explicit and implicit, throughout what follows of the so-called *maximum likelihood* decoding rule.

Definition 6.6. *The maximum likelihood decoding rule states that a string* $\mathbf{x} \in \{0,1\}^n$ *received by a decoder should be decoded as (one of) the codeword(s) at the smallest Hamming distance from* $\mathbf{x}$.

Notice that, although this decoding rule is mathematically attractive, it may be impractical if C is large and there is often no known way of finding the codeword at the smallest distance from a particular $\mathbf{x}$ in an acceptable number of steps. (We can always make a complete search through all the members of C but unless there are very special circumstances this is likely to involve an unacceptable amount of work.)

7 Hamming's breakthrough

Although we have used simple probabilistic arguments to justify it, the maximum likelihood decoding rule will often enable us to avoid probabilistic considerations (though not in the very important part of this concerned with Shannon's noisy coding theorem) and concentrate on algebra and combinatorics. The spirit of most of the course is exemplified in the next two definitions.

Definition 7.1. *We say that* C *is* d error detecting *if changing up to* d *digits in a codeword never produces another codeword.*

Definition 7.2. *We say that C is e* error correcting *if knowing that a string of n binary digits differs from some codeword of C in at most e places we can deduce the codeword.*

Here are some simple schemes. Some of them use alphabets with more than two symbols but the principles remain the same.
Repetition coding of length n. We take codewords of the form

$$\mathbf{c} = (c, c, c, \dots, c)$$

with $c = 0$ or $c = 1$. The code C is $n - 1$ error detecting, and $\lfloor (n-1)/2 \rfloor$ error correcting. The maximum likelihood decoder chooses the symbol that occurs most often. (Here and elsewhere $\lfloor \alpha \rfloor$ is the largest integer $N \leq \alpha$ and $\lceil \alpha \rceil$ is the smallest integer $M \geq \alpha$.) Unfortunately, the information rate is $1/n$ which is rather low[10].
The Cambridge examination paper code Each candidate is asked to write down a Candidate Identifier of the form $1234A$, $1235B$, $1236C$, …(the eleven[11] possible letters are repeated cyclically) and a desk number. The first four numbers in the Candidate Identifier identify the candidate uniquely. If the letter written by the candidate does not correspond to to the first four numbers the candidate is identified by using the desk number.

Exercise 7.3. *Show that if the candidate makes one error in the Candidate Identifier, then this will be detected. Would this be true if there were* 9 *possible letters repeated cyclically? Would this be true if there were* 12 *possible letters repeated cyclically? Give reasons.*

Show that, if we also use the Desk Number then the combined code Candidate Number/Desk Number is one error correcting

The paper tape code. Here and elsewhere, it is convenient to give $\{0, 1\}$ the structure of the field $\mathbb{F}_2 = \mathbb{Z}_2$ by using arithmetic modulo 2. The codewords have the form

$$\mathbf{c} = (c_1, c_2, c_3, \dots, c_n)$$

with c_1, c_2, …, c_{n-1} freely chosen elements of $\mathbb{F}_2$ and c_n (the check digit) the element of $\mathbb{F}_2$ which gives

$$c_1 + c_2 + \dots + c_{n-1} + c_n = 0.$$

The resulting code C is 1 error detecting since, if $\mathbf{x} \in \mathbb{F}_2^n$ is obtained from $\mathbf{c} \in C$ by making a single error, we have

$$x_1 + x_2 + \dots + x_{n-1} + x_n = 1.$$

[10] Compare the chorus 'Oh no John, no John, no John, no'.
[11] My guess.

However it is not error correcting since, if

$$x_1 + x_2 + \cdots + x_{n-1} + x_n = 1,$$

there are n codewords $\mathbf{y}$ with Hamming distance $d(\mathbf{x}, \mathbf{y}) = 1$. The information rate is $(n-1)/n$. Traditional paper tape had 8 places per line each of which could have a punched hole or not, so $n = 8$.

Exercise 7.4. *If you look at the inner title page of almost any book published between 1970 and 2006 you will find its International Standard Book Number (ISBN). The ISBN uses single digits selected from 0, 1, ..., 8, 9 and X representing 10. Each ISBN consists of nine such digits a_1, a_2, ..., a_9 followed by a single check digit a_{10} chosen so that*

$$10a_1 + 9a_2 + \cdots + 2a_9 + a_{10} \equiv 0 \mod 11. \tag{*}$$

(In more sophisticated language, our code C consists of those elements $\mathbf{a} \in \mathbb{F}_{11}^{10}$ such that $\sum_{j=1}^{10}(11-j)a_j = 0$.)

(i) Find a couple of books[12] *and check that $(*)$ holds for their ISBNs*[13].

(ii) Show that $(*)$ will not work if you make a mistake in writing down one digit of an ISBN.

(iii) Show that $(*)$ may fail to detect two errors.

(iv) Show that $(*)$ will not work if you interchange two distinct adjacent digits (a transposition error).

(v) Does (iv) remain true if we replace 'adjacent' by 'different'?

Errors of type (ii) and (iv) are the most common in typing[14]. *In communication between publishers and booksellers, both sides are anxious that errors should be detected but would prefer the other side to query errors rather than to guess what the error might have been.*

(vi) After January 2007, the appropriate ISBN is a 13 digit number $x_1x_2\ldots x_{13}$ with each digit selected from $0, 1, \ldots, 8, 9$ and the check digit x_{13} computed by using the formula

$$x_{13} \equiv -(x_1 + 3x_2 + x_3 + 3x_4 + \cdots + x_{11} + 3x_{12}) \mod 10.$$

Show that we can detect single errors. Give an example to show that we cannot detect all transpositions.

[12] In case of difficulty, your college library may be of assistance.

[13] In fact, X is only used in the check digit place.

[14] Thus a syllabus for an earlier version of this course contained the rather charming misprint of 'snydrome' for 'syndrome'.

Hamming had access to an early electronic computer but was low down in the priority list of users. He would submit his programs encoded on paper tape to run over the weekend but often he would have his tape returned on Monday because the machine had detected an error in the tape. 'If the machine can detect an error' he asked himself 'why can the machine not correct it?' and he came up with the following scheme.
Hamming's original code. We work in $\mathbb{F}_2^7$. The codewords $\mathbf{c}$ are chosen to satisfy the three conditions

$$\begin{aligned} c_1 + c_3 + c_5 + c_7 &= 0 \\ c_2 + c_3 + c_6 + c_7 &= 0 \\ c_4 + c_5 + c_6 + c_7 &= 0. \end{aligned}$$

By inspection, we may choose c_3, c_5, c_6 and c_7 freely and then c_1, c_2 and c_4 are completely determined. The information rate is thus $4/7$.

Suppose that we receive the string $\mathbf{x} \in \mathbb{F}_2^7$. We form the *syndrome* $(z_1, z_2, z_4) \in \mathbb{F}_2^3$ given by

$$\begin{aligned} z_1 &= x_1 + x_3 + x_5 + x_7 \\ z_2 &= x_2 + x_3 + x_6 + x_7 \\ z_4 &= x_4 + x_5 + x_6 + x_7. \end{aligned}$$

If $\mathbf{x}$ is a codeword, then $(z_1, z_2, z_4) = (0, 0, 0)$. If $\mathbf{c}$ is a codeword and the Hamming distance $d(\mathbf{x}, \mathbf{c}) = 1$, then the place in which $\mathbf{x}$ differs from $\mathbf{c}$ is given by $z_1 + 2z_2 + 4z_4$ (using ordinary addition, not addition modulo 2) as may be easily checked using linearity and a case by case study of the seven binary sequences $\mathbf{x}$ containing one 1 and six 0s. The Hamming code is thus 1 error correcting.

Exercise 7.5. *Suppose we use eight hole tape with the standard paper tape code and the probability that an error occurs at a particular place on the tape (i.e. a hole occurs where it should not or fails to occur where it should) is* 10^{-4}. *A program requires about 10 000 lines of tape (each line containing eight places) using the paper tape code. Using the Poisson approximation, direct calculation (possible with a hand calculator but really no advance on the Poisson method), or otherwise, show that the probability that the tape will be accepted as error free by the decoder is less than .04%.*

Suppose now that we use the Hamming scheme (making no use of the last place in each line). Explain why the program requires about 17 500 lines of tape but that any particular line will be correctly decoded with probability about $1 - (21 \times 10^{-8})$ *and the probability that the entire program will be correctly decoded is better than 99.6%.*

Hamming's scheme is easy to implement. It took a little time for his company to realise what he had done[15] but they were soon trying to patent it. In retrospect, the idea of an error correcting code seems obvious (Hamming's scheme had actually been used as the basis of a Victorian party trick) and indeed two or three other people discovered it independently, but Hamming and his co-discoverers had done more than find a clever answer to a question. They had asked an entirely new question and opened a new field for mathematics and engineering.

The times were propitious for the development of the new field. Before 1940, error correcting codes would have been luxuries, solutions looking for problems, after 1950, with the rise of the computer and new communication technologies, they became necessities. Mathematicians and engineers returning from wartime duties in code breaking, code making and general communications problems were primed to grasp and extend the ideas. The mathematical engineer Claude Shannon may be considered the presiding genius of the new field.

The reader will observe that data compression shortens the length of our messages by removing redundancy and Hamming's scheme (like all error correcting codes) lengthens them by introducing redundancy. This is true, but data compression removes redundancy which we do not control and which is not useful to us and error correction coding then replaces it with carefully controlled redundancy which we can use.

The reader will also note an analogy with ordinary language. The idea of data compression is illustrated by the fact that many common words are short[16]. On the other hand the redund of ordin lang makes it poss to understa it even if we do no catch everyth that is said.

8 General considerations

How good can error correcting and error detecting[17] codes be? The following discussion is a natural development of the ideas we have already discussed. Later, in our discussion of Shannon's noisy coding theorem we shall see another and deeper way of looking at the question.

Definition 8.1. *The minimum distance d of a code is the smallest Hamming*

[15] Experienced engineers came away from working demonstrations muttering 'I still don't believe it'.

[16] Note how 'horseless carriage' becomes 'car' and 'telephone' becomes 'phone'.

[17] If the error rate is low and it is easy to ask for the message to be retransmitted, it may be cheaper to concentrate on error detection. If there is no possibility of retransmission (as in long term data storage), we have to concentrate on error correction.

distance between distinct code words.

We call a code of length n, size m and distance d an $[n, m, d]$ code. Less briefly, a set $C \subseteq \mathbb{F}_2^n$, with $|C| = m$ and

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

is called an $[n, m, d]$ code. By an $[n, m]$ code we shall simply mean a code of length n and size m.

Lemma 8.2. *A code of minimum distance d can detect $d-1$ errors*[18] *and correct $\lfloor \frac{d-1}{2} \rfloor$ errors. It cannot detect all sets of d errors and cannot correct all sets of $\lfloor \frac{d-1}{2} \rfloor + 1$ errors.*

It is natural, here and elsewhere, to make use of the geometrical insight provided by the (closed) Hamming ball

$$B(\mathbf{x}, r) = \{\mathbf{y} : d(\mathbf{x}, \mathbf{y}) \leq r\}.$$

Observe that

$$|B(\mathbf{x}, r)| = |B(\mathbf{0}, r)|$$

for all $\mathbf{x}$ and so, writing

$$V(n, r) = |B(\mathbf{0}, r)|,$$

we know that $V(n, r)$ is the number of points in any Hamming ball of radius r. A simple counting argument shows that

$$V(n, r) = \sum_{j=0}^{r} \binom{n}{j}.$$

Theorem 8.3. [Hamming's bound] *If a code C is e error correcting, then*

$$|C| \leq \frac{2^n}{V(n, e)}.$$

There is an obvious fascination (if not utility) in the search for codes which attain the exact Hamming bound.

Definition 8.4. *A code C of length n and size m which can correct e errors is called* perfect *if*

$$m = \frac{2^n}{V(n, e)}.$$

[18] This is not as useful as it looks when d is large. If we know that our message is likely to contain many errors, all that an error detecting code can do is confirm our expectations. Error detection is only useful when errors are unlikely.

Lemma 8.5. *Hamming's original code is a* $[7, 16, 3]$ *code. It is perfect.*

It may be worth remarking in this context that, if a code which can correct e errors is perfect (i.e. has a perfect packing of Hamming balls of radius e), then the decoder must invariably give the wrong answer when presented with $e + 1$ errors. We note also that, if (as will usually be the case) $2^n/V(n, e)$ is not an integer, no perfect e error correcting code can exist.

Exercise 8.6. *Even if* $2^n/V(n, e)$ *is an integer, no perfect code may exist.*

(i) Verify that

$$\frac{2^{90}}{V(90, 2)} = 2^{78}.$$

(ii) Suppose that C *is a perfect 2 error correcting code of length* 90 *and size* 2^{78}*. Explain why we may suppose without loss of generality that* $\mathbf{0} \in C$*.*

(iii) Let C *be as in (ii) with* $\mathbf{0} \in C$*. Consider the set*

$$X = \{\mathbf{x} \in \mathbb{F}_2^{90} : x_1 = 1,\ x_2 = 1,\ d(\mathbf{0}, \mathbf{x}) = 3\}.$$

Show that, corresponding to each $\mathbf{x} \in X$*, we can find a unique* $\mathbf{c}(\mathbf{x}) \in C$ *such that* $d(\mathbf{c}(\mathbf{x}), \mathbf{x}) = 2$*.*

(iv) Continuing with the argument of (iii), show that

$$d(\mathbf{c}(\mathbf{x}), \mathbf{0}) = 5$$

and that $c_i(\mathbf{x}) = 1$ *whenever* $x_i = 1$*. If* $\mathbf{y} \in X$*, find the number of solutions to the equation* $\mathbf{c}(\mathbf{x}) = \mathbf{c}(\mathbf{y})$ *with* $\mathbf{x} \in X$ *and, by considering the number of elements of* X*, obtain a contradiction.*

(v) Conclude that there is no perfect $[90, 2^{78}]$ *code.*

The result of Exercise 8.6 was obtained by Golay. Far more importantly, he found another case when $2^n/V(n, e)$ is an integer and there does exist an associated perfect code (the Golay code).

Exercise 8.7. *Show that* $V(23, 3)$ *is a power of* 2*.*

Unfortunately the proof that the Golay code is perfect is too long to be given in the course,

We obtained the Hamming bound, which places an upper bound on how good a code can be, by a *packing* argument. A *covering* argument gives us the GSV (Gilbert, Shannon, Varshamov) bound in the opposite direction. Let us write $A(n, d)$ for the size of the largest code with minimum distance d.

Theorem 8.8. [Gilbert, Shannon, Varshamov] *We have*

$$A(n,d) \geq \frac{2^n}{V(n,d-1)}.$$

Until recently there were no general explicit constructions for codes which achieved the GSV bound (i.e. codes whose minimum distance d satisfied the inequality $A(n,d)V(n,d-1) \geq 2^n$). Such a construction was finally found by Garcia and Stichtenoth by using 'Goppa' codes.

9 Some elementary probability

Engineers are, of course, interested in 'best codes' of length n for reasonably small values of n, but mathematicians are particularly interested in what happens as $n \to \infty$.

We recall some elementary probability.

Lemma 9.1. [Tchebychev's inequality] *If X is a bounded real valued random variable and $a > 0$, then*

$$\Pr(|X - \mathbb{E}X| \geq a) \leq \frac{\operatorname{var} X}{a^2}.$$

Theorem 9.2. [Weak law of large numbers] *If $X_1, X_2, \ldots$ is a sequence of independent identically distributed real valued bounded random variables and $a > 0$, then*

$$\Pr\left(\left|n^{-1}\sum_{j=1}^{n} X_j - \mathbb{E}X\right| \geq a\right) \to 0$$

as $N \to \infty$.

Applying the weak law of large numbers, we obtain the following important result.

Lemma 9.3. *Consider the model of a noisy transmission channel used in this course in which each digit has probability p of being wrongly transmitted independently of what happens to the other digits. If $\epsilon > 0$, then*

$$\Pr\big(\textit{number of errors in transmission for message of } n \textit{ digits} \geq (1+\epsilon)pn\big) \to 0$$

as $n \to \infty$.

By Lemma 8.2, a code of minimum distance d can correct $\lfloor \frac{d-1}{2} \rfloor$ errors. Thus, if we have an error rate p and $\epsilon > 0$, we know that the probability that a code of length n with error correcting capacity $\lceil (1+\epsilon)pn \rceil$ will fail to correct a transmitted message falls to zero as $n \to \infty$. By definition, the biggest code with minimum distance $\lceil 2(1+\epsilon)pn \rceil$ has size $A(n, \lceil 2(1+\epsilon)pn \rceil)$ and so has information rate $\log_2 A(n, \lceil 2(1+\epsilon)pn \rceil)/n$. Study of the behaviour of $\log_2 A(n, n\delta)/n$ will thus tell us how large an information rate is possible in the presence of a given error rate.

Definition 9.4. *If $0 < \delta < 1/2$ we write*

$$\alpha(\delta) = \limsup_{n\to\infty} \frac{\log_2 A(n, n\delta)}{n}.$$

Definition 9.5. *We define the entropy function $H : [0,1] \to \mathbb{R}$ by $H(0) = H(1) = 0$ and*

$$H(t) = -t\log_2(t) - (1-t)\log_2(1-t).$$

Exercise 9.6. *(i) We have already met Shannon entropy in Definition 4.7. Give a simple system such that, using the notation of that definition,*

$$H(A) = H(t).$$

(ii) Sketch H. What is the value of $H(1/2)$?

Theorem 9.7. *With the definitions just given,*

$$1 - H(\delta) \leq \alpha(\delta) \leq 1 - H(\delta/2)$$

for all $0 \leq \delta < 1/2$.

Using the Hamming bound (Theorem 8.3) and the GSV bound (Theorem 8.8), we see that Theorem 9.7 follows at once from the following result.

Theorem 9.8. *We have*

$$\frac{\log_2 V(n, n\delta)}{n} \to H(\delta)$$

as $n \to \infty$.

Our proof of Theorem 9.8 depends, as one might expect, on a version of Stirling's formula. We only need the very simplest version proved in IA.

Lemma 9.9 (Stirling)**.** *We have*

$$\log_e n! = n\log_e n - n + O(\log_2 n).$$

We combine this with the remarks that

$$V(n, n\delta) = \sum_{0 \le j \le n\delta} \binom{n}{j}$$

and that very simple estimates give

$$\binom{n}{m} \le \sum_{0 \le j \le n\delta} \binom{n}{j} \le (m+1)\binom{n}{m}$$

where $m = \lfloor n\delta \rfloor$.

Although the GSV bound is very important, Shannon showed that a stronger result can be obtained for the error correcting power of the best long codes.

10 Shannon's noisy coding theorem

In the backstreets of Cambridge (Massachusetts) there is a science museum devoted to the glory of MIT. Since MIT has a great deal of glory and since much thought has gone into the presentation of the exhibits, it is well worth a visit. However, for any mathematician, the highlight is a glass case containing such things as a juggling machine, an electronic calculator[19] that uses Roman numerals both externally and internally, the remnants of a machine built to guess which of heads and tails its opponent would choose next[20] and a mechanical maze running mouse. These objects were built by Claude Shannon.

In his 1937 master's thesis, Shannon showed how to analyse circuits using Boolean algebra and binary arithmetic. During the war he worked on gunnery control and cryptography at Bell labs and in 1948 he published *A Mathematical Theory of Communication*[21]. Shannon had several predecessors and many successors, but it is his vision which underlies this course.

Hamming's bound together with Theorem 9.7 gives a very strong hint that it is not possible to have an information rate greater than $1 - H(\delta)$ for an error rate $\delta < 1/2$. (We shall prove this explicitly in Theorem 10.3.) On the other hand the GSV bound together with Theorem 9.7 shows that it is

[19]THROBAC the THrifty ROman numeral BAckwards-looking Computer. Google 'MIT Museum', go to 'objects' and then search 'Shannon'.

[20]That is to say a prediction machine. Google 'Shannon Mind-Reading Machine' for sites giving demonstrations and descriptions of the underlying program.

[21]This beautiful paper is available on the web and in his *Collected Works*.

always possible to have an information rate greater than $1 - H(2\delta)$ for an error rate $\delta < 1/4$.

Although we can use repetition codes to get a positive information rate when $1/4 \leq \delta < 1/2$ it looks very hard at first (and indeed second) glance to improve these results.

However, Shannon realised that we do not care whether errors arise because of noise in transmission or imperfections in our coding scheme. By allowing our coding scheme to be less than perfect (in this connection, see Question 25.13) we can actually improve the information rate whilst still keeping the error rate low.

Theorem 10.1. **[Shannon's noisy coding theorem]** *Suppose $0 < p < 1/2$ and $\eta > 0$. Then there exists an $n_0(p, \eta)$ such that, for any $n > n_0$, we can find codes of length n which have the property that (under our standard model of a symmetric binary channel with probability of error p) the probability that any codeword is mistaken is less than η and still have information rate $1 - H(p) - \eta$.*

Shannon's theorem is a masterly display of the power of elementary probabilistic arguments to overcome problems which appear insuperable by other means[22].

However, it merely asserts that good codes exist and gives no means of finding them apart from exhaustive search. More seriously, random codes will have no useful structure and the only way to use them is to 'search through a large dictionary' at the coding end and 'search through an enormous dictionary' at the decoding end. It should also be noted that $n_0(p, \eta)$ will be very large when p is close to $1/2$.

Exercise 10.2. *Why, in the absence of suitable structure, is the dictionary at the decoding end much larger than the dictionary at the coding end?*

It is relatively simple to obtain a converse to Shannon's theorem.

Theorem 10.3. *Suppose $0 < p < 1/2$ and $\eta > 0$. Then there exists an $n_0(p, \eta)$ such that, for any $n > n_0$, it is impossible to find codes of length n which have the property that (under our standard model of a symmetric binary channel with probability of error p) the probability that any codeword is mistaken is less than $1/2$ and the code has information rate $1 - H(p) + \eta$.*

[22] Conway says that in order to achieve success in a mathematical field you must either be first or be clever. However, as in the case of Shannon, most of those who are first to recognise a new mathematical field are also clever.

As might be expected, Shannon's theorem and its converse extend to more general noisy channels (in particular, those where the noise is governed by a Markov chain M). It is possible to define the entropy $H(M)$ associated with M and to show that the information rate cannot exceed $1 - H(M)$ but that any information rate lower than $1-H(M)$ can be attained with arbitrarily low error rates. However, we must leave something for more advanced courses, and as we said earlier, it is rare in practice to have very clear information about the nature of the noise we encounter.

There is one very important theorem of Shannon which is not covered in this course. In it, he reinterprets a result of Whittaker to show that any continuous signal whose Fourier transform vanishes outside a range of length R can be reconstructed from its value at equally spaced sampling points provided those points are less than A/R apart. (The constant A depends on the conventions used in defining the Fourier transform.) This enables us to apply the 'digital' theory of information transmission developed here to continuous signals.

11 A holiday at the race track

Although this section is examinable[23], the material is peripheral to the course. Suppose a very rich friend makes you the following offer. Every day, at noon, you may make a bet with her for any amount k you chose. You give her k pounds which she keeps whatever happens. She then tosses a coin and, if it shows heads, she pays you ku and, if it shows tails, she pays you nothing. You know that the probability of heads is p. What should you do?

If $pu < 1$, you should not bet, because your expected winnings are negative. If $pu > 1$, most mathematicians would be inclined to bet, but how much? If you bet your entire fortune and win, you will be better off than if you bet a smaller sum, but, if you lose, then you are bankrupt and cannot continue playing.

Thus your problem is to discover the proportion w of your present fortune that you should bet. Observe that your choice of w will always be the same (since you expect to go on playing for ever). Only the size of your fortune will vary. If your fortune after n goes is Z_n, then

$$Z_{n+1} = Z_n Y_{n+1}$$

[23] When the author of the present notes gives the course. This is his interpretation of the sentence in the schedules 'Applications to gambling and the stock market.' Other lecturers may view matters differently.

where $Y_{n+1} = uw + (1-w)$ if the $n+1$st throw is heads and $Y_{n+1} = 1-w$ if it is tails.

Using the weak law of large numbers, we have the following result.

Lemma 11.1. *Suppose Y, Y_1, Y_2, ... are identically distributed independent random variables taking values in $[a, b]$ with $0 < a < b$. If we write $Z_n = Y_1 \dots Y_n$, then*

$$\Pr(|n^{-1}\log Z_n - \mathbb{E}\log Y| > \epsilon) \to 0$$

as $n \to 0$.

Thus you should choose w to maximise

$$\mathbb{E}\log Y_n = p\log\big(uw + (1-w)\big) + (1-p)\log(1-w).$$

Exercise 11.2. *(i) Show that, for the situation described, you should not bet if $up \leq 1$ and should take*

$$w = \frac{up-1}{u-1}$$

if $up > 1$.

(ii) We write $q = 1-p$. Show that, if $up > 1$ and we choose the optimum w,

$$\mathbb{E}\log Y_n = p\log p + q\log q + \log u - q\log(u-1).$$

We have seen the expression $-(p\log p + q\log q)$ before as (a multiple of) the Shannon information entropy of a simple probabilistic system. In a paper entitled *A New Interpretation of Information Rate*[24] Kelly showed how to interpret this and similar situations using communication theory. In his model a gambler receives information over a noisy channel about which horse is going to win. Just as Shannon's theorem shows that information can be transmitted over such a channel at a rate close to channel capacity with negligible risk of error (provided the messages are long enough), so that the gambler can (with arbitrarily high probability) increase her fortune at a certain optimum rate provided that she can continue to bet long enough.

Although the analogy between betting and communication channels is very pretty, it was the suggestion that those making a long sequence of bets should aim to maximise the expectation of the logarithm (now called Kelly's criterion) which made the paper famous. Although Kelly seems never to have used his idea in practice, mathematicians like Thorp, Berlekamp and

[24]Available on the web. The exposition is slightly opaque because the Bell company which employed Kelly was anxious not draw attention to the use of telephones for betting fraud.

Shannon himself have made substantial fortunes in the stock market and claim to have used Kelly's ideas[25].

Kelly is also famous for an early demonstration of speech synthesis in which a computer sang 'Daisy Bell'. This inspired the corresponding scene in the film *2001*.

Before rushing out to the race track or stock exchange[26], the reader is invited to run computer simulations of the result of Kelly gambling for various values of u and p. She will observe that although, *in the very long run*, the system works, the short run can be be very unpleasant indeed.

Exercise 11.3. *Returning to our original problem, show that, if you bet less than the optimal proportion, your fortune will still tend to increase but more slowly, but, if you bet more than some proportion w_1, your fortune will decrease. Write down the equation for w_1.*

[*Moral: If you use the Kelly criterion veer on the side under-betting.*]

12 Linear codes

The next few sections involve no probability at all. We shall only be interested in constructing codes which are easy to handle and have all their code words at least a certain Hamming distance apart.

Just as $\mathbb{R}^n$ is a vector space over $\mathbb{R}$ and $\mathbb{C}^n$ is a vector space over $\mathbb{C}$, so $\mathbb{F}_2^n$ is a vector space over $\mathbb{F}_2$. (If you know about vector spaces over fields, so much the better, if not, just follow the obvious paths.) A *linear code* is a subspace of $\mathbb{F}_2^n$. More formally, we have the following definition.

Definition 12.1. *A linear code is a subset of $\mathbb{F}_2^n$ such that*
(i) $\mathbf{0} \in C$,
(ii) if $\mathbf{x}, \mathbf{y} \in C$, then $\mathbf{x} + \mathbf{y} \in C$.

Note that, if $\lambda \in \mathbb{F}$, then $\lambda = 0$ or $\lambda = 1$, so that condition (i) of the definition just given guarantees that $\lambda\mathbf{x} \in C$ whenever $\mathbf{x} \in C$. We shall see that linear codes have many useful properties.

Example 12.2. *(i) The repetition code with*

$$C = \{\mathbf{x} : \mathbf{x} = (x, x, \dots x)\}$$

is a linear code.

[25] However, we hear more about mathematicians who win on the stock market than those who lose.

[26] A sprat which thinks it's a shark will have a very short life.

(ii) The paper tape code

$$C = \left\{ \mathbf{x} : \sum_{j=0}^{n} x_j = 0 \right\}$$

is a linear code.

(iii) Hamming's original code is a linear code.

The verification is easy. In fact, examples (ii) and (iii) are 'parity check codes' and so automatically linear as we see from the next lemma.

Definition 12.3. *Consider a set P in $\mathbb{F}_2^n$. We say that C is the code defined by the set of* parity checks *P if the elements of C are precisely those $\mathbf{x} \in \mathbb{F}_2^n$ with*

$$\sum_{j=1}^{n} p_j x_j = 0$$

for all $\mathbf{p} \in P$.

Lemma 12.4. *If C is a code defined by parity checks, then C is linear.*

We now prove the converse result.

Definition 12.5. *If C is a linear code, we write $C^\perp$ for the set of $\mathbf{p} \in \mathbb{F}^n$ such that*

$$\sum_{j=1}^{n} p_j x_j = 0$$

for all $\mathbf{x} \in C$.

Thus $C^\perp$ is the set of parity checks satisfied by C.

Lemma 12.6. *If C is a linear code, then*

(i) $C^\perp$ is a linear code,

(ii) $(C^\perp)^\perp \supseteq C$.

We call $C^\perp$ the *dual code* to C.

In the language of the course on linear mathematics, $C^\perp$ is the annihilator of C. The following is a standard theorem of that course.

Lemma 12.7. *If C is a linear code in $\mathbb{F}_2^n$ then*

$$\dim C + \dim C^\perp = n.$$

Since the treatment of dual spaces is not the most popular piece of mathematics in IB, we shall give an independent proof later (see the note after Lemma 12.13). Combining Lemma 12.6 (ii) with Lemma 12.7, we get the following corollaries.

Lemma 12.8. *If C is a linear code, then $(C^\perp)^\perp = C$.*

Lemma 12.9. *Every linear code is defined by parity checks.*

Our treatment of linear codes has been rather abstract. In order to put computational flesh on the dry theoretical bones, we introduce the notion of a generator matrix.

Definition 12.10. *If C is a linear code of length n, any $r \times n$ matrix whose rows form a basis for C is called a* generator matrix *for C. We say that C has* dimension *or* rank *r.*

Example 12.11. *As examples, we can find generator matrices for the repetition code, the paper tape code and the original Hamming code.*

Remember that the Hamming code is the code of length 7 given by the parity conditions

$$\begin{aligned} x_1 + x_3 + x_5 + x_7 &= 0 \\ x_2 + x_3 + x_6 + x_7 &= 0 \\ x_4 + x_5 + x_6 + x_7 &= 0. \end{aligned}$$

By using row operations and column permutations to perform Gaussian elimination, we can give a constructive proof of the following lemma.

Lemma 12.12. *Any linear code of length n has (possibly after permuting the order of coordinates) a generator matrix of the form*

$$(I_r|B).$$

Notice that this means that any codeword $\mathbf{x}$ can be written as

$$(\mathbf{y}|\mathbf{z}) = (\mathbf{y}|\mathbf{y}B)$$

where $\mathbf{y} = (y_1, y_2, \dots, y_r)$ may be considered as the message and the vector $\mathbf{z} = \mathbf{y}B$ of length $n - r$ may be considered the check digits. Any code whose codewords can be split up in this manner is called *systematic*.

We now give a more computational treatment of parity checks.

Lemma 12.13. *If C is a linear code of length n with generator matrix G, then $\mathbf{a} \in C^\perp$ if and only if*

$$G\mathbf{a}^T = \mathbf{0}^T.$$

Thus

$$C^\perp = (\ker G)^T.$$

Using the rank, nullity theorem, we get a second proof of Lemma 12.7.

Lemma 12.13 enables us to characterise $C^\perp$.

Lemma 12.14. *If C is a linear code of length n and dimension r with generator the $n \times r$ matrix G, then, if H is any $n \times (n-r)$– matrix with columns forming a basis of $\ker G$, we know that H is a parity check matrix for C and its transpose H^T is a generator for $C^\perp$.*

Example 12.15. *(i) The dual of the paper tape code is the repetition code.*

(ii) Hamming's original code has dual with generator matrix

$$\begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

We saw above that the codewords of a linear code can be written

$$(\mathbf{y}|\mathbf{z}) = (\mathbf{y}|\mathbf{y}B)$$

where $\mathbf{y}$ may be considered as the vector of message digits and $\mathbf{z} = \mathbf{y}B$ as the vector of check digits. Thus *encoders* for linear codes are easy to construct.

What about decoders? Recall that every linear code of length n has a (non-unique) associated parity check matrix H with the property that $\mathbf{x} \in C$ if and only if $\mathbf{x}H = \mathbf{0}$. If $\mathbf{z} \in \mathbb{F}_2^n$, we define the *syndrome* of $\mathbf{z}$ to be $\mathbf{z}H$. The following lemma is mathematically trivial but forms the basis of the method of *syndrome decoding*.

Lemma 12.16. *Let C be a linear code with parity check matrix H. If we are given $\mathbf{z} = \mathbf{x} + \mathbf{e}$ where $\mathbf{x}$ is a code word and the 'error vector' $\mathbf{e} \in \mathbb{F}_2^n$, then*

$$\mathbf{z}H = \mathbf{e}H.$$

Suppose we have tabulated the syndrome $\mathbf{u}H$ for all $\mathbf{u}$ with 'few' non-zero entries (say, all $\mathbf{u}$ with $d(\mathbf{u}, \mathbf{0}) \leq K$). When our decoder receives $\mathbf{z}$, it computes the syndrome $\mathbf{z}H$. If the syndrome is zero, then $\mathbf{z} \in C$ and the decoder assumes the transmitted message was $\mathbf{z}$. If the syndrome of the received message is a non-zero vector $\mathbf{w}$, the decoder searches its list until it finds an $\mathbf{e}$ with $\mathbf{e}H = \mathbf{w}$. The decoder then assumes that the transmitted message was $\mathbf{x} = \mathbf{z} - \mathbf{e}$ (note that $\mathbf{z} - \mathbf{e}$ will always be a codeword, even if not the right one). This procedure will fail if $\mathbf{w}$ does not appear in the list, but, for this to be case, at least $K + 1$ errors must have occurred.

If we take $K = 1$, that is we only want a 1 error correcting code, then, writing $\mathbf{e}^{(i)}$ for the vector in $\mathbb{F}_2^n$ with 1 in the ith place and 0 elsewhere, we see that the syndrome $\mathbf{e}^{(i)}H$ is the ith row of H. If the transmitted message $\mathbf{z}$ has syndrome $\mathbf{z}H$ equal to the ith row of H, then the decoder assumes that there has been an error in the ith place and nowhere else. (Recall the special case of Hamming's original code.)

If K is large the task of searching the list of possible syndromes becomes onerous and, unless (as sometimes happens) we can find another trick, we find that 'decoding becomes dear' although 'encoding remains cheap'.

We conclude this section by looking at weights and the weight enumeration polynomial for a linear code. The idea here is to exploit the fact that, if C is linear code and $\mathbf{a} \in C$, then $\mathbf{a} + C = C$. Thus the 'view of C' from any codeword $\mathbf{a}$ is the same as the 'view of C' from the particular codeword $\mathbf{0}$.

Definition 12.17. *The* weight $w(\mathbf{x})$ *of a vector* $\mathbf{x} \in \mathbb{F}_2^n$ *is given by*

$$w(\mathbf{x}) = d(\mathbf{0}, \mathbf{x}).$$

Lemma 12.18. *If w is the weight function on $\mathbb{F}_2^n$ and $\mathbf{x}$, $\mathbf{y} \in \mathbb{F}_2^n$, then*
(i) $w(\mathbf{x}) \geq 0$,
(ii) $w(\mathbf{x}) = 0$ *if and only if* $\mathbf{x} = \mathbf{0}$,
(iii) $w(\mathbf{x}) + w(\mathbf{y}) \geq w(\mathbf{x} + \mathbf{y})$.

Since the minimum (non-zero) weight in a linear code is the same as the minimum (non-zero) distance, we can talk about linear codes of minimum weight d when we mean linear codes of minimum distance d.

The pattern of distances in a linear code is encapsulated in the weight enumeration polynomial.

Definition 12.19. *Let C be a linear code of length n. We write A_j for the number of codewords of weight j and define the* weight enumeration polynomial W_C *to be the polynomial in two real variables given by*

$$W_C(s, t) = \sum_{j=0}^{n} A_j s^j t^{n-j}.$$

Here are some simple properties of W_C.

Lemma 12.20. *Under the assumptions and with the notation of Definition 12.19, the following results are true.*

(i) W_C *is a homogeneous polynomial of degree* n.

(ii) If C *has rank* r, *then* $W_C(1,1) = 2^r$.

(iii) $W_C(0,1) = 1$.

(iv) $W_C(1,0)$ *takes the value* 0 *or* 1.

(v) $W_C(s,t) = W_C(t,s)$ *for all* s *and* t *if and only if* $W_C(1,0) = 1$.

Lemma 12.21. *For our standard model of communication along an error prone channel with independent errors of probability* p *and a linear code* C *of length* n,

$$W_C(p, 1-p) = \Pr(\textit{receive a code word} \mid \textit{code word transmitted})$$

and

$$\Pr(\textit{receive incorrect code word} \mid \textit{code word transmitted}) = W_C(p, 1-p) - (1-p)^n.$$

Example 12.22. *(i) If* C *is the repetition code,* $W_C(s,t) = s^n + t^n$.

(ii) If C *is the paper tape code of length* n, $W_C(s,t) = \frac{1}{2}((s+t)^n + (t-s)^n)$.

Example 12.22 is a special case of the MacWilliams identity.

Theorem 12.23. [**MacWilliams identity**] *If* C *is a linear code*

$$W_{C^\perp}(s,t) = 2^{-\dim C} W_C(t-s, t+s).$$

We give a proof as Exercise 26.9. (The result is thus not bookwork though it could be set as a problem with appropriate hints.)

13 Some general constructions

However interesting the theoretical study of codes may be to a pure mathematician, the engineer would prefer to have an arsenal of practical codes so that she can select the one most suitable for the job in hand. In this section we discuss the general Hamming codes and the Reed-Muller codes as well as some simple methods of obtaining new codes from old.

Definition 13.1. *Let* d *be a strictly positive integer and let* $n = 2^d - 1$. *Consider the (column) vector space* $D = \mathbb{F}_2^d$. *Write down a* $d \times n$ *matrix* H *whose columns are the* $2^d - 1$ *distinct non-zero vectors of* D. *The Hamming* $(n, n-d)$ *code is the linear code of length* n *with* H^T *as parity check matrix.*

Of course the Hamming $(n, n-d)$ code is only defined up to permutation of coordinates. We note that H has rank d, so a simple use of the rank nullity theorem shows that our notation is consistent.

Lemma 13.2. *The Hamming $(n, n-d)$ code is a linear code of length n and rank $n-d$ $[n = 2^d - 1]$.*

Example 13.3. *The Hamming $(7, 4)$ code is the original Hamming code.*

The fact that any two rows of H are linearly independent and a look at the appropriate syndromes gives us the main property of the general Hamming code.

Lemma 13.4. *The Hamming $(n, n-d)$ code has minimum weight 3 and is a perfect 1 error correcting code $[n = 2^d - 1]$.*

Hamming codes are ideal in situations where very long strings of binary digits must be transmitted but the chance of an error in any individual digit is very small. (Look at Exercise 7.5.) Although the search for perfect codes other than the Hamming codes produced the Golay code (not discussed here) and much interesting combinatorics, the reader is warned that, from a practical point of view, it represents a dead end[27].

Here are a number of simple tricks for creating new codes from old.

Definition 13.5. *If C is a code of length n, the* parity check extension C^+ *of C is the code of length $n+1$ given by*

$$C^+ = \left\{ \mathbf{x} \in \mathbb{F}_2^{n+1} : (x_1, x_2, \dots, x_n) \in C,\ \sum_{j=1}^{n+1} x_j = 0 \right\}.$$

Definition 13.6. *If C is a code of length n, the* truncation C^- *of C is the code of length $n-1$ given by*

$$C^- = \{(x_1, x_2, \dots, x_{n-1}) : (x_1, x_2, \dots, x_n) \in C \text{ for some } x_n \in \mathbb{F}_2\}.$$

[27]If we confine ourselves to the binary codes discussed in this course, it is known that perfect codes of length n with Hamming spheres of radius ρ exist for $\rho = 0$, $\rho = n$, $\rho = (n-1)/2$, with n odd (the three codes just mentioned are easy to identify), $\rho = 3$ and $n = 23$ (the Golay code, found by direct search) and $\rho = 1$ and $n = 2^m - 1$. There are known to be non-Hamming codes with $\rho = 1$ and $n = 2^m - 1$, it is suspected that there are many of them and they are the subject of much research, but, of course they present no practical advantages. The only linear perfect codes with $\rho = 1$ and $n = 2^m - 1$ are the Hamming codes.

Definition 13.7. *If C is a code of length n, the* shortening *(or* puncturing*) C' of C by the symbol α (which may be 0 or 1) is the code of length $n-1$ given by*

$$C' = \{(x_1, x_2, \dots, x_{n-1}) : (x_1, x_2, \dots, x_{n-1}, \alpha) \in C\}.$$

Lemma 13.8. *If C is linear, so is its parity check extension C^+, its truncation C^- and its shortening C' (provided that the symbol chosen is 0).*

How can we combine two linear codes C_1 and C_2? Our first thought might be to look at their direct sum

$$C_1 \oplus C_2 = \{(\mathbf{x}|\mathbf{y}) : \mathbf{x} \in C_1,\ \mathbf{y} \in C_2\},$$

but this is unlikely to be satisfactory.

Lemma 13.9. *If C_1 and C_2 are linear codes, then we have the following relation between minimum distances.*

$$d(C_1 \oplus C_2) = \min\big(d(C_1), d(C_2)\big).$$

On the other hand, if C_1 and C_2 satisfy rather particular conditions, we can obtain a more promising construction.

Definition 13.10. *Suppose C_1 and C_2 are linear codes of length n with $C_1 \supseteq C_2$ (i.e. with C_2 a subspace of C_1). We define the* bar product *$C_1|C_2$ of C_1 and C_2 to be the code of length $2n$ given by*

$$C_1|C_2 = \{(\mathbf{x}|\mathbf{x}+\mathbf{y}) : \mathbf{x} \in C_1,\ \mathbf{y} \in C_2\}.$$

Lemma 13.11. *Let C_1 and C_2 be linear codes of length n with $C_1 \supseteq C_2$. Then the bar product $C_1|C_2$ is a linear code with*

$$\operatorname{rank} C_1|C_2 = \operatorname{rank} C_1 + \operatorname{rank} C_2.$$

The minimum distance of $C_1|C_2$ satisfies the equality

$$d(C_1|C_2) = \min(2d(C_1), d(C_2)).$$

We now return to the construction of specific codes. Recall that the Hamming codes are suitable for situations when the error rate p is very small and we want a high information rate. The Reed-Muller are suitable when the error rate is very high and we are prepared to sacrifice information rate. They were used by NASA for the radio transmissions from its planetary probes (a task which has been compared to signalling across the Atlantic with a child's torch[28]).

We start by considering the 2^d points P_0, P_1, ..., P_{2^d-1} of the space $X = \mathbb{F}_2^d$. Our code words will be of length $n = 2^d$ and will correspond to the indicator functions $\mathbb{I}_A$ on X. More specifically, the possible code word $\mathbf{c}^A$ is given by

$$\begin{aligned} c_i^A &= 1 && \text{if } P_i \in A \\ c_i^A &= 0 && \text{otherwise.} \end{aligned}$$

for some $A \subseteq X$.

In addition to the usual vector space structure on $\mathbb{F}_2^n$, we define a new operation

$$\mathbf{c}^A \wedge \mathbf{c}^B = \mathbf{c}^{A \cap B}.$$

Thus, if $\mathbf{x}, \mathbf{y} \in \mathbb{F}_2^n$,

$$(x_0, x_1, \dots, x_{n-1}) \wedge (y_0, y_1, \dots, y_{n-1}) = (x_0 y_0, x_1 y_1, \dots, x_{n-1} y_{n-1}).$$

Finally we consider the collection of d hyperplanes

$$\pi_j = \{\mathbf{p} \in X : p_j = 0\} \quad [1 \leq j \leq d]$$

in $\mathbb{F}_2^n$ and the corresponding indicator functions

$$\mathbf{h}^j = \mathbf{c}^{\pi_j},$$

together with the special vector

$$\mathbf{h}^0 = \mathbf{c}^X = (1, 1, \dots, 1).$$

Exercise 13.12. *Suppose that* $\mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathbb{F}_2^n$ *and* $A, B \subseteq X$.

(i) Show that $\mathbf{x} \wedge \mathbf{y} = \mathbf{y} \wedge \mathbf{x}$.

(ii) Show that $(\mathbf{x} + \mathbf{y}) \wedge \mathbf{z} = \mathbf{x} \wedge \mathbf{z} + \mathbf{y} \wedge \mathbf{z}$.

(iii) Show that $\mathbf{h}^0 \wedge \mathbf{x} = \mathbf{x}$.

(iv) If $\mathbf{c}^A + \mathbf{c}^B = \mathbf{c}^E$, *find* E *in terms of* A *and* B.

(v) If $\mathbf{h}^0 + \mathbf{c}^A = \mathbf{c}^E$, *find* E *in terms of* A.

We refer to $\mathcal{A}_0 = \{\mathbf{h}^0\}$ as the set of terms of order zero. If $\mathcal{A}_k$ is the set of terms of order at most k, then the set $\mathcal{A}_{k+1}$ of terms of order at most $k+1$ is defined by

$$\mathcal{A}_{k+1} = \{\mathbf{a} \wedge \mathbf{h}^j : \mathbf{a} \in \mathcal{A}_k,\ 1 \leq j \leq d\}.$$

Less formally, but more clearly, the elements of order 1 are the $\mathbf{h}^i$, the elements of order 2 are the $\mathbf{h}^i \wedge \mathbf{h}^j$ with $i < j$, the elements of order 3 are the $\mathbf{h}^i \wedge \mathbf{h}^j \wedge \mathbf{h}^k$ with $i < j < k$ and so on.

[28]Strictly speaking, the comparison is meaningless. However, it sounds impressive and that is the main thing.

Definition 13.13. *Using the notation established above, the Reed-Muller code $RM(d,r)$ is the linear code (i.e. subspace of $\mathbb{F}_2^n$) generated by the terms of order r or less.*

Although the formal definition of the Reed-Muller codes looks pretty impenetrable at first sight, once we have looked at sufficiently many examples it should become clear what is going on.

Example 13.14. *(i) The $RM(3,0)$ code is the repetition code of length 8.*

(ii) The $RM(3,1)$ code is the parity check extension of Hamming's original code.

(iii) The $RM(3,2)$ code is the paper tape code of length 8.

(iii) The $RM(3,3)$ code is the trivial code consisting of all the elements of $\mathbb{F}_2^3$.

We now prove the key properties of the Reed-Muller codes. We use the notation established above.

Theorem 13.15. *(i) The elements of order d or less (that is the collection of all possible wedge products formed from the $\mathbf{h}^i$) span $\mathbb{F}_2^n$.*

(ii) The elements of order d or less are linearly independent.

(iii) The dimension of the Reed-Muller code $RM(d,r)$ is

$$\binom{d}{0}+\binom{d}{1}+\binom{d}{2}+\dots+\binom{d}{r}.$$

(iv) Using the bar product notation, we have

$$RM(d,r)=RM(d-1,r)|RM(d-1,r-1).$$

(v) The minimum weight of $RM(d,r)$ is exactly 2^{d-r}.

Exercise 13.16. *The Mariner mission to Mars used the $RM(5,1)$ code. What was its information rate? What proportion of errors could it correct in a single code word?*

Exercise 13.17. *Show that the $RM(d,d-2)$ code is the parity extension code of the Hamming $(N,N-d)$ code with $N=2^d-1$. (This is useful because we often want codes of length 2^d.)*

14 Polynomials and fields

This section is starred. Its object is to make plausible the few facts from modern[29] algebra that we shall need. They were covered, along with much else, in various post-IA algebra courses, but attendance at those courses is no more required for this course than is reading Joyce's *Ulysses* before going for a night out at an Irish pub. Anyone capable of criticising the imprecision and general slackness of the account that follows obviously can do better themselves and should rewrite this section in an appropriate manner.

A field K is an object equipped with addition and multiplication which follow the same rules as do addition and multiplication in $\mathbb{R}$. The only rule which will cause us trouble is

$$\text{If } x \in K \text{ and } x \neq 0, \text{ then we can find } y \in K \text{ such that } xy = 1. \qquad \bigstar$$

Obvious examples of fields include $\mathbb{R}$, $\mathbb{C}$ and $\mathbb{F}_2$.

We are particularly interested in polynomials over fields, but here an interesting difficulty arises.

Example 14.1. *We have $t^2 + t = 0$ for all $t \in \mathbb{F}_2$.*

To get round this, we distinguish between the polynomial in the 'indeterminate' X

$$P(X) = \sum_{j=0}^{n} a_j X^j$$

with coefficients $a_j \in K$ and its evaluation $P(t) = \sum_{j=0}^{n} a_j t^j$ for some $t \in K$. We manipulate polynomials in X according to the standard rules for polynomials, but say that

$$\sum_{j=0}^{n} a_j X^j = 0$$

if and only if $a_j = 0$ for all j. Thus $X^2 + X$ is a non-zero polynomial over $\mathbb{F}_2$ all of whose values are zero.

The following result is familiar, in essence, from school mathematics.

Lemma 14.2. [Remainder theorem] *(i) If P is a polynomial over a field K and $a \in K$, then we can find a polynomial Q and an $r \in K$ such that*

$$P(X) = (X - a)Q(X) + r.$$

(ii) If P is a polynomial over a field K and $a \in K$ is such that $P(a) = 0$, then we can find a polynomial Q such that

$$P(X) = (X - a)Q(X).$$

[29] Modern, that is, in 1920.

The key to much of the elementary theory of polynomials lies in the fact that we can apply Euclid's algorithm to obtain results like the following.

Theorem 14.3. *Suppose that $\mathcal{P}$ is a set of polynomials which contains at least one non-zero polynomial and has the following properties.*

(i) If Q is any polynomial and $P \in \mathcal{P}$, then the product $PQ \in \mathcal{P}$.

(ii) If $P_1, P_2 \in \mathcal{P}$, then $P_1 + P_2 \in \mathcal{P}$.

Then we can find a non-zero $P_0 \in \mathcal{P}$ which divides every $P \in \mathcal{P}$.

Proof. Consider a non-zero polynomial P_0 of smallest degree in $\mathcal{P}$. □

Recall that the polynomial $P(X) = X^2 + 1$ has no roots in $\mathbb{R}$ (that is $P(t) \neq 0$ for all $t \in \mathbb{R}$). However, by considering the collection of formal expressions $a + bi$ $[a, b \in \mathbb{R}]$ with the obvious formal definitions of addition and multiplication and subject to the further condition $i^2 + 1 = 0$, we obtain a field $\mathbb{C} \supseteq \mathbb{R}$ in which P has a root (since $P(i) = 0$). We can perform a similar trick with other fields.

Example 14.4. *If $P(X) = X^2 + X + 1$, then P has no roots in $\mathbb{F}_2$. However, if we consider*

$$\mathbb{F}_2[\omega] = \{0,\ 1,\ \omega,\ 1 + \omega\}$$

with obvious formal definitions of addition and multiplication and subject to the further condition $\omega^2 + \omega + 1 = 0$, then $\mathbb{F}_2[\omega]$ is a field containing $\mathbb{F}_2$ in which P has a root (since $P(\omega) = 0$).

Proof. The only thing we really need prove is that $\mathbb{F}_2[\omega]$ is a field and to do that the only thing we need to prove is that ★ holds. Since

$$(1 + \omega)\omega = 1$$

this is easy. □

In order to state a correct generalisation of the ideas of the previous paragraph we need a preliminary definition.

Definition 14.5. *If P is a polynomial over a field K, we say that P is* reducible *if there exists a non-constant polynomial Q of degree strictly less than P which divides P. If P is a non-constant polynomial which is not reducible, then P is* irreducible.

Theorem 14.6. *If P is an irreducible polynomial of degree $n \geq 2$ over a field K, then P has no roots in K. However, if we consider*

$$K[\omega] = \left\{ \sum_{j=0}^{n-1} a_j \omega^j : a_j \in K \right\}$$

with the obvious formal definitions of addition and multiplication and subject to the further condition $P(\omega) = 0$, then $K[\omega]$ is a field containing K in which P has a root.

Proof. The only thing we really need prove is that $K[\omega]$ is a field and to do that the only thing we need to prove is that ★ holds. Let Q be a non-zero polynomial of degree at most $n-1$. Since P is irreducible, the polynomials P and Q have no common factor of degree 1 or more. Hence, by Euclid's algorithm, we can find polynomials R and S such that

$$R(X)Q(X) + S(X)P(X) = 1$$

and so $R(\omega)Q(\omega) + S(\omega)P(\omega) = 1$. But $P(\omega) = 0$, so $R(\omega)Q(\omega) = 1$ and we have proved ★. □

In a proper algebra course we would simply define

$$K[\omega] = K[X]/(P(X))$$

where $(P(X))$ is the ideal generated by $P(X)$. This is a cleaner procedure which avoids the use of such phrases as 'the obvious formal definitions of addition and multiplication' but the underlying idea remains the same.

Lemma 14.7. *If P is a polynomial over a field K which does not factorise completely into linear factors, then we can find a field $L \supseteq K$ in which P has more linear factors.*

Proof. Factor P into irreducible factors and choose a factor Q which is not linear. By Theorem 14.6, we can find a field $L \supseteq K$ in which Q has a root α say and so, by Lemma 14.2, a linear factor $X - \alpha$. Since any linear factor of P in K remains a factor in the bigger field L, we are done. □

Theorem 14.8. *If P is a polynomial over a field K, then we can find a field $L \supseteq K$ in which P factorises completely into linear factors.*

We shall be interested in finite fields (that is fields K with only a finite number of elements). A glance at our method of proving Theorem 14.8 shows that the following result holds.

Lemma 14.9. *If P is a polynomial over a finite field K, then we can find a finite field $L \supseteq K$ in which P factorises completely.*

In this context, we note yet another useful simple consequence of Euclid's algorithm.

Lemma 14.10. *Suppose that P is an irreducible polynomial over a field K which has a linear factor $X-\alpha$ in some field $L \supseteq K$. If Q is a polynomial over K which has the factor $X-\alpha$ in L, then P divides Q.*

We shall need a lemma on repeated roots.

Lemma 14.11. *Let K be a field. If $P(X)=\sum_{j=0}^{n} a_j X^j$ is a polynomial over K, we define $P'(X)=\sum_{j=1}^{n} j a_j X^{j-1}$.*

(i) If P and Q are polynomials, $(P+Q)'=P'+Q'$ and $(PQ)'=P'Q+PQ'$.

(ii) If P and Q are polynomials with $P(X)=(X-a)^2Q(X)$, then

$$P'(X)=2(X-a)Q(X)+(X-a)^2Q'(X).$$

(iii) If P is divisible by $(X-a)^2$, then $P(a)=P'(a)=0$.

If L is a field containing $\mathbb{F}_2$, then $2y=(1+1)y=0y=0$ for all $y \in L$. We can thus deduce the following result which will be used in the next section.

Lemma 14.12. *If L is a field containing $\mathbb{F}_2$ and n is an odd integer, then X^n-1 can have no repeated linear factors as a polynomial over L.*

We also need a result on roots of unity given as part (v) of the next lemma.

Lemma 14.13. *(i) If G is a finite Abelian group and $x, y \in G$ have coprime orders r and s, then xy has order rs.*

(ii) If G is a finite Abelian group and $x, y \in G$ have orders r and s, then we can find an element z of G with order the lowest common multiple of r and s.

(iii) If G is a finite Abelian group, then there exists an N and an $h \in G$ such that h has order N and $g^N=e$ for all $g \in G$.

(iv) If G is a finite subset of a field K which is a group under multiplication, then G is cyclic.

(v) Suppose n is an odd integer. If L is a field containing $\mathbb{F}_2$ such that X^n-1 factorises completely into linear terms, then we can find an $\omega \in L$ such that the roots of X^n-1 are 1, ω, ω^2, $\dots \omega^{n-1}$. (We call ω a primitive *nth root of unity.)*

Proof. (ii) Consider $z=x^u y^v$ where u is a divisor of r, v is a divisor of s, r/u and s/v are coprime and $rs/(uv)=\operatorname{lcm}(r,s)$.

(iii) Let h be an element of highest order in G and use (ii).

(iv) By (iii) we can find an integer N and a $h \in G$ such that h has order N and any element $g \in G$ satisfies $g^N=1$. Thus X^N-1 has a linear factor

$X - g$ for each $g \in G$ and so $\prod_{g\in G}(X-g)$ divides $X^N - 1$. It follows that the order $|G|$ of G cannot exceed N. But by Lagrange's theorem N divides G. Thus $|G| = N$ and g generates G.

(v) Observe that $G = \{\omega : \omega^n = 1\}$ is an Abelian group with exactly n elements (since $X^n - 1$ has no repeated roots) and use (iv). $\square$

Here is another interesting consequence of Lemma 14.13 (iv).

Lemma 14.14. *If K is a field with m elements, then there is an element k of K such that*

$$K = \{0\} \cup \{k^r : 0 \le r \le m-2\}$$

and $k^{m-1} = 1$.

Proof. Observe that $K \setminus \{0\}$ forms an Abelian group under multiplication. $\square$

We call an element k with the properties given in Lemma 14.14 a *primitive element* of K.

Exercise 14.15. *Find all the primitive elements of $\mathbb{F}_7$.*

With this hint, it is not hard to show that there is indeed a field with 2^n elements containing $\mathbb{F}_2$.

Lemma 14.16. *Let L be some field containing $\mathbb{F}_2$ in which $X^{2^n-1} - 1 = 0$ factorises completely. Then*

$$K = \{x \in L : x^{2^n} = x\}$$

is a field with 2^n elements containing $\mathbb{F}_2$.

Lemma 14.14 shows that there is (up to field isomorphism) only one field with 2^n elements containing $\mathbb{F}_2$. We call it $\mathbb{F}_{2^n}$.

15 Cyclic codes

In this section, we discuss a subclass of linear codes, the so-called *cyclic codes*.

Definition 15.1. *A linear code C in $\mathbb{F}_2^n$ is called* cyclic *if*

$$(a_0, a_1, \dots, a_{n-2}, a_{n-1}) \in C \Rightarrow (a_1, a_2, \dots, a_{n-1}, a_0) \in C.$$

Let us establish a correspondence between $\mathbb{F}_2^n$ and the polynomials on $\mathbb{F}_2$ modulo $X^n - 1$ by setting

$$P_{\mathbf{a}} = \sum_{j=0}^{n-1} a_j X^j$$

whenever $\mathbf{a} \in \mathbb{F}_2^n$. (Of course, $X^n - 1 = X^n + 1$ but in this context the first expression seems more natural.)

Exercise 15.2. *With the notation just established, show that*
(i) $P_{\mathbf{a}} + P_{\mathbf{b}} = P_{\mathbf{a}+\mathbf{b}}$,
(ii) $P_{\mathbf{a}} = 0$ *if and only if* $\mathbf{a} = \mathbf{0}$.

Lemma 15.3. *A code C in $\mathbb{F}_2^n$ is cyclic if and only if $\mathcal{P}_C = \{P_{\mathbf{a}} : \mathbf{a} \in C\}$ satisfies the following two conditions (working modulo $X^n - 1$).*
(i) If $f, g \in \mathcal{P}_C$, then $f + g \in \mathcal{P}_C$.
(ii) If $f \in \mathcal{P}_C$ and g is any polynomial, then the product $fg \in \mathcal{P}_C$.

(In the language of abstract algebra, C is cyclic if and only if $\mathcal{P}_C$ is an *ideal* of the quotient ring $\mathbb{F}_2[X]/(X^n - 1)$.)

From now on we shall talk of the code word $f(X)$ when we mean the code word $\mathbf{a}$ with $P_{\mathbf{a}}(X) = f(X)$. An application of Euclid's algorithm gives the following useful result.

Lemma 15.4. *A code C of length n is cyclic if and only if (working modulo $X^n - 1$, and using the conventions established above) there exists a polynomial g such that*

$$C = \{f(X)g(X) : f \text{ a polynomial}\}$$

(In the language of abstract algebra, $\mathbb{F}_2[X]$ is a Euclidean domain and so a principal ideal domain. Thus the quotient $\mathbb{F}_2[X]/(X^n - 1)$ is a principal ideal domain.) We call $g(X)$ a generator polynomial for C.

Lemma 15.5. *A polynomial g is a generator for a cyclic code of length n if and only if it divides $X^n - 1$.*

Thus we must seek generators among the factors of $X^n - 1 = X^n + 1$. If there are no conditions on n, the result can be rather disappointing.

Exercise 15.6. *If we work with polynomials over $\mathbb{F}_2$, then*

$$X^{2^r} + 1 = (X + 1)^{2^r}.$$

In order to avoid this problem and to be able to make use of Lemma 14.12, we shall take n odd from now on. (In this case, the cyclic codes are said to be separable.) Notice that the task of finding irreducible factors (that is factors with no further factorisation) is a finite one.

Lemma 15.7. *Consider codes of length n. Suppose that $g(X)h(X) = X^n - 1$. Then g is a generator of a cyclic code C and h is a generator for a cyclic code which is the reverse of $C^\perp$.*

As an immediate corollary, we have the following remark.

Lemma 15.8. *The dual of a cyclic code is itself cyclic.*

Lemma 15.9. *If a cyclic code C of length n has generator g of degree $n-r$ then $g(X)$, $Xg(X)$, ..., $X^{r-1}g(X)$ form a basis for C.*

Cyclic codes are thus easy to specify (we just need to write down the generator polynomial g) and to encode.

We know that $X^n + 1$ factorises completely over some larger finite field and, since n is odd, we know, by Lemma 14.12, that it has no repeated factors. The same is therefore true for any polynomial dividing it.

Lemma 15.10. *Suppose that g is a generator of a cyclic code C of odd length n. Suppose further that g factorises completely into linear factors in some field K containing $\mathbb{F}_2$. If $g = g_1g_2\dots g_k$ with each g_j irreducible over $\mathbb{F}_2$ and A is a subset of the set of all the roots of all the g_j and containing at least one root of each g_j $[1 \le j \le k]$, then*

$$C = \{f \in \mathbb{F}_2[X] : f(\alpha) = 0 \text{ for all } \alpha \in A\}.$$

Definition 15.11. *A* defining set *for a cyclic code C is a set A of elements in some field K containing $\mathbb{F}_2$ such that $f \in \mathbb{F}_2[X]$ belongs to C if and only if $f(\alpha) = 0$ for all $\alpha \in A$.*

(Note that, if C has length n, A must be a set of zeros of $X^n - 1$.)

Lemma 15.12. *Suppose that*

$$A = \{\alpha_1, \alpha_2, \dots, \alpha_r\}$$

is a defining set for a cyclic code C in some field K containing $\mathbb{F}_2$. Let B be the $n \times r$ matrix over K whose jth column is

$$(1, \alpha_j, \alpha_j^2, \dots, \alpha_j^{n-1})^T$$

Then a vector $\mathbf{a} \in \mathbb{F}_2^n$ is a code word in C if and only if

$$\mathbf{a}B = \mathbf{0}$$

in K.

The columns in B are not parity checks in the usual sense since the code entries lie in $\mathbb{F}_2$ and the computations take place in the larger field K.

With this background we can discuss a famous family of codes known as the BCH (Bose, Ray-Chaudhuri, Hocquenghem) codes. Recall that a primitive nth root of unity is an root α of $X^n - 1 = 0$ such that every root is a power of α.

Definition 15.13. *Suppose that n is odd and K is a field containing $\mathbb{F}_2$ in which $X^n - 1$ factorises into linear factors. Suppose that $\alpha \in K$ is a primitive nth root of unity. A cyclic code C with defining set*

$$A = \{\alpha, \alpha^2, \dots, \alpha^{\delta-1}\}$$

is a BCH code of design distance δ.

Note that the rank of C will be $n - k$, where k is the degree of the product of those irreducible factors of $X^n - 1$ over $\mathbb{F}_2$ which have a zero in A. Notice also that k may be very much larger than δ.

Example 15.14. *(i) If K is a field containing $\mathbb{F}_2$, then $(a + b)^2 = a^2 + b^2$ for all $a, b \in K$.*

(ii) If $P \in \mathbb{F}_2[X]$ and K is a field containing $\mathbb{F}_2$, then $P(a)^2 = P(a^2)$ for all $a \in K$.

(iii) Let K be a field containing $\mathbb{F}_2$ in which $X^7 - 1$ factorises into linear factors. If β is a root of $X^3 + X + 1$ in K, then β is a primitive root of unity and β^2 is also a root of $X^3 + X + 1$.

(iv) We continue with the notation (iii). The BCH code with $\{\beta, \beta^2\}$ as defining set is Hamming's original (7,4) code.

The next theorem contains the key fact about BCH codes.

Theorem 15.15. *The minimum distance for a BCH code is at least as great as the design distance.*

Our proof of Theorem 15.15 relies on showing that the matrix B of Lemma 15.12 is of full rank for a BCH. To do this we use a result which every undergraduate knew in 1950.

Lemma 15.16. [The van der Monde determinant] *We work over a field K. The determinant*

$$\begin{vmatrix} 1 & 1 & 1 & \dots & 1 \\ x_1 & x_2 & x_3 & \dots & x_n \\ x_1^2 & x_2^2 & x_3^2 & \dots & x_n^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_1^{n-1} & x_2^{n-1} & x_3^{n-1} & \dots & x_n^{n-1} \end{vmatrix} = \prod_{1 \le j < i \le n} (x_i - x_j).$$

How can we construct a decoder for a BCH code? From now on, until the end of this section, we shall suppose that we are using the BCH code C described in Definition 15.13. In particular, C will have length n and defining set

$$A = \{\alpha, \alpha^2, \dots, \alpha^{\delta-1}\}$$

where α is a primitive nth root of unity in K. Let t be the largest integer with $2t + 1 \leq \delta$. We show how we can correct up to t errors.

Suppose that a codeword $\mathbf{c} = (c_0, c_1, \dots, c_{n-1})$ is transmitted and that the string received is $\mathbf{r}$. We write $\mathbf{e} = \mathbf{r} - \mathbf{c}$ and assume that

$$\mathcal{E} = \{0 \leq j \leq n-1 : e_j \neq 0\}$$

has no more than t members. In other words, $\mathbf{e}$ is the error vector and we assume that there are no more than t errors. We write

$$\begin{aligned} c(X) &= \sum_{j=0}^{n-1} c_j X^j, \\ r(X) &= \sum_{j=0}^{n-1} r_j X^j, \\ e(X) &= \sum_{j=0}^{n-1} e_j X^j. \end{aligned}$$

Definition 15.17. *The* error locator polynomial *is*

$$\sigma(X) = \prod_{j \in \mathcal{E}} (1 - \alpha^j X)$$

and the error co-locator *is*

$$\omega(X) = \sum_{i=0}^{n-1} e_i \alpha^i \prod_{j \in \mathcal{E},\ j \neq i} (1 - \alpha^j X).$$

Informally, we write

$$\omega(X) = \sum_{i=0}^{n-1} e_i \alpha^i \frac{\sigma(X)}{1 - \alpha^i X}.$$

We take $\omega(X) = \sum_j \omega_j X^j$ and $\sigma(X) = \sum_j \sigma_j X^j$. Note that ω has degree at most $t-1$ and σ degree at most t. Note that we know that $\sigma_0 = 1$ so both the polynomials ω and σ have t unknown coefficients.

Lemma 15.18. *If the error locator polynomial is given the value of* $\mathbf{e}$ *and so of* $\mathbf{c}$ *can be obtained directly.*

We wish to make use of relations of the form

$$\frac{1}{1-\alpha^j X} = \sum_{r=0}^{\infty} (\alpha^j X)^r.$$

Unfortunately, it is not clear what meaning to assign to such a relation. One way round is to work modulo Z^{2t} (more formally, to work in $K[Z]/(Z^{2t})$). We then have $Z^u \equiv 0$ for all integers $u \geq 2t$.

Lemma 15.19. *If we work modulo* Z^{2t} *then*

$$(1-\alpha^j Z) \sum_{m=0}^{2t-1} (\alpha^j Z)^m \equiv 1.$$

Thus, if we work modulo Z^{2t}, as we shall from now on, we may define

$$\frac{1}{1-\alpha^j Z} = \sum_{m=0}^{2t-1} (\alpha^j Z)^m.$$

Lemma 15.20. *With the conventions already introduced.*

(i) $\dfrac{\omega(Z)}{\sigma(Z)} \equiv \displaystyle\sum_{m=0}^{2t-1} Z^m e(\alpha^{m+1})$.

(ii) $e(\alpha^m) = r(\alpha^m)$ *for all* $0 \leq m \leq 2t-1$.

(iii) $\dfrac{\omega(Z)}{\sigma(Z)} \equiv \displaystyle\sum_{m=0}^{2t-1} Z^m r(\alpha^{m+1})$.

(iv) $\omega(Z) \equiv \sum_{m=0}^{2t-1} Z^m r(\alpha^{m+1}) \sigma(Z)$.

(v) $\omega_j = \displaystyle\sum_{u+v=j} r(\alpha^{u+1})\sigma_v$ *for all* $0 \leq j \leq t-1$.

(vi) $0 = \displaystyle\sum_{u+v=j} r(\alpha^{u+1})\sigma_v$ *for all* $t \leq j \leq 2t-1$.

(vii) The conditions in (vi) determine σ *completely.*

Part (vi) of Lemma 15.20 completes our search for a decoding method, since σ determines $\mathcal{E}$, $\mathcal{E}$ determines $\mathbf{e}$ and $\mathbf{e}$ determines $\mathbf{c}$. It is worth noting that the system of equations in part (v) suffice to determine the pair σ and ω directly.

Compact disc players use BCH codes. Of course, errors are likely to occur in bursts (corresponding to scratches etc) and this is dealt with by distributing the bits (digits) in a single codeword over a much longer stretch of track. The code used can correct a burst of 4000 consecutive errors (2.5 mm of track).

Unfortunately, none of the codes we have considered work anywhere near the Shannon bound (see Theorem 10.1). We might suspect that this is because they are linear, but Elias has shown that this is not the case. (We just state the result without proof.)

Theorem 15.21. *In Theorem 10.1 we can replace 'code' by 'linear code'.*

The advance of computational power and the ingenuity of the discoverers[30] have lead to new codes which appear to come close to the Shannon bounds. But that is another story.

Just as pure algebra has contributed greatly to the study of error correcting codes, so the study of error correcting codes has contributed greatly to the study of pure algebra. The story of one such contribution is set out in T. M. Thompson's *From Error-correcting Codes through Sphere Packings to Simple Groups* [9] — a good, not too mathematical, account of the discovery of the last sporadic simple groups by Conway and others.

16 Shift registers

In this section we move towards cryptography, but the topic discussed will turn out to have connections with the decoding of BCH codes as well.

Definition 16.1. *A* general feedback shift register *is a map* $f : \mathbb{F}_2^d \to \mathbb{F}_2^d$ *given by*

$$f(x_0, x_1, \dots, x_{d-2}, x_{d-1}) = (x_1, x_2, \dots, x_{d-1}, C(x_0, x_1, \dots, x_{d-2}, x_{d-1}))$$

with C *a map* $C : \mathbb{F}_2^d \to \mathbb{F}_2$*. The* stream *associated to an* initial fill $(y_0, y_1, \dots, y_{d-1})$ *is the sequence*

$$y_0, y_1, \dots, y_j, y_{j+1}, \dots \text{ with } y_n = C(y_{n-d}, y_{n-d+1}, \dots, y_{n-1}) \text{ for all } n \geq d.$$

Example 16.2. *If the general feedback shift* f *given in Definition 16.1 is a permutation, then* C *is linear in the first variable, i.e.*

$$C(x_0, x_1, \dots, x_{d-2}, x_{d-1}) = x_0 + C'(x_1, x_2, \dots, x_{d-2}, x_{d-1}).$$

[30]People like David MacKay, now better known for his superb 'Sustainable Energy Without the Hot Air' — rush out and read it.

Definition 16.3. *We say that the function f of Definition 16.1 is a* linear feedback register *if*

$$C(x_0, x_1, \ldots, x_{d-1}) = a_0x_0 + a_1x_1 + \ldots + a_{d-1}x_{d-1},$$

with $a_0 = 1$.

Exercise 16.4. *Discuss briefly the effect of omitting the condition $a_0 = 1$ from Definition 16.3.*

The discussion of the linear recurrence

$$x_n = a_0x_{n-d} + a_1x_{n-d+1} + \cdots + a_{d-1}x_{n-1}$$

over $\mathbb{F}_2$ follows the IA discussion of the same problem over $\mathbb{R}$ but is complicated by the fact that

$$n^2 = n$$

in $\mathbb{F}_2$. We assume that $a_0 \neq 0$ and consider the *auxiliary polynomial*

$$C(X) = X^d - a_{d-1}X^{d-1} - \cdots - a_1X - a_0.$$

In the exercise below, $\binom{n}{v}$ is the appropriate polynomial in n.

Exercise 16.5. *Consider the linear recurrence*

$$x_n = a_0x_{n-d} + a_1x_{n-d+1} + \ldots + a_{d-1}x_{n-1} \qquad \bigstar$$

with $a_j \in \mathbb{F}_2$ and $a_0 \neq 0$.

(i) Suppose K is a field containing $\mathbb{F}_2$ such that the auxiliary polynomial C has a root α in K. Show that $x_n = \alpha^n$ is a solution of $\bigstar$ in K.

(ii) Suppose K is a field containing $\mathbb{F}_2$ such that the auxiliary polynomial C has d distinct roots α_1, α_2, ..., α_d in K. Show that the general solution of $\bigstar$ in K is

$$x_n = \sum_{j=1}^{d} b_j\alpha_j^n$$

for some $b_j \in K$. If $x_0, x_1, \ldots, x_{d-1} \in \mathbb{F}_2$, show that $x_n \in \mathbb{F}_2$ for all n.

(iii) Work out the first few lines of Pascal's triangle modulo 2. Show that the functions $f_j : \mathbb{Z} \to \mathbb{F}_2$

$$f_j(n) = \binom{n}{j}$$

are linearly independent in the sense that

$$\sum_{j=0}^{m} b_j f_j(n) = 0$$

for all n implies $b_j = 0$ for $0 \leq j \leq m$.

(iv) Suppose K is a field containing $\mathbb{F}_2$ such that the auxiliary polynomial C factorises completely into linear factors. If the root α_u has multiplicity $m(u)$, $[1 \leq u \leq q]$, show that the general solution of $\bigstar$ in K is

$$x_n = \sum_{u=1}^{q} \sum_{v=0}^{m(u)-1} b_{u,v} \binom{n}{v} \alpha_u^n$$

for some $b_{u,v} \in K$. If $x_0, x_1, \ldots, x_{d-1} \in \mathbb{F}_2$, show that $x_n \in \mathbb{F}_2$ for all n.

A strong link with the problem of BCH decoding is provided by Theorem 16.7 below.

Definition 16.6. *If we have a sequence (or stream) x_0, x_1, x_2, ... of elements of $\mathbb{F}_2$ then its* generating function *G is given by*

$$G(Z) = \sum_{n=0}^{\infty} x_j Z^j.$$

If the recurrence relation for a linear feedback generator is

$$\sum_{j=0}^{d} c_j x_{n-j} = 0$$

for $n \geq d$ with $c_0, c_d \neq 0$ we call

$$C(z) = \sum_{j=0}^{d} c_j Z^j$$

the auxiliary polynomial of the generator.

Theorem 16.7. *The stream (x_n) comes from a linear feedback generator with auxiliary polynomial C if and only if the generating function for the stream is (formally) of the form*

$$G(Z) = \frac{B(Z)}{C(Z)}$$

with B a polynomial of degree strictly smaller than that of C.

If we can recover C from G then we have recovered the linear feedback generator from the stream.

The link with BCH codes is established by looking at Lemma 15.20 (iii) and making the following remark.

Lemma 16.8. *If a stream (x_n) comes from a linear feedback generator with auxiliary polynomial C of degree d, then C is determined by the condition*

$$G(Z)C(Z) \equiv B(Z) \mod Z^{2d}$$

with B a polynomial of degree at most $d-1$.

We thus have the following problem.

Problem *Given a generating function G for a stream and knowing that*

$$G(Z) = \frac{B(Z)}{C(Z)}$$

with B a polynomial of degree less than that of C and the constant term in C is $c_0 = 1$, recover C.

The Berlekamp–Massey method In this method we do not assume that the degree d of C is known. The Berlekamp–Massey solution to this problem is based on the observation that, since

$$\sum_{j=0}^{d} c_j x_{n-j} = 0$$

(with $c_0 = 1$) for all $n \geq d$, we have

$$\begin{pmatrix} x_d & x_{d-1} & \dots & x_1 & x_0 \\ x_{d+1} & x_d & \dots & x_2 & x_1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{2d} & x_{2d-1} & \dots & x_{d+1} & x_d \end{pmatrix} \begin{pmatrix} 1 \\ c_1 \\ \vdots \\ c_d \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}. \qquad \bigstar$$

The Berlekamp–Massey method tells us to look successively at the matrices

$$A_1 = (x_0),\ A_2 = \begin{pmatrix} x_1 & x_0 \\ x_2 & x_1 \end{pmatrix},\ A_3 = \begin{pmatrix} x_2 & x_1 & x_0 \\ x_3 & x_2 & x_1 \\ x_4 & x_3 & x_2 \end{pmatrix}, \dots$$

starting at A_r if it is known that $r \geq d$. For each A_j we evaluate $\det A_j$. If $\det A_j \neq 0$, then $j-1 \neq d$. If $\det A_j = 0$, then $j-1$ is a good candidate for d so we solve ★ on the assumption that $d = j - 1$. (Note that a one dimensional subspace of $\mathbb{F}^{d+1}$ contains only one non-zero vector.) We then check our candidate for $(c_0, c_1, \dots, c_d)$ over as many terms of the stream as we wish. If it fails the test, we then know that $d \geq j$ and we start again[31].

As we have stated it, the Berlekamp–Massey method is not an algorithm in the strict sense of the term although it becomes one if we put an upper bound on the possible values of d. (A little thought shows that, if no upper bound is put on d, no algorithm is possible because, with a suitable initial stream, a linear feedback register with large d can be made to produce a stream whose initial values would be produced by a linear feedback register with much smaller d. For the same reason the Berlekamp–Massey method will produce the B of smallest degree which gives G and not necessarily the original B.) In practice, however, the Berlekamp–Massey method is very effective in cases when d is unknown.

By careful arrangement of the work it is possible to cut down considerably on the labour involved.

The solution of linear equations gives us a method of 'secret sharing'.

Problem 16.9. *It is not generally known that CMS when reversed forms the initials of of 'Secret Missile Command'. If the University is attacked by HEFCE*[32]*, the Faculty Board will retreat to a bunker known as Meeting Room 23. Entry to the room involves tapping out a positive integer S (the secret) known only to the Chairman of the Faculty Board. Each of the n members of the Faculty Board knows a certain pair of numbers (their shadow) and it is required that, in the absence of the Chairman, any k members of the Faculty can reconstruct S from their shadows, but no $k-1$ members can do so. How can this be done?*

Here is one neat solution. Suppose S must lie between 0 and N (it is sensible to choose S at random). The chairman chooses a prime $p > N, n$. She then chooses integers $a_1, a_2, \dots, a_{k-1}$ at random and distinct integers $x_1, x_2, \dots, x_n$ at random subject to $0 \leq a_j \leq p-1$, $1 \leq x_j \leq p-1$, sets $a_0 = S$ and computes

$$P(r) \equiv a_0 + a_1 x_r + a_2 x_r^2 + \cdots + a_{k-1} x_r^{k-1} \mod p$$

choosing $0 \leq P(r) \leq p-1$. She then gives the rth member of the Faculty Board the pair of numbers $\big(x_r, P(r)\big)$ (the shadow pair), to be kept secret

[31] Note that, over $\mathbb{F}_2$, $\det A_j$ can only take two values so there will be many false alarms. Note also that the determinant may be evaluated much faster using reduction to (rearranged) triangular form than by Cramer's rule and that once the system is in (rearranged) triangular form it is easy to solve the associated equations.

[32] An institution like SPECTRE but without the charm.

from everybody else) and tells everybody the value of p. She then burns her calculations.

Suppose that k members of the Faculty Board with shadow pairs $\big(y_j, Q(j)\big) = \big(x_{r_j}, P(r_j)\big)$ $[1 \le j \le k]$ are together. By the properties of the Van der Monde determinant (see Lemma 15.16)

$$\begin{vmatrix} 1 & y_1 & y_1^2 & \dots & y_1^{k-1} \\ 1 & y_2 & y_2^2 & \dots & y_2^{k-1} \\ 1 & y_3 & y_3^2 & \dots & y_3^{k-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & y_k & y_k^2 & \dots & y_k^{k-1} \end{vmatrix} \equiv \begin{vmatrix} 1 & 1 & 1 & \dots & 1 \\ y_1 & y_2 & y_3 & \dots & y_k \\ y_1^2 & y_2^2 & y_3^2 & \dots & y_k^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_1^{k-1} & y_2^{k-1} & y_3^{k-1} & \dots & y_k^{k-1} \end{vmatrix}$$
$$\equiv \prod_{1 \le j < i \le k-1} (y_i - y_j) \not\equiv 0 \mod p.$$

Thus the system of equations

$$\begin{aligned} z_0 + y_1 z_1 + y_1^2 z_2 + \dots + y_1^{k-1} z_{k-1} &\equiv Q_1 \\ z_0 + y_2 z_1 + y_2^2 z_2 + \dots + y_2^{k-1} z_{k-1} &\equiv Q_2 \\ z_0 + y_3 z_1 + y_3^2 z_2 + \dots + y_3^{k-1} z_{k-1} &\equiv Q_3 \\ &\vdots \\ z_0 + y_k z_1 + y_k^2 z_2 + \dots + y_k^{k-1} z_{k-1} &\equiv Q_k \end{aligned}$$

has a unique solution $\mathbf{z}$. But we know that $\mathbf{a}$ is a solution, so $\mathbf{z} = \mathbf{a}$ and the secret $S = z_0$.

On the other hand,

$$\begin{vmatrix} y_1 & y_1^2 & \dots & y_1^{k-1} \\ y_2 & y_2^2 & \dots & y_2^{k-1} \\ y_3 & y_3^2 & \dots & y_3^{k-1} \\ \vdots & \vdots & \ddots & \vdots \\ y_{k-1} & y_{k-1}^2 & \dots & y_{k-1}^{k-1} \end{vmatrix} \equiv y_1 y_2 \dots y_{k-1} \prod_{1 \le j < i \le k-1} (y_i - y_j) \not\equiv 0 \mod p,$$

so the system of equations

$$\begin{aligned} z_0 + y_1 z_1 + y_1^2 z_2 + \dots + y_1^{k-1} z_{k-2} &\equiv Q_1 \\ z_0 + y_2 z_1 + y_2^2 z_2 + \dots + y_2^{k-1} z_{k-2} &\equiv Q_2 \\ z_0 + y_3 z_1 + y_3^2 z_2 + \dots + y_3^{k-1} z_{k-2} &\equiv Q_3 \\ &\vdots \\ z_0 + y_{k-1} z_1 + y_{k-1}^2 z_2 + \dots + y_{k-1}^{k-1} z_{k-2} &\equiv Q_{k-1} \end{aligned}$$

has a solution, *whatever value of* z_0 *we take*, so $k-1$ members of the Faculty Board have no way of saying that any possible values of S is more likely than any other.

One way of looking at this method of 'secret sharing' is to note that a polynomial of degree $k-1$ can be recovered from its value at k points but not from its value at $k-1$ points. However, the proof that the method works needs to be substantially more careful.

Exercise 16.10. *Is the secret compromised if the values of the* x_j *become known?*

17 A short homily on cryptography

Cryptography is the science of code making. Cryptanalysis is the art of code breaking.

Two thousand years ago, Lucretius wrote that 'Only recently has the true nature of things been discovered'. In the same way, mathematicians are apt to feel that 'Only recently has the true nature of cryptography been discovered'. The new mathematical science of cryptography with its promise of codes which are 'provably hard to break' seems to make everything that has gone before irrelevant.

It should, however, be observed that the best cryptographic systems of our ancestors (such as diplomatic 'book codes') served their purpose of ensuring secrecy for a relatively small number of messages between a relatively small number of people extremely well. It is the modern requirement for secrecy *on an industrial scale* to cover endless streams of messages between many centres which has made necessary the modern science of cryptography.

More pertinently, it should be remembered that the German Naval Enigma codes not only appeared to be 'provably hard to break' (though not against the modern criteria of what this should mean) but, *considered in isolation*, probably were unbreakable in practice[33]. Fortunately the Submarine codes formed part of an 'Enigma system' with certain exploitable weaknesses. (For an account of how these weaknesses arose and how they were exploited see Kahn's *Seizing the Enigma* [4].)

Even the best codes are like the lock on a safe. However good the lock is, the safe may be broken open by brute force, or stolen together with its contents, or a key holder may be persuaded by fraud or force to open the lock, or the presumed contents of the safe may have been tampered with before they go into the safe, or .... The coding schemes we shall consider,

[33] Some versions remained unbroken until the end of the war.

are at best, cryptographic *elements* of larger possible cryptographic *systems*. The planning of cryptographic systems requires not only mathematics but also engineering, economics, psychology, humility and an ability to learn from past mistakes. Those who do not learn the lessons of history are condemned to repeat them.

In considering a cryptographic system, it is important to consider its purpose. Consider a message M sent by A to B. Here are some possible aims.

Secrecy A and B can be sure that no third party X can read the message M.

Integrity A and B can be sure that no third party X can alter the message M.

Authenticity B can be sure that A sent the message M.

Non-repudiation B can prove to a third party that A sent the message M.

When you fill out a cheque giving the sum both in numbers and words you are seeking to protect the *integrity* of the cheque. When you sign a traveller's cheque 'in the presence of the paying officer' the process is intended, from your point of view, to protect *authenticity* and, from the bank's point of view, to produce *non-repudiation*.

Another point to consider is the level of security aimed at. It hardly matters if a few people use forged tickets to travel on the underground, it does matter if a single unauthorised individual can gain privileged access to a bank's central computer system. If *secrecy* is aimed at, how long must the secret be kept? Some military and financial secrets need only remain secret for a few hours, others must remain secret for years.

We must also, to conclude this non-exhaustive list, consider the level of security required. Here are three possible levels.

(1) Prospective opponents should find it hard to compromise your system even if they are in possession of a plentiful supply of encoded messages C_i.

(2) Prospective opponents should find it hard to compromise your system even if they are in possession of a plentiful supply of pairs (M_i, C_i) of messages M_i together with their encodings C_i.

(3) Prospective opponents should find it hard to compromise your system even if they are allowed to produce messages M_i and given their encodings C_i.

Clearly, safety at level (3) implies safety at level (2) and safety at level (2) implies safety at level (1). Roughly speaking, the best Enigma codes satisfied (1). The German Navy believed on good but mistaken grounds that they satisfied (2). Level (3) would have appeared evidently impossible to attain until a few years ago. Nowadays, level (3) is considered a minimal requirement for a really secure system.

18 Stream ciphers

One natural way of enciphering is to use a *stream cipher*. We work with streams (that is, sequences) of elements of $\mathbb{F}_2$. We use a *cipher stream* k_0, k_1, k_2 .... The *plain text stream* p_0, p_1, p_2, ... is enciphered as the *cipher text stream* z_0, z_1, z_2, ... given by

$$z_n = p_n + k_n.$$

This is an example of a *private key* or *symmetric* system. The security of the system depends on a secret (in our case the cipher stream) $\mathbf{k}$ shared between the cipherer and the encipherer. Knowledge of an enciphering method makes it easy to work out a deciphering method and vice versa. In our case a deciphering method is given by the observation that

$$p_n = z_n + k_n.$$

(Indeed, writing $\alpha(\mathbf{p}) = \mathbf{p} + \mathbf{z}$, we see that the enciphering function α has the property that $\alpha^2 = \iota$ the identity map. Ciphers like this are called *symmetric*.)

In the one-time pad, first discussed by Vernam in 1926, the cipher stream is a random sequence $k_j = K_j$, where the K_j are independent random variables with

$$\Pr(K_j = 0) = \Pr(K_j = 1) = 1/2.$$

If we write $Z_j = p_j + K_j$, then we see that the Z_j are independent random variables with

$$\Pr(Z_j = 0) = \Pr(Z_j = 1) = 1/2.$$

Thus (in the absence of any knowledge of the ciphering stream) the codebreaker is just faced by a stream of perfectly random binary digits. Decipherment is impossible in principle.

It is sometimes said that it is hard to find random sequences, and it is, indeed, rather harder than might appear at first sight, but it is not too difficult to rig up a system for producing 'sufficiently random' sequences[34]. The secret services of the former Soviet Union were particularly fond of one-time pads. The real difficulty lies in the necessity for sharing the secret

[34] Take ten of your favourite long books, convert to binary sequences $x_{j,n}$ and set $k_n = \sum_{j=1}^{10} x_{j,1000+j+n} + s_n$ where s_n is the output of your favourite 'pseudo-random number generator' (in this connection see Exercise 27.16). Give a memory stick with a copy of $\mathbf{k}$ to your friend and, provided both of you obey some elementary rules, your correspondence will be safe from MI5. The anguished debate in the US about codes and privacy refers to the privacy of large organisations and their clients, not the privacy of communication from individual to individual.

sequence $\mathbf{k}$. If a random sequence is reused it ceases to be random (it becomes 'the same code as last Wednesday' or the 'the same code as Paris uses') so, when there is a great deal of code traffic[35], new one-time pads must be sent out. If random bits can be safely communicated, so can ordinary messages and the exercise becomes pointless.

In practice, we would like to start from a short shared secret 'seed' and generate a ciphering string $\mathbf{k}$ that 'behaves like a random sequence'. This leads us straight into deep philosophical waters[36]. As might be expected, there is an illuminating discussion in Chapter III of Knuth's marvellous *The Art of Computing Programming* [7]. Note, in particular, his warning:

> *...random numbers should not be generated with a method chosen at random.* Some theory should be used.

One way that we might try to generate our ciphering string is to use a general feedback shift register f of length d with the initial fill $(k_0, k_1, \dots, k_{d-1})$ as the secret seed.

Lemma 18.1. *If f is a general feedback shift register of length d, then, given any initial fill $(k_0, k_1, \dots, k_{d-1})$, there will exist $N, M \leq 2^d$ such that the output stream $\mathbf{k}$ satisfies $k_{r+N} = k_r$ for all $r \geq M$.*

Exercise 18.2. *Show that the decimal expansion of a rational number must be a recurrent expansion. Give a bound for the period in terms of the quotient. Conversely, by considering geometric series, or otherwise, show that a recurrent decimal represents a rational number.*

Lemma 18.3. *Suppose that f is a linear feedback register of length d.*

(i) $f(x_0, x_1, \dots, x_{d-1}) = (x_0, x_1, \dots, x_{d-1})$ if $(x_0, x_1, \dots, x_{d-1}) = (0, 0, \dots, 0)$.

(ii) Given any initial fill $(k_0, k_1, \dots, k_{d-1})$, there will exist $N, M \leq 2^d - 1$ such that the output stream $\mathbf{k}$ satisfies $k_{r+N} = k_r$ for all $r \geq M$.

We can complement Lemma 18.3 by using Lemma 14.16 and the associated discussion.

---

[35] In 1941, the Soviet Union's need for one-time pads suddenly increased and it appears that pages were reused in different pads. If the reader reflects, she will see that, though this is a mistake, it is one which it is very difficult to exploit. However, under the pressure of the cold war, US code-breakers managed to decode messages which, although several years old, still provided useful information. After 1944, the Soviet Union's one-time pads became genuinely one-time again and the coded messages became indecipherable.

[36] Where we drown at once, since the best (at least, in my opinion) modern view is that any sequence that can be generated by a program of reasonable length from a 'seed' of reasonable size is automatically non-random.

Lemma 18.4. *A linear feedback register of length d attains its maximal period $2^d - 1$ (for a non-trivial initial fill) when the roots of the auxiliary polynomial*[37] *are primitive elements of $\mathbb{F}_{2^d}$.*

(We will note why this result is plausible, but we will not prove it. See Exercise 27.19 for a proof.)

It is well known that short period streams are dangerous. During World War II the British Navy used codes whose period was adequately long for peace time use. The massive increase in traffic required by war time conditions meant that the period was now too short. By dint of immense toil, German naval code breakers were able to identify coincidences and crack the British codes.

Unfortunately, whilst short periods are definitely unsafe, it does not follow that long periods guarantee safety. Using the Berlekamp–Massey method we see that stream codes based on linear feedback registers are unsafe at level (2).

Lemma 18.5. *Suppose that an unknown* cipher stream $k_0, k_1, k_2 \dots$ *is produced by an unknown linear feedback register f of unknown length $d \leq D$. The* plain text stream $p_0, p_1, p_2, \dots$ *is enciphered as the* cipher text stream $z_0, z_1, z_2, \dots$ *given by*

$$z_n = p_n + k_n.$$

If we are given $p_0, p_1, \dots p_{2D-1}$ and $z_0, z_1, \dots z_{2D-1}$ then we can find k_r for all r.

Thus if we have a message of length twice the length of the linear feedback register together with its encipherment the code is broken.

It is easy to construct immensely complicated looking linear feedback registers with hundreds of registers. Lemma 18.5 shows that, from the point of view of a determined, well equipped and technically competent opponent, cryptographic systems based on such registers are the equivalent of leaving your house key hidden under the door mat. Professionals say that such systems seek 'security by obscurity'.

However, if you do not wish to baffle the CIA, but merely prevent little old ladies in tennis shoes watching subscription television without paying for it, systems based on linear feedback registers are cheap and quite effective. Whatever they may say in public, large companies are happy to tolerate a certain level of fraud. So long as 99.9% of the calls made are paid for, the

[37] In this sort of context we shall sometimes refer to the 'auxiliary polynomial' as the 'feedback polynomial'.

profits of a telephone company are essentially unaffected by the .1% which 'break the system'.

What happens if we try some simple tricks to increase the complexity of the cipher text stream?

Lemma 18.6. *If x_n is a stream produced by a linear feedback system of length N with auxiliary polynomial P and y_n is a stream produced by a linear feedback system of length M with auxiliary polynomial Q, then $x_n + y_n$ is a stream produced by a linear feedback system of length $N + M$ with auxiliary polynomial $P(X)Q(X)$.*

Note that this means that adding streams from two linear feedback system is no more economical than producing the same effect with one. Indeed the situation may be worse since *a stream produced by linear feedback system of given length may, possibly, also be produced by another linear feedback system of shorter length.*

Lemma 18.7. *Suppose that x_n is a stream produced by a linear feedback system of length N with auxiliary polynomial P and y_n is a stream produced by a linear feedback system of length M with auxiliary polynomial Q. Let P have roots α_1, α_2, $\dots \alpha_N$ and Q have roots β_1, β_2, $\dots \beta_M$ over some field $K \supseteq \mathbb{F}_2$. Then $x_n y_n$ is a stream produced by a linear feedback system of length NM with auxiliary polynomial*

$$\prod_{1\leq i\leq N}\prod_{1\leq i\leq M}(X - \alpha_i\beta_j).$$

We shall probably only prove Lemmas 18.6 and 18.7 in the case when all roots are distinct, leaving the more general case as an easy exercise. We shall also not prove that the polynomial $\prod_{1\leq i\leq N}\prod_{1\leq i\leq M}(X-\alpha_i\beta_j)$ obtained in Lemma 18.7 actually lies in $\mathbb{F}_2[X]$ but (for those who are familiar with the phrase in quotes) this is an easy exercise in 'symmetric functions of roots'.

Here is an even easier remark.

Lemma 18.8. *Suppose that x_n is a stream which is periodic with period N and y_n is a stream which is periodic with period M. Then the streams $x_n + y_n$ and $x_n y_n$ are periodic with periods dividing the lowest common multiple of N and M.*

Exercise 18.9. *One of the most confidential German codes (called FISH by the British) involved a complex mechanism which the British found could be simulated by two loops of paper tape of length* 1501 *and* 1497. *If $k_n = x_n + y_n$ where x_n is a stream of period* 1501 *and y_n is a stream of period* 1497, *what* *is the longest possible period of k_n? How many consecutive values of k_n would you need to to find the underlying linear feedback register using the Berlekamp–Massey method if you did not have the information given in the question? If you had all the information given in the question how many values of k_n would you need? (Hint, look at $x_{n+1497} - x_n$.)*

You have shown that, given k_n for sufficiently many consecutive n we can find k_n for all n. Can you find x_n for all n?

It might be thought that the lengthening of the underlying linear feedback system obtained in Lemma 18.7 is worth having, but it is bought at a substantial price. Let me illustrate this by an informal argument. Suppose we have 10 streams $x_{j,n}$ (without any peculiar properties) produced by linear feedback registers of length about 100. If we form $k_n = \prod_{j=1}^{10} x_{j,n}$, then the Berlekamp–Massey method requires of the order of 10^{20} consecutive values of k_n and the periodicity of k_n can be made still more astronomical. Our cipher key stream k_n appears safe from prying eyes. However it is doubtful if the prying eyes will mind. Observe that (under reasonable conditions) about 2^{-1} of the $x_{j,n}$ will have the value 1 and about 2^{-10} of the $k_n = \prod_{j=1}^{10} x_{j,n}$ will have value 1. Thus, if $z_n = p_n + k_n$, in more than 999 cases out of a 1000 we will have $z_n = p_n$. Even if we just combine two streams x_n and y_n in the way suggested we may expect $x_n y_n = 0$ for about 75% of the time.

Here is another example where the apparent complexity of the cipher key stream is substantially greater than its true complexity.

Example 18.10. *The following is a simplified version of a standard satellite TV decoder. We have 3 streams x_n, y_n, z_n produced by linear feedback registers. If the cipher key stream is defined by*

$$\begin{aligned} k_n &= x_n \qquad \text{if } z_n = 0, \\ k_n &= y_n \qquad \text{if } z_n = 1, \end{aligned}$$

then

$$k_n = (y_n + x_n) z_n + x_n$$

and the cipher key stream is that produced by linear feedback register.

We must not jump to the conclusion that the best way round these difficulties is to use a non-linear feedback generator f. This is not the easy way out that it appears. If chosen by an amateur, the complicated looking f so produced will have the apparent advantage that we do not know what is wrong with it and the very real disadvantage that we do not know what is wrong with it.

Another approach is to observe that, so far as the potential code breaker is concerned, the cipher stream method only combines the 'unknown secret' (here the feedback generator f together with the seed $(k_0, k_1, \ldots, k_{d-1})$) with the unknown message $\mathbf{p}$ in a rather simple way. It might be better to consider a system with two functions $F : \mathbb{F}_2^m \times \mathbb{F}_2^n \to \mathbb{F}_2^q$ and $G : \mathbb{F}_2^m \times \mathbb{F}_2^q \to \mathbb{F}_2^n$ such that

$$G(\mathbf{k}, F(\mathbf{k}, \mathbf{p})) = \mathbf{p}.$$

Here $\mathbf{k}$ will be the shared secret, $\mathbf{p}$ the message, and $\mathbf{z} = F(\mathbf{k}, \mathbf{p})$ the encoded message which can be decoded by using the fact that $G(\mathbf{k}, \mathbf{z}) = \mathbf{p}$.

In the next section we shall see that an even better arrangement is possible. However, arrangements like this have the disadvantage that the message $\mathbf{p}$ must be entirely known before it is transmitted and the encoded message $\mathbf{z}$ must have been entirely received before it can be decoded. Stream ciphers have the advantage that they can be decoded 'on the fly'. They are also much more error tolerant. A mistake in the coding, transmission or decoding of a single element only produces an error in a single place of the sequence. There will continue to be circumstances where stream ciphers are appropriate.

There is one further remark to be made. Suppose, as is often the case, that we know F, that $n = q$ and we know the 'encoded message' $\mathbf{z}$. Suppose also that we know that the 'unknown secret' or 'key' $\mathbf{k} \in \mathcal{K} \subseteq \mathbb{F}_2^m$ and the 'unknown message' $\mathbf{p} \in \mathcal{P} \subseteq \mathbb{F}_2^n$. We are then faced with the problem:- Solve the system

$$\mathbf{z} = F(\mathbf{k}, \mathbf{p}) \text{ where } \mathbf{k} \in \mathcal{K},\ \mathbf{p} \in \mathcal{P}. \qquad \bigstar$$

Speaking roughly, the task is hopeless unless $\bigstar$ has a unique solution[38] Speaking even more roughly, this is unlikely to happen if $|\mathcal{K}||\mathcal{P}| > 2^n$ and is likely to happen if 2^n is substantially greater than $|\mathcal{K}||\mathcal{P}|$. (Here, as usual, $|\mathcal{B}|$ denotes the number of elements of $\mathcal{B}$.)

Now recall the definition of the information rate given in Definition 6.2. If the message set $\mathcal{M}$ has information rate μ and the key set (that is the shared secret set) $\mathcal{K}$ has information rate κ, then, taking logarithms, we see that, if

$$n - m\kappa - n\mu$$

[38]'According to some, the primordial Torah was inscribed in black flames on white fire. At the moment of its creation, it appeared as a series of letters not yet joined up in the form of words. For this reason, in the Torah rolls there appear neither vowels nor punctuation, nor accents; for the original Torah was nothing but a disordered heap of letters. Furthermore, had it not been for Adam's sin, these letters might have been joined differently to form another story. For the kabalist, God will abolish the present ordering of the letters, or else will teach us how to read them according to a new disposition only after the coming of the Messiah.' ([1], Chapter 2.) A reader of this footnote has directed me to the *International Torah Codes Society*.

is substantially greater than 0, then ★ is likely to have a unique solution, but, if it is substantially smaller, this is unlikely.

Example 18.11. *Suppose that, instead of using binary code, we consider an alphabet of 27 letters (the English alphabet plus a space). We must take logarithms to the base 27, but the considerations above continue to apply. The English language treated in this way has information rate about .4. (This is very much a ball park figure. The information rate is certainly less than .5 and almost certainly greater than .2.)*

(i) In the Caesar code, we replace the ith element of our alphabet by the $i+j$th (modulo 27). The shared secret is a single letter (the code for A say). We have $m = 1$, $\kappa = 1$ and $\mu \approx .4$. Thus

$$n - m\kappa - n\mu \approx .6n - 1.$$

If $n = 1$ (so $n - m\kappa - n\mu \approx -.4$) it is obviously impossible to decode the message. If $n = 10$ (so $n - m\kappa - n\mu \approx 5$) a simple search through the 27 possibilities will almost always give a single possible decode.

(ii) In a simple substitution code, a permutation of the alphabet is chosen and applied to each letter of the code in turn. The shared secret is a sequence of 26 letters (given the coding of the first 26 letters, the 27th can then be deduced). We have $m = 26$, $\kappa = 1$ and $\mu \approx .4$. Thus

$$n - m\kappa - n\mu \approx .6n - 26.$$

In The Dancing Men, *Sherlock Holmes solves such a code with $n = 68$ (so $n - m\kappa - n\mu \approx 15$) without straining the reader's credulity too much and I would think that, unless the message is very carefully chosen, most of my audience could solve such a code with $n = 200$ (so $n - m\kappa - n\mu \approx 100$).*

(iii) In the one-time pad $m = n$ and $\kappa = 1$, so (if $\mu > 0$)

$$n - m\kappa - n\mu = -n\mu \to -\infty$$

as $n \to \infty$.

(iv) Note that the larger μ is, the slower $n - m\kappa - n\mu$ increases. This corresponds to the very general statement that the higher the information rate of the messages, the harder it is to break the code in which they are sent.

The ideas just introduced can be formalised by the notion of unicity distance.

Definition 18.12. *The* unicity distance *of a code is the number of bits of message required to exceed the number of bits of information in the key plus the number of bits of information in the message.*

(The notion of information content brings us back to Shannon whose paper *Communication theory of secrecy systems*[39], published in 1949, forms the first modern treatment of cryptography in the open literature.)

If we only use our code once to send a message which is substantially shorter than the unicity distance, we can be confident that no code breaker, however gifted, could break it, simply because there is no unambiguous decode. (A one-time pad has unicity distance infinity.) However, the fact that there is a unique solution to a problem does not mean that it is easy to find. We have excellent reasons, some of which are spelled out in the next section, to believe that there exist codes for which the unicity distance is essentially irrelevant to the maximum safe length of a message. For these codes, even though there may be a unique solution, the amount of work required to find the solutions makes (it is hoped) any attempt impractical.

19 Asymmetric systems

Towards the end of the previous section, we discussed a general coding scheme depending on a shared secret key $\mathbf{k}$ known to the encoder and the decoder. The scheme can be generalised still further by splitting the secret in two. Consider a system with two functions $F : \mathbb{F}_2^m \times \mathbb{F}_2^n \to \mathbb{F}_2^q$ and $G : \mathbb{F}_2^p \times \mathbb{F}_2^q \to \mathbb{F}_2^n$ such that

$$G(\mathbf{l}, F(\mathbf{k}, \mathbf{p})) = \mathbf{p}.$$

Here $(\mathbf{k}, \mathbf{l})$ will be be a pair of secrets, $\mathbf{p}$ the message and $\mathbf{z} = F(\mathbf{k}, \mathbf{p})$ the encoded message which can be decoded by using the fact that $G(\mathbf{l}, \mathbf{z}) = \mathbf{p}$. In this scheme, the encoder must know $\mathbf{k}$, but need not know $\mathbf{l}$ and the decoder must know $\mathbf{l}$, but need not know $\mathbf{k}$. Such a system is called asymmetric.

So far the idea is interesting but not exciting. Suppose, however, that we can show that

(i) knowing F, G and $\mathbf{k}$ it is very hard to find $\mathbf{l}$

(ii) if we do not know $\mathbf{l}$ then, even if we know F, G and $\mathbf{k}$, it is very hard to find $\mathbf{p}$ from $F(\mathbf{k}, \mathbf{p})$.

Then the code is secure at what we called level (3).

Lemma 19.1. *Suppose that the conditions specified above hold. Then an opponent who is entitled to demand the encodings $\mathbf{z}_i$ of any messages $\mathbf{p}_i$ they choose to specify will still find it very hard to find $\mathbf{p}$ when given $F(\mathbf{k}, \mathbf{p})$.*

Let us write $F(\mathbf{k}, \mathbf{p}) = \mathbf{p}^{K_A}$ and $G(\mathbf{l}, \mathbf{z}) = \mathbf{z}^{K_A^{-1}}$ and think of $\mathbf{p}^{K_A}$ as participant A's encipherment of $\mathbf{p}$ and $\mathbf{z}^{K_A^{-1}}$ as participant B's decipherment

[39] Available on the web and in his *Collected Papers*.

of $\mathbf{z}$. We then have
$$(\mathbf{p}^{K_A})^{K_A^{-1}} = \mathbf{p}.$$
Lemma 19.1 tells us that such a system is secure however many messages are sent. Moreover, if we think of A as a spy-master, he can broadcast K_A to the world (that is why such systems are called public key systems) and invite anybody who wants to spy for him to send him secret messages in total confidence[40].

It is all very well to describe such a code, but do they exist? There is very strong evidence that they do, but, so far, all mathematicians have been able to do is to show that provided certain mathematical problems which are believed to be hard are indeed hard, then good codes exist.

The following problem is believed to be hard.

Problem *Given an integer N, which is known to be the product $N = pq$ of two primes p and q, find p and q.*

Several schemes have been proposed based on the assumption that this factorisation is hard. (Note, however, that it is easy to find large 'random' primes p and q.) We give a very elegant scheme due to Rabin and Williams. It makes use of some simple number theoretic results from IA and IB.

The reader may well have seen the following results before. In any case, they are easy to obtain by considering primitive roots.

Lemma 19.2. *If p is an odd prime the congruence*
$$x^2 \equiv d \mod p$$
is soluble if and only if $d \equiv 0$ or $d^{(p-1)/2} \equiv 1$ modulo p.

Lemma 19.3. *Suppose p is a prime such that $p = 4k - 1$ for some integer k. Then, if the congruence*
$$x^2 \equiv d \mod p$$
has any solution, it has d^k as a solution.

We now call on the Chinese remainder theorem.

Lemma 19.4. *Let p and q be primes of the form $4k - 1$ and set $N = pq$. Then the following two problems are of equivalent difficulty.*

(A) Given N and d find all the m satisfying
$$m^2 \equiv d \mod N.$$
(B) Given N find p and q.

[40] Although we make statements about certain codes along the lines of 'It does not matter who knows this', you should remember the German naval saying 'All radio traffic is high treason'. If any aspect of a code can be kept secret, it should be kept secret.

(Note that, provided that $d \not\equiv 0$, knowing the solution to (A) for any d gives us the four solutions for the case $d = 1$.) The result is also true but much harder to prove for general primes p and q.

At the risk of giving aid and comfort to followers of the Lakatosian heresy, it must be admitted that the statement of Lemma 19.4 does not really tell us what the result we are proving is, although the proof makes it clear that the result (whatever it may be) is certainly true. However, with more work, everything can be made precise.

We can now give the Rabin–Williams scheme. The spy-master A selects two very large primes p and q. (Since he has only done an undergraduate course in mathematics, he will take p and q of the form $4k - 1$.) He keeps the pair (p, q) secret, but broadcasts the public key $N = pq$. If B wants to send him a message, she writes it in binary code and splits it into blocks of length m with $2^m < N < 2^{m+1}$. Each of these blocks is a number r_j with $0 \leq r_j < N$. B computes s_j such that $r_j^2 \equiv s_j$ modulo N and sends s_j. The spy-master (who knows p and q) can use the method of Lemma 19.4 to find one of four possible values for r_j (the four square roots of s_j). Of these four possible message blocks it is almost certain that three will be garbage, so the fourth will be the desired message.

If the reader reflects, she will see that the ambiguity of the root is genuinely unproblematic. (If the decoding is mechanical then fixing 50 bits scattered throughout each block will reduce the risk of ambiguity to negligible proportions.) Slightly more problematic, from the practical point of view, is the possibility that someone could be known to have sent a very short message, that is to have started with an m such that $1 \leq m \leq N^{1/2}$ but, provided sensible precautions are taken, this should not occur.

If I Google 'Casino', then I am instantly put in touch with several of the world's 'most trusted electronic casinos' who subscribe to 'responsible gambling' and who have their absolute probity established by 'internationally recognised Accredited Test Facilities'. Given these assurances, it seems churlish to introduce Alice and Bob who live in different cities, can only communicate by e-mail and are so suspicious of each other that neither will accept the word of the other as to the outcome of the toss of a coin.

If, in spite of this difficulty, Alice and Bob wish to play heads and tails (the technical expression is 'bit exchange' or 'bit sharing'), then the ambiguity of the Rabin–Williams scheme becomes an advantage. Let us set out the steps of a 'bit sharing scheme' based on Rabin–Williams.

STEP 1 Alice chooses at random two large primes p and q such that $p \equiv q \equiv 3 \mod 4$. She computes $n = pq$ and sends n to Bob.

STEP 2 Bob chooses a random integer r with $1 < r < n/2$. (He wishes to hide r from Alice, so he may take whatever other precautions he wishes in

choosing r.) He computes $m \equiv r^2 \mod n$ and sends m to Alice.
STEP 3 Since Alice knows p and q she can easily compute the 4 square roots of m modulo n. Exactly two of the roots r_1 and r_2 will satisfy $1 < r < n/2$. (If s is a root, so is $-s$.) However, Alice has no means of telling which is r. Alice writes out r_1 and r_2 in binary and chooses a place (the kth digit say) where they differ. She then tells Bob 'I choose the value u for the kth bit'.
STEP 4 Bob tells Alice the value of r. If the value of the kth bit of r is u, then Alice wins. If not, Bob wins. Alice checks that $r^2 \equiv m \mod n$. Since, $r_1 r_2^{-!}$ is a square root of unity which is neither 1 nor -1, knowing r_1 and r_2 is equivalent to factoring n, she knows that Bob could not lie about the value of r. Thus Alice is happy.
STEP 5 Alice tells Bob the values of p and q. He checks that p and q are primes (see Exercise 27.12 for why he does this) and finds r_1 and r_2. After Bob has verified that r_1 and r_2 do indeed differ in the kth bit, he also is happy, since there is no way Alice could know from inspection of m which root he started with.

20 Commutative public key systems

In the previous sections we introduced the coding and decoding functions K_A and K_A^{-1} with the property that

$$(\mathbf{p}^{K_A})^{K_A^{-1}} = \mathbf{p},$$

and satisfying the condition that knowledge of K_A did not help very much in finding K_A^{-1}. We usually require, in addition, that our system be *commutative* in the sense that

$$(\mathbf{p}^{K_A^{-1}})^{K_A} = \mathbf{p}.$$

and that knowledge of K_A^{-1} does not help very much in finding K_A. The Rabin–Williams scheme, as described in the last section, does not have this property.

Commutative public key codes are very flexible and provide us with simple means for maintaining integrity, authenticity and non-repudiation. (This is not to say that non-commutative codes can not do the same; simply that commutativity makes many things easier.)

Integrity and non-repudiation Let A 'own a code', that is know both K_A and K_A^{-1}. Then A can broadcast K_A^{-1} to everybody so that everybody can decode but only A can encode. (We say that K_A^{-1} is the *public key* and K_A the *private key*.) Then, for example, A could issue tickets to the castle ball carrying the coded message 'admit Joe Bloggs' which could be read by the recipients and the guards but would be unforgeable. However, for the same reason, A could not deny that he had issued the invitation.

Authenticity If B wants to be sure that A is sending a message then B can send A a harmless random message $\mathbf{q}$. If B receives back a message $\mathbf{p}$ such that $\mathbf{p}^{K_A^{-1}}$ ends with the message $\mathbf{q}$ then A must have sent it to B. (Anybody can *copy* a coded message but only A can control the content.)

Signature Suppose now that B also owns a commutative code pair (K_B, K_B^{-1}) and has broadcast K_B^{-1}. If A wants to send a message $\mathbf{p}$ to B he computes $\mathbf{q} = \mathbf{p}^{K_A}$ and sends $\mathbf{p}^{K_B^{-1}}$ followed by $\mathbf{q}^{K_B^{-1}}$. B can now use the fact that

$$(\mathbf{q}^{K_B^{-1}})^{K_B} = \mathbf{q}$$

to recover $\mathbf{p}$ and $\mathbf{q}$. B then observes that $\mathbf{q}^{K_A^{-1}} = \mathbf{p}$. Since only A can produce a pair $(\mathbf{p}, \mathbf{q})$ with this property, A must have written it.

There is now a charming little branch of the mathematical literature based on these ideas in which Albert gets Bertha to authenticate a message from Caroline to David using information from Eveline, Fitzpatrick, Gilbert and Harriet whilst Ingrid, Jacob, Katherine and Laszlo play bridge without using a pack of cards. However, a cryptographic system is only as strong as its weakest link. Unbreakable password systems do not prevent computer systems being regularly penetrated by 'hackers' and however 'secure' a transaction on the net may be it can still involve a rogue at one end and a fool at the other.

The most famous candidate for a commutative public key system is the RSA (Rivest, Shamir, Adleman) system. It was the RSA system[41] that first convinced the mathematical community that public key systems might be feasible. The reader will have met the RSA in IA, but we will push the ideas a little bit further.

Lemma 20.1. *Let p and q be primes. If $N = pq$ and $\lambda(N) = \operatorname{lcm}(p-1, q-1)$, then*

$$M^{\lambda(N)} \equiv 1 \pmod N$$

for all integers M coprime to N.

[41] A truly patriotic lecturer would refer to the ECW system, since Ellis, Cocks and Williamson discovered the system earlier. However, they worked for GCHQ and their work was kept secret.

Since we wish to appeal to Lemma 19.4, we shall assume in what follows that we have secretly chosen large primes p and q. We choose an integer e and then use Euclid's algorithm to check that e and $\lambda(N)$ are coprime and to find an integer d such that

$$de \equiv 1 \pmod{\lambda(N)}.$$

If Euclid's algorithm reveals that e and $\lambda(N)$ are not coprime, we try another e. Since others may be better psychologists than we are, we would be wise to use some sort of random method for choosing p, q and e.

The public key includes the value of e and N, but we keep secret the value of d. Given a number M with $1 \leq M \leq N-1$, we encode it as the integer E with $1 \leq E \leq N-1$

$$E \equiv M^d \pmod{N}.$$

The *public* decoding method is given by the observation that

$$E^e \equiv M^{de} \equiv M$$

for M coprime to N. (The probability that M is not coprime to N is so small that it can be neglected.) As was observed in IA, high powers are easy to compute.

Exercise 20.2. *Show how M^{2^n} can be computed using n multiplications. If $1 \leq r \leq 2^n$ show how M^r can be computed using at most $2n$ multiplications.*

To show that (providing that factoring N is indeed hard) finding d from e and N is hard we use the following lemma.

Lemma 20.3. *Suppose that d, e and N are as above. Set $de-1 = 2^a b$ where b is odd.*

(i) $a \geq 1$.

(ii) If $y \equiv x^b \pmod{N}$ and $y \not\equiv 1$ then there exists an r with $0 \leq r \leq a-1$ such that

$$z = y^{2^r} \not\equiv 1 \text{ but } z^2 \equiv 1 \pmod{N}.$$

Combined with Lemma 19.4, the idea of Lemma 20.3 gives a fast *probabilistic algorithm* where, by making random choices of x, we very rapidly reduce the probability that we can not find p and q to as close to zero as we wish.

Lemma 20.4. *The problem of finding d from the public information e and N is essentially as hard as factorising N.*

Remark 1 At first glance, we seem to have done as well for the RSA code as for the Rabin–Williams code. But this is not so. In Lemma 19.4 we showed that finding the four solutions of $M^2 \equiv E \pmod{N}$ was equivalent to factorising N. In the absence of further information, finding one root is as hard as finding another. Thus the ability to break the Rabin-Williams code (without some tremendous stroke of luck) is equivalent to the ability to factor N. On the other hand it is, a priori, possible that someone may find a decoding method for the RSA code which does not involve knowing d. They would have broken the RSA code without finding d. It must, however, be said that, in spite of this problem, the RSA code is much used in practice and the Rabin–Williams code is not.

Remark 2 It is natural to ask what evidence there is that the factorisation problem really is hard. Properly organised, trial division requires $O(N^{1/2})$ operations to factorise a number N. This order of magnitude was not bettered until 1972 when Lehman produced a $O(N^{1/3})$ method. In 1974, Pollard[42] produced a $O(N^{1/4})$ method. In 1979, as interest in the problem grew because of its connection with secret codes, Lenstra made a breakthrough to a $O(e^{c((\log N)(\log\log N))^{1/2}})$ method with $c \approx 2$. Since then some progress has been made (Pollard reached $O(e^{2((\log N)(\log\log N))^{1/3}})$ but, in spite of intense efforts, mathematicians have not produced anything which would be a real threat to codes based on the factorisation problem. A series of challenge numbers is hosted on the Wikipedia article entitled RSA. In 1996, it was possible to factor 100 (decimal) digit numbers routinely, 150 digit numbers with immense effort but 200 digit numbers were out of reach. In May 2005, the 200 digit challenge number was factored by F. Bahr, M. Boehm, J. Franke

[42] Although mathematically trained, Pollard worked outside the professional mathematical community.

and T. Kleinjunge as follows

$$
\begin{aligned}
&2799783391122132787082946763872260162107044678695542853756000992932612840010\\
&7609345671052955360856061822351910951365788637105954482006576775098580557613\\
&57909873495014417886317894629518723786\\
&9221823983\\
&= 3532461934402770121272604978198464368671197400197625023649303468776121253679423200058547956528088349\\
&\times 7925869954478333033347085841480059687737975857364219960734330341455767872818152135381409304740185467
\end{aligned}
$$

but the 210 digit challenge

$$
\begin{aligned}
&245246644900278211976517663573088018467026787678332759743414451715061600830038587216952208399332071549103626827191679864079776723243005600592035631246561218465817904100131859299619933817012149335034875870551067
\end{aligned}
$$

remains (as of mid-2008) unfactored. Organisations which use the RSA and related systems rely on 'security through publicity'. Because the problem of cracking RSA codes is so notorious, any breakthrough is likely to be publicly announced[43]. Moreover, even if a breakthrough occurs, it is unlikely to be one which can be easily exploited by the average criminal. So long as the secrets covered by RSA-type codes need only be kept for a few months rather than forever[44], the codes can be considered to be one of the strongest links in the security chain.

[43] And if not, is most likely to be a government rather than a Mafia secret.

[44] If a sufficiently robust 'quantum computer' could be built, then it could solve the factorisation problem and the discrete logarithm problem (mentioned later) with high probability extremely fast. It is highly unlikely that such a machine would be or could be kept secret, since it would have many more important applications than code breaking.

21 Trapdoors and signatures

It might be thought that secure codes are all that are needed to ensure the security of communications, but this is not so. It is not necessary to read a message to derive information from it[45]. In the same way, it may not be necessary to be able to write a message in order to tamper with it.

Here is a somewhat far fetched but worrying example. Suppose that, by wire tapping or by looking over people's shoulders, I discover that a bank creates messages in the form M_1, M_2 where M_1 is the name of the client and M_2 is the sum to be transferred to the client's account. The messages are then encoded according to the RSA scheme discussed after Lemma 20.1 as $Z_1 = M_1^d$ and $Z_2 = M_2^d$. I then enter into a transaction with the bank which adds \$ 1000 to my account. I observe the resulting Z_1 and Z_2 and then transmit Z_1 followed by Z_2^3.

Example 21.1. *What will (I hope) be the result of this transaction?*

We say that the RSA scheme is vulnerable to 'homomorphism attack' that is to say an attack which makes use of the fact our code is a homomorphism. (If $\theta(M) = M^d$, then $\theta(M_1M_2) = \theta(M_1)\theta(M_2)$.)

One way of increasing security against tampering is to first code our message by a classical coding method and then use our RSA (or similar) scheme on the result.

Exercise 21.2. *Discuss briefly the effect of first using an RSA scheme and then a classical code.*

However there is another way forward which has the advantage of wider applicability since it also can be used to protect the integrity of open (non-coded) messages and to produce password systems. These are the so-called *signature systems*. (Note that we shall be concerned with the 'signature of the message' and not the signature of the sender.)

Definition 21.3. *A* signature *or* trapdoor *or* hashing *function is a mapping $H : \mathcal{M} \to \mathcal{S}$ from the space $\mathcal{M}$ of possible messages to the space $\mathcal{S}$ of possible signatures.*

(Let me admit, at once, that Definition 21.3 is more of a statement of notation than a useful definition.) The first requirement of a good signature function is that the space $\mathcal{M}$ should be much larger than the space $\mathcal{S}$ so that H is a many-to-one function (in fact a great-many-to-one function) and we can not

[45] During World War II, British bomber crews used to spend the morning before a night raid testing their equipment, this included the radios.

work back from $H(M)$ to M. The second requirement is that $\mathcal{S}$ should be large so that a forger can not (sensibly) hope to hit on $H(M)$ by luck.

Obviously we should aim at the same kind of security as that offered by our 'level 2' for codes:-

> Prospective opponents should find it hard to find $H(M)$ given M even if they are in possession of a plentiful supply of message–signature pairs $(M_i, H(M_i))$ of messages M_i together with their encodings C_i.

I leave it to the reader to think about level 3 security (or to look at section 12.6 of [10]).

Here is a signature scheme due to Elgamal[46]. The message sender A chooses a very large prime p, some integer $1 < g < p$ and some other integer u with $1 < u < p$ (as usual, some randomisation scheme should be used). A then releases the values of p, g and $y = g^u$ (modulo p) but keeps the value of u secret. Whenever he sends a message m (some positive integer), he chooses another integer k with $1 \leq k \leq p-2$ at random and computes r and s with $1 \leq r \leq p-1$ and $0 \leq s \leq p-2$ by the rules[47]

$$r \equiv g^k \pmod{p}, \tag{*}$$

$$m \equiv ur + ks \pmod{p-1}. \tag{**}$$

Lemma 21.4. *If conditions (*) and (**) are satisfied, then*

$$g^m \equiv y^r r^s \pmod{p}.$$

If A sends the message m followed by the signature (r, s), the recipient need only verify the relation $g^m \equiv y^r r^s \pmod{p}$ to check that the message is authentic[48].

Since k is random, it is *believed* that the only way to forge signatures is to find u from g^u (or k from g^k) and it is *believed* that this problem, which is known as the discrete logarithm problem, is very hard.

Needless to say, even if it is impossible to tamper with a message–signature pair it is always possible to copy one. Every message should thus contain a unique identifier such as a time stamp.

---

[46]This is Dr Elgamal's own choice of spelling according to Wikipedia.

[47]There is a small point which I have glossed over here and elsewhere. Unless k and and $p-1$ are coprime the equation (**) may not be soluble. However the quickest way to solve (**), if it is soluble, is Euclid's algorithm which will also reveal if (**) is insoluble. If (**) is insoluble, we simply choose another k at random and try again.

[48]Sometimes, m is replaced by some hash function $H(m)$ of m so $(**)$ becomes $H(m) \equiv ur + ks \pmod{p-1}$. In this case the recipient checks that $g^{H(m)} \equiv y^r r^s \pmod{p}$.

The evidence that the discrete logarithm problem is very hard is of the same kind of nature and strength as the evidence that the factorisation problem is very hard. We conclude our discussion with a description of the Diffie–Hellman key exchange system which is also based on the discrete logarithm problem.

The modern coding schemes which we have discussed have the disadvantage that they require lots of computation. This is not a disadvantage when we deal slowly with a few important messages. For the Web, where we must deal speedily with a lot of less than world shattering messages sent by impatient individuals, this is a grave disadvantage. Classical coding schemes are fast but become insecure with reuse. *Key exchange schemes* use modern codes to communicate a new secret key for each message. Once the secret key has been sent slowly, a fast classical method based on the secret key is used to encode and decode the message. Since a different secret key is used each time, the classical code is secure.

How is this done? Suppose A and B are at opposite ends of a tapped telephone line. A sends B a (randomly chosen) large prime p and a randomly chosen g with $1 < g < p - 1$. Since the telephone line is insecure, A and B must assume that p and g are public knowledge. A now chooses randomly a secret number α and tells B the value of g^α (modulo p). B chooses randomly a secret number β and tells A the value of g^β (modulo p). Since

$$g^{\alpha\beta} \equiv (g^\alpha)^\beta \equiv (g^\beta)^\alpha,$$

both A and B can compute $k = g^{\alpha\beta}$ modulo p and k becomes the shared secret key.

The eavesdropper is left with the problem of finding $k \equiv g^{\alpha\beta}$ from knowledge of g, g^α and g^β (modulo p). It is conjectured that this is essentially as hard as finding α and β from the values of g, g^α and g^β (modulo p) and this is the discrete logarithm problem.

22 Quantum cryptography

In the days when messages were sent in the form of letters, suspicious people might examine the creases where the paper was folded for evidence that the letter had been read by others. Our final cryptographic system has the advantage that it too will reveal attempts to read it. It also has the advantage that, instead of relying on the unproven belief that a certain mathematical task is hard, it depends on the fact that a certain physical task is impossible[49].

[49] If you believe our present theories of the universe.

We shall deal with a highly idealised system. The business of dealing with realistic systems is a topic of active research within the faculty. The system we sketch is called the BB84 system (since it was invented by Bennett and Brassard in 1984) but there is another system invented by Ekert.

Quantum mechanics tells us that a polarised photon has a state

$$\phi = \alpha|\updownarrow\rangle + \beta|\leftrightarrow\rangle$$

where α, $\beta \in \mathbb{R}$, $\alpha^2 + \beta^2 = 1$, $\updownarrow\rangle$ is the vertically polarised state and $\leftrightarrow\rangle$ is the horizontally polarised state. Such a photon will pass through a vertical polarising filter with probability α^2 and its state will then be $\updownarrow\rangle$. It will pass through a horizontal polarising filter with probability β^2 and its state will then be $\leftrightarrow\rangle$. We have an orthonormal basis consisting of $\updownarrow\rangle$ and $\leftrightarrow\rangle$ by $+$.

We now consider a second basis given by

$$\nearrow\rangle = \frac{1}{\sqrt{2}}|\updownarrow\rangle + \frac{1}{\sqrt{2}}|\leftrightarrow\rangle \text{ and } \searrow\rangle = \frac{1}{\sqrt{2}}|\updownarrow\rangle - \frac{1}{\sqrt{2}}|\leftrightarrow\rangle$$

in which the states correspond to polarisation at angles $\pi/4$ and $-\pi/4$ to the horizontal. Observe that a photon in either state will have a probability 1/2 of passing through either a vertical or a horizontal filter and will then be in the appropriate state.

Suppose Eve[50] intercepts a photon passing between Alice and Bob. If Eve knows that it is either horizontally or vertically polarised, then she can use a vertical filter. If the photon passes through, she knows that it was vertically polarised when Alice sent it and can pass on a vertically polarised photon to Bob. If the photon does not pass through through, she knows that the photon was horizontally polarised and can pass on a horizontally polarised photon to Bob. However, if Alice's photon was actually diagonally polarised (at angle $\pm\pi/4$), this procedure will result in Eve sending Bob a photon which is horizontally or vertically polarised.

It is possible that the finder of a fast factorising method would get a Field's medal. It is certain that anyone who can do better than Eve would get the Nobel prize for physics since they would have overturned the basis of Quantum Mechanics.

Let us see how this can (in principle) be used to produce a key exchange scheme (so that Alice and Bob can agree on a random number to act as the basis for a classical code).

STEP 1 Alice produces a secret random sequence $a_1a_2\dots$ of bits (zeros and ones) and Bob produces another secret random sequence $b_1b_2\dots$ of bits.

STEP 2 Alice produces another secret random sequence $c_1c_2\dots$. She transmits it to Bob as follows.

[50] This is a traditional pun.

If $a_j = 0$ and $c_j = 0$, she uses a vertically polarised photon.
If $a_j = 0$ and $c_j = 1$, she uses a horizontally polarised photon.
If $a_j = 1$ and $c_j = 0$, she uses a 'left diagonally' polarised photon.
If $a_j = 1$ and $c_j = 1$, she uses a 'right diagonally' polarised photon.

STEP 3 If $b_j = 0$, Bob uses a vertical polariser to examine the jth photon. If he records a vertical polarisation, he sets $d_j = 0$, if a horizontal he sets $d_j = 1$. If $b_j = 1$, Bob uses a $\pi/4$ diagonal polariser to examine the jth photon. If he records a left diagonal polarisation, he sets $d_j = 0$, if a right he sets $d_j = 1$.
STEP 4 Bob and Alice use another communication channel to tell each other the values of the a_j and b_j. Of course, they should try to keep these communication secret, but we shall assume that worst has happened and these values become known to Eve.
STEP 5 If the sequences are long, we can be pretty sure, by the law of large numbers, that $a_j = b_j$ in about half the cases. (If not, Bob and Alice can agree to start again.) In particular, we can ensure that, with probability of at least $1 - \epsilon/4$ (where ϵ is chosen in advance), the number of agreements is sufficiently large for the purposes set out below. Alice and Bob only look at the 'good cases' when $a_j = b_j$. In such cases, if Eve does not examine the associated photon, then $d_j = c_j$. If Eve does examine the associated photon, then with probability $1/4$, $d_j \neq c_j$.

To see this, we examine the case when $c_j = 0$ and Eve uses a diagonal polariser. (The other cases may be treated in exactly the same way.) With probability $1/2$, $a_j = 1$ so the photon is diagonally polarised, Eve records the correct polarisation and sends Bob a correctly polarised photon. Thus $d_j = c_j$. With probability $1/2$, $a_j = 0$ so the photon is vertically or horizontally polarised. Since Eve records a diagonal polarisation she will send a diagonally polarised photon to Bob and, since Bob's polariser is vertical, he will record a vertical polarisation with probability $1/2$.
STEP 6 Alice uses another communication channel to tell Bob the value of a randomly chosen sample of good cases. Standard statistical techniques tell Alice and Bob that, if the number of discrepancies is below a certain level, the probability that Eve is intercepting more than a previously chosen proportion p of photons is less than $\epsilon/4$. If the number of discrepancies is greater than the chosen level, Alice and Bob will abandon the attempt to communicate.
STEP 7 If Eve is intercepting less than a proportion p of photons and $q > p$ (with q chosen in advance) the probability that she will have intercepted more than a proportion q of the remaining 'good' photons is less than $\epsilon/4$. Although we shall not do this, the reader who has ploughed through these

notes will readily accept that Bob and Alice can use the message conveyed through the remaining good photons to construct a common secret such that Eve has probability less than $\epsilon/4$ of guessing it.

Thus, unless they decide that their messages are being partially read, Alice and Bob can agree a shared secret with probability less than ϵ that an eavesdropper can guess it.

There are various gaps in the exposition above. First we have assumed that Eve must hold her polariser at a small fixed number of angles. A little thought shows that allowing her a free choice of angle will make little difference. Secondly, since physical systems always have imperfections, some 'good' photons will produce errors even in the absence of Eve. This means that p in STEP 5 must be chosen above the 'natural noise level' and the sequences must be longer but, again, this ought to make little difference. There is a further engineering problem that it is very difficult just to send single photons every time. If there are too many groups of photons, then Eve only need capture one and let the rest go, so we can not detect eavesdropping. If there are only a few, then the values of p and q can be adjusted to take account of this. There are several networks in existence which employ quantum cryptography.

Quantum cryptography has definite advantages when matched individually against RSA, secret sharing (using a large number of independent channels) or one-time pads. It is less easy to find applications where it is better than the best choice of one of these three 'classical' methods[51].

Of course, quantum cryptography will appeal to those who need to persuade others that they are using the latest and most expensive technology to guard their secrets. However as I said before *coding schemes are at best, cryptographic elements of larger possible cryptographic systems.* If smiling white coated technicians install big gleaming machines with 'Unbreakable Quantum Code Company' painted in large letters above the keyboard in the homes of Alice and Bob, it does not automatically follow that their communications are safe. Money will buy the appearance of security. Only thought will buy the appropriate security for a given purpose at an appropriate cost. And even then we can not be sure.

> As we know,
> There are known knowns.
> There are things we know we know.
> We also know

[51] One problem is indicated by the first British military action in World War I which was to cut the undersea telegraph cables linking Germany to the outside world. Complex systems are easier to disrupt than simple ones.

There are known unknowns.
That is to say
We know there are some things
We do not know.
But there are also unknown unknowns,
The ones we don't know
We don't know[52].

23 Further reading

For many students this will be one of the last university mathematics course they will take. Although the twin subjects of error-correcting codes and cryptography occupy a small place in the grand panorama of modern mathematics, it seems to me that they form a very suitable topic for such a final course.

Outsiders often think of mathematicians as guardians of abstruse but settled knowledge. Even those who understand that there are still problems unsettled, ask what mathematicians will do when they run out of problems. At a more subtle level, Kline's magnificent *Mathematical Thought from Ancient to Modern Times* [5] is pervaded by the melancholy thought that, though the problems will not run out, they may become more and more baroque and inbred. 'You are not the mathematicians your parents were' whispers Kline 'and your problems are not the problems your parents' were.'

However, when we look at this course, we see that the idea of error-correcting codes did not exist before 1940. The best designs of such codes depend on the kind of 'abstract algebra' that historians like Kline and Bell consider a dead end, and lie behind the superior performance of CD players and similar artifacts.

In order to go further into the study of codes, whether secret or error correcting, we need to go into the question of how the information content of a message is to be measured. 'Information theory' has its roots in the code breaking of World War II (though technological needs would doubtless have led to the same ideas shortly thereafter anyway). Its development required a level of sophistication in treating probability which was simply not available in the 19th century. (Even the Markov chain is essentially 20th century[53].)

The question of what makes a calculation difficult could not even have

[52] Rumsfeld

[53] We are now in the 21st century, but I suspect that we are still part of the mathematical 'long 20th century' which started in the 1880s with the work of Cantor and like minded contemporaries.

been thought about until Gödel's theorem (itself a product of the great 'foundations crisis' at the beginning of the 20th century). Developments by Turing and Church of Gödel's theorem gave us a theory of computational complexity which is still under development today. The question of whether there exist 'provably hard' public codes is intertwined with still unanswered questions in complexity theory. There are links with the profound (and very 20th century) question of what constitutes a random number.

Finally, the invention of the electronic computer has produced a cultural change in the attitude of mathematicians towards algorithms. Before 1950, the construction of algorithms was a minor interest of a few mathematicians. (Gauss and Jacobi were considered unusual in the amount of thought they gave to actual computation.) Today, we would consider a mathematician as much as a maker of algorithms as a prover of theorems. The notion of the *probabilistic algorithm* which hovered over much of our discussion of secret codes is a typical invention of the last decades of the 20th century.

Although both the subjects of error correcting and secret codes are now 'mature' in the sense that they provide usable and well tested tools for practical application, they still contain deep unanswered questions. For example

How close to the Shannon bound can a 'computationally easy' error correcting code get?

Do provably hard public codes exist?

Even if these questions are too hard, there must surely exist error correcting and public codes based on new ideas[54]. Such ideas would be most welcome and, although they are most likely to come from the professionals, they might come from outside the usual charmed circles.

Those who wish to learn about error correction from the horse's mouth will consult Hamming's own book on the matter [2]. For the present course, the best book I know for further reading is Welsh [10]. After this, the book of Goldie and Pinch [8] provides a deeper idea of the meaning of information and its connection with the topic. The book by Koblitz [6] develops the number theoretic background. The economic and practical importance of transmitting, storing and processing data far outweighs the importance of hiding it. However, hiding data is more romantic. For budding cryptologists and cryptographers (as well as those who want a good read), Kahn's *The Codebreakers* [3] has the same role as is taken by Bell's *Men of Mathematics* for budding mathematicians.

I conclude with a quotation from Galbraith (referring to his time as ambassador to India) taken from Koblitz's entertaining text [6].

> I had asked that a cable from Washington to New Delhi . . . be

[54] Just as quantum cryptography was.

reported to me through the Toronto consulate. It arrived in code; no facilities existed for decoding. They brought it to me at the airport — a mass of numbers. I asked if they assumed I could read it. They said no. I asked how they managed. They said that when something arrived in code, they phoned Washington and had the original read to them.

References

[1] U. Eco *The Search for the Perfect Language* (English translation), Blackwell, Oxford 1995.

[2] R. W. Hamming *Coding and Information Theory* (2nd edition) Prentice Hall, 1986.

[3] D. Kahn *The Codebreakers: The Story of Secret Writing* MacMillan, New York, 1967. (A lightly revised edition has recently appeared.)

[4] D. Kahn *Seizing the Enigma* Houghton Mifflin, Boston, 1991.

[5] M. Kline *Mathematical Thought from Ancient to Modern Times* OUP, 1972.

[6] N. Koblitz *A Course in Number Theory and Cryptography* Springer, 1987.

[7] D. E. Knuth *The Art of Computing Programming* Addison-Wesley. The third edition of Volumes I to III is appearing during this year and the next (1998–9).

[8] G. M. Goldie and R. G. E. Pinch *Communication Theory* CUP, 1991.

[9] T. M. Thompson *From Error-correcting Codes through Sphere Packings to Simple Groups* Carus Mathematical Monographs **21**, MAA, Washington DC, 1983.

[10] D. Welsh *Codes and Cryptography* OUP, 1988.

There is a widespread superstition, believed both by supervisors and supervisees, that exactly twelve questions are required to provide full understanding of six hours of mathematics and that the same twelve questions should be appropriate for students of all abilities and all levels of diligence. I have tried to keep this in mind, but have provided some extra questions in the various exercise sheets for those who scorn such old wives' tales.

24 Exercise Sheet 1

Q 24.1. (Exercises 1.1 and 1.2.) (i) Consider Morse code.

$$\begin{array}{rrr} A \mapsto \bullet - * & B \mapsto - \bullet \bullet \bullet * & C \mapsto - \bullet - \bullet * \\ D \mapsto - \bullet \bullet * & E \mapsto \bullet * & F \mapsto \bullet \bullet - \bullet * \\ O \mapsto - - - * & S \mapsto \bullet \bullet \bullet * & 7 \mapsto - - \bullet \bullet \bullet * \end{array}$$

Decode $- \bullet - \bullet * - - - * - \bullet \bullet * \bullet *$.

(ii) Consider ASCII code.

$$\begin{array}{rrr} A \mapsto 1000001 & B \mapsto 1000010 & C \mapsto 1000011 \\ a \mapsto 1100001 & b \mapsto 1100010 & c \mapsto 1100011 \\ + \mapsto 0101011 & ! \mapsto 0100001 & 7 \mapsto 0110111 \end{array}$$

Encode $b7!$. Decode 110001111000011100010.

Q 24.2. (Exercises 1.3, 1.4 and 1.7.) Consider two alphabets $\mathcal{A}$ and $\mathcal{B}$ and a coding function $c : \mathcal{A} \to \mathcal{B}^*$

(i) Explain, without using the notion of prefix-free codes, why, if c is injective and fixed length, c is decodable. Explain why, if c is injective and fixed length, c is prefix-free.

(ii) Let $\mathcal{A} = \mathcal{B} = \{0, 1\}$. If $c(0) = 0$, $c(1) = 00$ show that c is injective but c^* is not.

(iii) Let $\mathcal{A} = \{1, 2, 3, 4, 5, 6\}$ and $\mathcal{B} = \{0, 1\}$. Show that there is a variable length coding c such that c is injective and all code words have length 2 or less. Show that there is no decodable coding c such that all code words have length 2 or less

Q 24.3. The product of two codes $c_j : \mathcal{A}_j \to \mathcal{B}_j^*$ is the code

$$g : \mathcal{A}_1 \times \mathcal{A}_2 \to (\mathcal{B}_1 \cup \mathcal{B}_2)^*$$

given by $g(a_1, a_2) = c_1(a_1)c_2(a_2)$.

Show that the product of two prefix-free codes is prefix free, but the product of a decodable code and a prefix-free code need not even be decodable.

Q 24.4. (Exercises 2.5 and 2.7)

(i) Apply Huffman's algorithm to the nine messages M_j where M_j has probability $j/45$ for $1 \leq j \leq 9$.

(ii) Consider 4 messages with the following properties. M_1 has probability .23, M_2 has probability .24, M_3 has probability .26 and M_4 has probability .27. Show that any assignment of the code words 00, 01, 10 and 11 produces a best code in the sense of this course.

Q 24.5. (Exercises 2.6 and 4.6.) (i) Consider 64 messages M_j. M_1 has probability $1/2$, M_2 has probability $1/4$ and M_j has probability $1/248$ for $3 \leq j \leq 64$. Explain why, if we use code words of equal length, then the length of a code word must be at least 6. By using the ideas of Huffman's algorithm (you should not need to go through all the steps) obtain a set of code words such that the *expected* length of a code word sent is no more than 3.

(ii) Let a, $b > 0$. Show that

$$\log_a b = \frac{\log b}{\log a}.$$

Q 24.6. (Exercise 4.10) (i) Let $\mathcal{A} = \{1, 2, 3, 4\}$. Suppose that the probability that letter k is chosen is $k/10$. Use your calculator to find $\lceil -\log_2 p_k \rceil$ and write down a Shannon–Fano code c.

(ii) We found a Huffman code c_h for the system in Example 2.4. Show that the entropy is approximately 1.85, that $\mathbb{E}|c(A)| = 2.4$ and that $\mathbb{E}|c_h(A)| = 1.9$. Check that these results are consistent with the appropriate theorems of the course.

Q 24.7. (Exercise 5.1) Suppose that we have a sequence X_j of random variables taking the values 0 and 1. Suppose that $X_1 = 1$ with probability $1/2$ and $X_{j+1} = X_j$ with probability .99 independent of what has gone before.

(i) Suppose we wish to send 10 successive bits $X_j X_{j+1} \ldots X_{j+9}$. Show that if we associate the sequence of ten zeros with 0, the sequence of ten ones with 10 and any other sequence $a_0 a_1 \ldots a_9$ with $11 a_0 a_1 \ldots a_9$, we have a decodable code which on average requires about $5/2$ bits to transmit the sequence.

(ii) Suppose we wish to send the bits $X_j X_{j+10^6} X_{j+2\times 10^6} \ldots X_{j+9\times 10^6}$. Explain why any decodable code will require on average at least 10 bits to transmit the sequence. (You need not do detailed computations.)

Q 24.8. In Bridge, a 52 card pack is dealt to provide 4 hands of 13 cards each.

(i) Purely as a matter of interest, we consider the following question. If the contents of a hand are conveyed by one player to their partner by a series of nods and shakes of the head how many movements of the head are required? Show that at least 40 movements are required. Give a simple code requiring 52 movements.

[You may assume for simplicity that the player to whom the information is being communicated does not look at her own cards. (In fact this does not make a difference since the two players do not acquire any shared information by looking at their own cards.)]

(ii) If instead the player uses the initial letters of words (say using the 16 most common letters), how many words will you need to utter[55]?

Q 24.9. (i) In a *comma code*, like Morse code, one symbol from an alphabet of m letters is reserved to end each code word. Show that this code is prefix-free and give a direct argument to show that it must satisfy Kraft's inequality.

(ii) Give an example of a code satisfying Kraft's inequality which is not decodable.

Q 24.10. Show that if an optimal binary code has word lengths s_1, s_2, $\dots s_m$ then

$$m \log_2 m \leq s_1 + s_2 + \cdots + s_m \leq (m^2 + m - 2)/2.$$

Q 24.11. (i) It is known that exactly one member of the starship Emphasise has contracted the Macguffin virus. A test is available that will detect the virus at any dilution. However, the power required is such that the ship's force shields must be switched off[56] for a minute during each test. Blood samples are taken from all crew members. The ship's computer has worked out that the probability of crew member number i harbouring the virus is p_i. (Thus the probability that the captain, who is, of course, number 1, has the disease is p_1.) Explain how, by testing pooled samples, the expected number of tests can be minimised. Write down the exact form of the test when there are 2^n crew members and $p_i = 2^{-n}$.

(ii) Questions like (i) are rather artificial, since they require that exactly one person carries the virus. Suppose that the probability that any member of a population of 2^n has a certain disease is p (and that the probability

[55]'Marked cards, M. l'Anglais?' I said, with a chilling sneer. 'They are used, I am told, to trap players–not unbirched schoolboys.'

'Yet I say that they are marked!' he replied hotly, in his queer foreign jargon. 'In my last hand I had nothing. You doubled the stakes. Bah, sir, you knew! You have swindled me!'

'Monsieur is easy to swindle – when he plays with a mirror behind him,' I answered tartly. *Under the Red Robe* S. J. Weyman

[56]'Captain, ye canna be serious.'

is independent of the health of the others) and there exists an error free test which can be carried out on pooled blood samples which indicates the presence of the disease in at least one of the samples or its absence from all.

Explain why there cannot be a testing scheme which can be guaranteed to require less than 2^n tests to diagnose all members of the population. How does the scheme suggested in the last sentence of (i) need to be modified to take account of the fact that more than one person may be ill (or, indeed, no one may be ill)? Show that the expected number of tests required by the modified scheme is no greater than $pn2^{n+1} + 1$. Explain why the cost of testing a large population of size x is no more than about $2pcx \log_2 x$ with c the cost of a test.

(iii) In practice, pooling schemes will be less complicated. Usually a group of x people are tested jointly and, if the joint test shows the disease, each is tested individually. Explain why this is not sensible if p is large but is sensible (with a reasonable choice of x) if p is small. If p is small, explain why there is an optimum value for x Write down (but do not attempt to solve) an equation which indicates (in a 'mathematical methods' sense) that optimum value in terms of p, the probability that an individual has the disease.

Schemes like these are only worthwhile if the disease is rare and the test is both expensive and will work on pooled samples. However, these circumstances do occur together from time to time and the idea then produces public health benefits much more cheaply than would otherwise be possible.

Q 24.12. (i) Give the appropriate generalisation of Huffman's algorithm to an alphabet with a symbols when you have m messages and $m \equiv 1 \mod a - 1$.

(ii) Prove that your algorithm gives an optimal solution.

(iii) Extend the algorithm to cover general m by introducing messages of probability zero.

Q 24.13. (i) A set of m apparently identical coins consists of $m - 1$ coins and one heavier coin. You are given a balance in which you can weigh equal numbers of the coins and determine which side (if either) contains the heavier coin. You wish to find the heavy coin in the fewest average number of weighings.

If $3^r + 1 \leq m \leq 3^{r+1}$ show that you can label each coin with a ternary number $a_1 a_2 \dots a_{r+1}$ with $a_j \in \{0, 1, 2\}$ in such a way that the number of coins having 1 in the jth place equals the number of coins with 2 in the jth place for each j (think Huffman ternary trees).

By considering the Huffman algorithm problem for prefix-free codes on an alphabet with three letters, solve the problem stated in the first part and show that you do indeed have a solution. Show that your solution also minimises the maximum number of weighings that you might have to do.

(ii) Suppose the problem is as before but $m = 12$ and the odd coin may be heavier or lighter. Show that you need at least 3 weighings.

[In fact you can always do it in 3 weighings, but the problem of showing this 'is said to have been planted during the war ... by enemy agents since Operational Research spent so many man-hours on its solution.'[57]]

Q 24.14. Extend the definition of entropy to a random variable X taking values in the non-negative integers. (You must allow for the possibility of infinite entropy.)

Compute the expected value $\mathbb{E}Y$ and entropy $H(Y)$ in the case when Y has the geometric distribution, that is to say $\Pr(Y = k) = p^k(1-p)$ $[0 < p < 1]$. Show that, amongst all random variables X taking values in the non-negative integers with the same expected value μ $[0 < \mu < \infty]$, the geometric distribution maximises the entropy.

Q 24.15. A source produces a set $\mathcal{A}$ of messages M_1, M_2, $\dots$, M_n with non-zero probabilities p_1, p_2, $\dots p_n$. Let S be the codeword length when the message is encoded by a decodable code $c : \mathcal{A} \to \mathcal{B}^*$ where $\mathcal{B}$ is an alphabet of k letters.

(i) Show that

$$\left(\sum_{i=1}^{n} \sqrt{p_i}\right)^2 \leq \mathbb{E}(k^S)$$

[Hint: Cauchy–Schwarz, $p_i^{1/2} = p_i^{1/2} k^{s_i/2} k^{-s_i/2}$.]

(ii) Show that

$$\min \mathbb{E}(k^S) \leq k \left(\sum_{i=1}^{n} \sqrt{p_i}\right)^2.$$

where the minimum is taken over all decodable codes.
[Hint: Look for a code with codeword lengths $s_i = \lceil -\log_k p_i^{1/2}/\lambda \rceil$ for an appropriate λ.]

[57] The quotation comes from Pedoe's *The Gentle Art of Mathematics* which also gives a very pretty solution. As might be expected, there are many accounts of this problem on the web.

25 Exercise Sheet 2

Q 25.1. (Exercise 7.3.) In an exam each candidate is asked to write down a Candidate Number of the form $3234A$, $3235B$, $3236C$,... (the eleven possible letters are repeated cyclically) and a desk number. (Thus candidate 0004 sitting at desk 425 writes down $0004D - -425$.) The first four numbers in the Candidate Identifier identify the candidate uniquely. Show that if the candidate makes one error in the Candidate Identifier then that error can be detected without using the Desk Number. Would this be true if there were 9 possible letters repeated cyclically? Would this be true if there were 12 possible letters repeated cyclically? Give reasons.

Show that if we combine the Candidate Number and the Desk Number the combined code is one error correcting.

Q 25.2. (Exercise 6.1) In the model of a communication channel, we take the probability p of error to be less than $1/2$. Why do we not consider the case $1 \geq p > 1/2$? What if $p = 1/2$?

Q 25.3. (Exercise 7.4.) If you look at the inner title page of almost any book published between 1974 and 2007, you will find its International Standard Book Number (ISBN). The ISBN uses single digits selected from 0, 1, ..., 8, 9 and X representing 10. Each ISBN consists of nine such digits a_1, a_2, ..., a_9 followed by a single check digit a_{10} chosen so that

$$10a_1 + 9a_2 + \cdots + 2a_9 + a_{10} \equiv 0 \mod 11. \qquad (*)$$

(In more sophisticated language, our code C consists of those elements $\mathbf{a} \in \mathbb{F}_{11}^{10}$ such that $\sum_{j=1}^{10}(11-j)a_j = 0$.)

(i) Find a couple of books and check that $(*)$ holds for their ISBNs.

(ii) Show that $(*)$ will not work if you make a mistake in writing down one digit of an ISBN.

(iii) Show that $(*)$ may fail to detect two errors.

(iv) Show that $(*)$ will not work if you interchange two distinct adjacent digits (a transposition error).

(v) Does (iv) remain true if we replace 'adjacent' by 'different'? Errors of type (ii) and (iv) are the most common in typing.

In communication between publishers and booksellers, both sides are anxious that errors should be detected but would prefer the other side to query errors rather than to guess what the error might have been.

(vi) Since the ISBN contained information such as the name of the publisher, only a small proportion of possible ISBNs could be used[58] and the

[58]The same problem occurs with telephone numbers. If we use the Continent, Country,

system described above started to 'run out of numbers'. A new system was introduced which was is compatible with the system used to label most consumer goods. After January 2007, the appropriate ISBN became a 13 digit number $x_1x_2\dots x_{13}$ with each digit selected from 0, 1, …, 8, 9 and the check digit x_{13} computed by using the formula

$$x_{13} \equiv -(x_1 + 3x_2 + x_3 + 3x_4 + \cdots + x_{11} + 3x_{12}) \mod 10.$$

Show that we can detect single errors. Give an example to show that we cannot detect all transpositions.

Q 25.4. (Exercise 7.5.) Suppose we use eight hole tape with the standard paper tape code and the probability that an error occurs at a particular place on the tape (i.e. a hole occurs where it should not or fails to occur where it should) is 10^{-4}. A program requires about 10 000 lines of tape (each line containing eight places) using the paper tape code. Using the Poisson approximation, direct calculation (possible with a hand calculator but really no advance on the Poisson method), or otherwise, show that the probability that the tape will be accepted as error free by the decoder is less than .04%.

Suppose now that we use the Hamming scheme (making no use of the last place in each line). Explain why the program requires about 17 500 lines of tape but that any particular line will be correctly decoded with probability about $1 - (21 \times 10^{-8})$ and the probability that the entire program will be correctly decoded is better than 99.6%.

Q 25.5. If $0 < \delta < 1/2$, find an $A(\delta) > 0$ such that, whenever $0 \leq r \leq n\delta$, we have

$$\sum_{j=0}^{r} \binom{n}{j} \leq A(\delta)\binom{n}{r}.$$

(We use weaker estimates in the course but this is the most illuminating. The particular value of $A(\delta)$ is unimportant so do not waste time trying to find a 'good' value.)

Q 25.6. Show that the n-fold repetition code is perfect if and only if n is odd.

Q 25.7. (i) What is the expected Hamming distance between two randomly chosen code words in $\mathbb{F}_2^n$. (As usual we suppose implicitly that the two choices are independent and all choices are equiprobable.)

Town, Subscriber system we will need longer numbers than if we just numbered each member of the human race.

(ii) Three code words are chosen at random from $\mathbb{F}_2^n$. If k_n is the expected value of the distance between the closest two, show that $n^{-1}k_n \to 1/2$ as $n \to \infty$.
[There are many ways to do (ii). One way is to consider Tchebychev's inequality.]

Q 25.8. (Exercises 11.2 and 11.3.) Consider the situation described in the first paragraph of Section 11.

(i) Show that for the situation described you should not bet if $up \leq 1$ and should take

$$w = \frac{up - 1}{u - 1}$$

if $up > 1$.

(ii) Let us write $q = 1 - p$. Show that, if $up > 1$ and we choose the optimum w,

$$\mathbb{E} \log Y_n = p \log p + q \log q + \log u - q \log(u - 1).$$

(iii) Show that, if you bet less than the optimal proportion, your fortune will still tend to increase but more slowly, but, if you bet more than some proportion w_1, your fortune will decrease. Write down the equation for w_1.

[Moral: If you use the Kelly criterion veer on the side under-betting.]

Q 25.9. Your employer announces that he is abandoning the old-fashioned paternalistic scheme under which he guarantees you a fixed sum Kx (where, of course, K, $x > 0$) when you retire. Instead, he will empower you by giving you a fixed sum x now, to invest as you wish. In order to help you and the rest of the staff, your employer arranges that you should obtain advice from a financial whizkid with a top degree from Cambridge. After a long lecture in which the whizkid manages to be simultaneously condescending, boring and incomprehensible, you come away with the following information.

When you retire, the world will be in exactly one of n states. By means of a piece of financial wizardry called ditching (or something like that) the whizkid can offer you a pension plan which for the cost of x_i will return $Kx_iq_i^{-1}$ if the world is in state i, but nothing otherwise. (Here $q_i > 0$ and $\sum_{i=1}^n q_i = 1$.) The probability that the world will be in state i is p_i. You must invest the entire fixed sum. (Formally, $\sum_{i=1}^n x_i = x$. You must also take $x_i \geq 0$.) On philosophical grounds you decide to maximise the expected value S of the logarithm of the sum received on retirement. Assuming that you will have to live off this sum for the rest of your life, explain, *in your opinion*, why this choice is reasonable or explain why it is unreasonable.

Find the appropriate choices of x_i. Do they depend on the q_i?

Suppose that K is fixed, but the whizkid can choose q_i. We may suppose that what is good for you is bad for him so he will seek to minimise S for your best choices. Show that he will choose $q_i = p_i$. Show that, with these choices,

$$S = \log Kx.$$

Q 25.10. Let C be the code consisting of the word 10111000100 and its cyclic shifts (that is 01011100010, 00101110001 and so on) together with the zero code word. Is C linear? Show that C has minimum distance 5.

Q 25.11. (i) The original Hamming code was a 7 bit code used in an 8 bit system (paper tape). Consider the code $c : \{0,1\}^4 \to \{0,1\}^8$ obtained by using the Hamming code for the first 7 bits and the final bit as a check digit so that

$$x_1 + x_2 + \cdots + x_8 \equiv 0 \mod 2.$$

Find the minimum distance for this code. How many errors can it detect? How many can it correct?

(ii) Given a code of length n which corrects e errors can you always construct a code of length $n+1$ which detects $2e+1$ errors?

Q 25.12. In general, we work under the assumption that all messages sent through our noisy channel are equally likely. In this question we drop this assumption. Suppose that each bit sent through a channel has probability 1/3 of being mistransmitted. There are 4 codewords 1100, 0110, 0001, 1111 sent with probabilities 1/4, 1/2, 1/12, 1/6. If you receive 1001 what will you decode it as, using each of the following rules?

(i) The ideal observer rule: find $\mathbf{b} \in C$ so as to maximise

$$\Pr(\mathbf{b} \text{ sent} \,|\, \mathbf{u} \text{ received}\}.$$

(ii) The maximum likelihood rule: find $\mathbf{b} \in C$ so as to maximise

$$\Pr(\mathbf{u} \text{ received} \,|\, \mathbf{b} \text{ sent}\}.$$

(iii) The minimum distance rule: find $\mathbf{b} \in C$ so as to minimise the Hamming distance $d(\mathbf{b}, \mathbf{u})$ from the received message $\mathbf{u}$.

Q 25.13. (i) Show that $-t \geq \log(1-t)$ for $0 \leq t < 1$.

(ii) Show that, if $\delta_N > 0$, $1 - N\delta_N > 0$ and $N^2\delta_N \to \infty$, then

$$\prod_{m=1}^{N-1}(1 - m\delta_N) \to 0.$$

(iii) Let $V(n, r)$ be the number of points in a Hamming ball of radius r in $\mathbb{F}_2^n$ and let $p(n, N, r)$ be the probability that N such balls chosen at random do not intersect. By observing that if m non-intersecting balls are already placed, then an $m+1$st ball which does not intersect them must certainly not have its centre in one of the balls already placed, show that, if $N_n^2 2^{-n} V(n, r_n) \to \infty$, then $p(n, N_n, r_n) \to 0$.

(iv) Show that, if $2\beta + H(\alpha) > 1$, then $p(n, 2^{\beta n}, \alpha n) \to 0$.

Thus simply throwing balls down at random will not give very good systems of balls with empty intersections.

26 Exercise Sheet 3

Q 26.1. A message passes through a binary symmetric channel with probability p of error for each bit and the resulting message is passed through a second binary symmetric channel which is identical except that there is probability q of error $[0 < p, q < 1/2]$. Show that the result behaves as if it had been passed through a binary symmetric channel with probability of error to be determined. Show that the probability of error is less than $1/2$. Can we improve the rate at which messages are transmitted (with low error) by coding, sending through the first channel, decoding with error correction and then recoding, sending through the second channel and decoding with error correction again or will this produce no improvement on treating the whole thing as a single channel and coding and decoding only once?

Q 26.2. Write down the weight enumerators of the trivial code (that is to say, $\mathbb{F}_2^n$), the zero code (that is to say, $\{\mathbf{0}\}$), the repetition code and the simple parity code.

Q 26.3. List the codewords of the Hamming (7,4) code and its dual. Write down the weight enumerators and verify that they satisfy the MacWilliams identity.

Q 26.4. (a) Show that if C is linear, then so are its extension C^+, truncation C^- and puncturing C', provided the symbol chosen to puncture by is 0. Give an example to show that C' may not be linear if we puncture by 1.

(b) Show that extension followed by truncation does not change a code. Is this true if we replace 'truncation' by 'puncturing'?

(c) Give an example where puncturing reduces the information rate and an example where puncturing increases the information rate.

(d) Show that the minimum distance of the parity extension C^+ is the least even integer n with $n \geq d(C)$.

(e) Show that the minimum distance of the truncation C^- is $d(C)$ or $d(C) - 1$ and that both cases can occur.

(f) Show that puncturing cannot decrease the minimum distance, but give examples to show that the minimum distance can stay the same or increase.

Q 26.5. If C_1 and C_2 are linear codes of appropriate type with generator matrices G_1 and G_2, write down a generator matrix for $C_1|C_2$.

Q 26.6. Show that the weight enumerator of $RM(d, 1)$ is

$$y^{2^d} + (2^{d+1} - 2)x^{2^{d-1}}y^{2^{d-1}} + x^{2^d}.$$

Q 26.7. (i) Show that every codeword in $RM(d, d-1)$ has even weight.

(ii) Show that $RM(m, m-r-1) \subseteq RM(m,r)^{\perp}$.

(iii) By considering dimension, or otherwise, show that $RM(m,r)$ has dual code $RM(m, m-r-1)$.

Q 26.8. (Exercises 8.6 and 8.7.) We show that, even if $2^n/V(n,e)$ is an integer, no perfect code may exist.

(i) Verify that

$$\frac{2^{90}}{V(90,2)} = 2^{78}.$$

(ii) Suppose that C is a perfect 2 error correcting code of length 90 and size 2^{78}. Explain why we may suppose, without loss of generality, that $\mathbf{0} \in C$.

(iii) Let C be as in (ii) with $\mathbf{0} \in C$. Consider the set

$$X = \{\mathbf{x} \in \mathbb{F}_2^{90} : x_1 = 1,\ x_2 = 1,\ d(\mathbf{0}, \mathbf{x}) = 3\}.$$

Show that, corresponding to each $\mathbf{x} \in X$, we can find a unique $\mathbf{c}(\mathbf{x}) \in C$ such that $d(\mathbf{c}(\mathbf{x}), \mathbf{x}) = 2$.

(iv) Continuing with the argument of (iii), show that

$$d(\mathbf{c}(\mathbf{x}), \mathbf{0}) = 5$$

and that $c_i(\mathbf{x}) = 1$ whenever $x_i = 1$. If $\mathbf{y} \in X$, find the number of solutions to the equation $\mathbf{c}(\mathbf{x}) = \mathbf{c}(\mathbf{y})$ with $\mathbf{x} \in X$ and, by considering the number of elements of X, obtain a contradiction.

(v) Conclude that there is no perfect $[90, 2^{78}]$ code.

(vi) Show that $V(3, 23)$ is a power of 2. (In this case a perfect code exists called the binary Golay code.)

Q 26.9. [The MacWilliams identity for binary codes] Let $C \subseteq \mathbb{F}_2^n$ be a linear code of dimension k.

(i) Show that

$$\sum_{\mathbf{x} \in C} (-1)^{\mathbf{x}.\mathbf{y}} = \begin{cases} 2^k & \text{if } \mathbf{y} \in C^{\perp} \\ 0 & \text{if } \mathbf{y} \notin C^{\perp}. \end{cases}$$

(ii) If $t \in \mathbb{R}$, show that

$$\sum_{\mathbf{y} \in \mathbb{F}_2^n} t^{w(\mathbf{y})} (-1)^{\mathbf{x}.\mathbf{y}} = (1-t)^{w(\mathbf{x})} (1+t)^{n - w(\mathbf{x})}.$$

(iii) By using parts (i) and (ii) to evaluate

$$\sum_{\mathbf{x} \in C} \left(\sum_{\mathbf{y} \in \mathbb{F}_2^n} (-1)^{\mathbf{x}.\mathbf{y}} \left(\frac{s}{t}\right)^{w(\mathbf{y})} \right)$$

in two different ways, obtain the MacWilliams identity

$$W_{C^\perp}(s,t) = 2^{-\dim C} W_C(t-s, t+s).$$

Q 26.10. An *erasure* is a digit which has been made unreadable in transmission. Why are they easier to deal with than errors? Find a necessary and sufficient condition on the parity check matrix for it to be always possible to correct t erasures. Find a necessary and sufficient condition on the parity check matrix for it never to be possible to correct t erasures (ie whatever message you choose and whatever t erasures are made the recipient cannot tell what you sent).

Q 26.11. Consider the collection K of polynomials

$$a_0 + a_1\omega$$

with $a_j \in \mathbb{F}_2$ manipulated subject to the usual rules of polynomial arithmetic and to the further condition

$$1 + \omega + \omega^2 = 0.$$

Show by finding a generator and writing out its powers that $K^* = K \setminus \{0\}$ is a cyclic group under multiplication and deduce that K is a finite field.
[Of course, this follows directly from general theory but direct calculation is not uninstructive.]

Q 26.12. (i) Identify the cyclic codes of length n corresponding to each of the polynomials 1, $X-1$ and $X^{n-1} + X^{n-2} + \dots + X + 1$.

(ii) Show that there are three cyclic codes of length 7 corresponding to irreducible polynomials of which two are versions of Hamming's original code. What are the other cyclic codes?

(iii) Identify the dual codes for each of the codes in (ii).

Q 26.13. (Example 15.14.) Prove the following results.

(i) If K is a field containing $\mathbb{F}_2$, then $(a+b)^2 = a^2 + b^2$ for all $a, b \in K$.

(ii) If $P \in \mathbb{F}_2[X]$ and K is a field containing $\mathbb{F}_2$, then $P(a)^2 = P(a^2)$ for all $a \in K$.

(iii) Let K be a field containing $\mathbb{F}_2$ in which $X^7 - 1$ factorises into linear factors. If β is a root of $X^3 + X + 1$ in K, then β is a primitive root of unity and β^2 is also a root of $X^3 + X + 1$.

(iv) We continue with the notation of (iii). The BCH code with $\{\beta, \beta^2\}$ as defining set is Hamming's original (7,4) code.

Q 26.14. Let C be a binary linear code of length n, rank k and distance d.

(i) Show that C contains a codeword $\mathbf{x}$ with exactly d non-zero digits.

(ii) Show that $n \geq d + k - 1$.

(iii) Prove that truncating C on the non-zero digits of $\mathbf{x}$ produces a code C' of length $n - d$, rank $k - 1$ and distance $d' \geq \lceil \frac{d}{2} \rceil$.
[Hint: To show $d' \geq \lceil \frac{d}{2} \rceil$, consider, for $\mathbf{y} \in C$, the coordinates where $x_j = y_j$ and the coordinates where $x_j \neq y_j$.]

(iv) Show that

$$n \geq d + \sum_{u=1}^{k-1} \lceil \tfrac{d}{2^u} \rceil.$$

Why does (iv) imply (ii)? Give an example where $n > d + k - 1$.

Q 26.15. Implement the secret sharing method of page 57 with $k = 2$, $n = 3$, $x_j = j + 1$ $p = 7$, $a_0 = S = 2$, $a_1 = 3$. Check directly that any two people can find S but no single individual can.

If we take $k = 3$, $n = 4$, $p = 6$, $x_j = j+1$ show that the first two members and the fourth member of the Faculty Board will be unable to determine S uniquely. Why does this not invalidate our method?

27 Exercise Sheet 4

Q 27.1. (Exercise 18.2.) Show that the decimal expansion of a rational number must be a recurrent expansion. Give a bound for the period in terms of the quotient. Conversely, by considering geometric series, or otherwise, show that a recurrent decimal represents a rational number.

Q 27.2. A binary non-linear feedback register of length 4 has defining relation

$$x_{n+1} = x_n x_{n-1} + x_{n-3}.$$

Show that the state space contains 4 cycles of lengths 1, 2, 4 and 9

Q 27.3. A binary LFR was used to generate the following stream

$$110001110001\ldots$$

Recover the feedback polynomial by the Berlekamp–Massey method. [The LFR has length 4 but you should work through the trials for length r for $1 \leq r \leq 4$.]

Q 27.4. (Exercise 16.5.) Consider the linear recurrence

$$x_n = a_0 x_{n-d} + a_1 x_{n-d+1} + \ldots + a_{d-1} x_{n-1} \qquad \bigstar$$

with $a_j \in \mathbb{F}_2$ and $a_0 \neq 0$.

(i) Suppose K is a field containing $\mathbb{F}_2$ such that the auxiliary polynomial C has a root α in K. Show that $x_n = \alpha^n$ is a solution of $\bigstar$ in K.

(ii) Suppose K is a field containing $\mathbb{F}_2$ such that the auxiliary polynomial C has d distinct roots α_1, α_2, …, α_d in K. Show that the general solution of $\bigstar$ in K is

$$x_n = \sum_{j=1}^{d} b_j \alpha_j^n$$

for some $b_j \in K$. If $x_0, x_1, \ldots, x_{d-1} \in \mathbb{F}_2$, show that $x_n \in \mathbb{F}_2$ for all n.

(iii) Work out the first few lines of Pascal's triangle modulo 2. Show that the functions $f_j : \mathbb{Z} \to \mathbb{F}_2$

$$f_j(n) = \binom{n}{j}$$

are linearly independent in the sense that

$$\sum_{j=0}^{m} b_j f_j(n) = 0$$

for all n implies $b_j = 0$ for $1 \leq j \leq m$.

(iv) Suppose K is a field containing $\mathbb{F}_2$ such that the auxiliary polynomial C factorises completely into linear factors. If the root α_u has multiplicity $m(u)$ $[1 \leq u \leq q]$, show that the general solution of $\bigstar$ in K is

$$x_n = \sum_{u=1}^{q} \sum_{v=0}^{m(u)-1} b_{u,v} \binom{n}{v} \alpha_u^n$$

for some $b_{u,v} \in K$. If $x_0, x_1, \ldots, x_{d-1} \in \mathbb{F}_2$, show that $x_n \in \mathbb{F}_2$ for all n.

Q 27.5. Consider the recurrence relation

$$u_{n+p} + \sum_{j=0}^{n-1} c_j u_{j+p} = 0$$

over a field (if you wish, you may take the field to be $\mathbb{R}$ but the algebra is the same for all fields.) We suppose $c_0 \neq 0$. Write down an $n \times n$ matrix M such that

$$\begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{pmatrix} = M \begin{pmatrix} u_0 \\ u_1 \\ \vdots \\ u_{n-1} \end{pmatrix}.$$

Find the characteristic and minimal polynomials for M. Would your answers be the same if $c_0 = 0$?

Q 27.6. (Exercise 18.9.) One of the most confidential German codes (called FISH by the British) involved a complex mechanism which the British found could be simulated by two loops of paper tape of length 1501 and 1497. If $k_n = x_n + y_n$ where x_n is a stream of period 1501 and y_n is a stream of period 1497, what is the longest possible period of k_n? How many consecutive values of k_n would you need to to find the underlying linear feedback register using the Berlekamp–Massey method if you did not have the information given in the question? If you had all the information given in the question how many values of k_n would you need? (Hint, look at $x_{n+1497} - x_n$.)

You have shown that, given k_n for sufficiently many consecutive n we can find k_n for all n. Can you find x_n for all n?

Q 27.7. We work in $\mathbb{F}_2$. I have a secret sequence k_1, k_2, ... and a message p_1, p_2, ..., p_N. I transmit $p_1 + k_1$, $p_2 + k_2$, ... $p_N + k_N$ and then, by error, transmit $p_1 + k_2$, $p_2 + k_3$, ... $p_N + k_{N+1}$. Assuming that you know this and that my message makes sense, how would you go about finding my message? Can you now decipher other messages sent using the same part of my secret sequence?

Q 27.8. Give an example of a homomorphism attack on an RSA code. Show in reasonable detail that the Elgamal signature scheme defeats it.

Q 27.9. I announce that I shall be using the Rabin–Williams scheme with modulus N. My agent in X'Dofdro sends me a message m (with $1 \leq m \leq N-1$) encoded in the requisite form. Unfortunately, my cat eats the piece of paper on which the prime factors of N are recorded, so I am unable to decipher it. I therefore find a new pair of primes and announce that I shall be using the Rabin–Williams scheme with modulus $N' > N$. My agent now recodes the message and sends it to me again.

The dreaded SNDO of X'Dofdro intercept both code messages. Show that they can find m. Can they decipher any other messages sent to me using only one of the coding schemes?

Q 27.10. Extend the Diffie–Hellman key exchange system to cover three participants in a way that is likely to be as secure as the two party scheme.

Extend the system to n parties in such a way that they can compute their common secret key by at most $n^2 - n$ communications of 'Diffie–Hellman type numbers'. (The numbers p and g of our original Diffie-Hellman system are known by everybody in advance.) Show that this can be done using at most $2n-2$ communications by including several 'Diffie–Hellman type numbers' in one message.

Q 27.11. St Abacus, who established written Abacan, was led, on theological grounds, to use an alphabet containing only three letters A, B and C and to avoid the use of spaces. (Thus an Abacan book consists of single word.) In modern Abacan, the letter A has frequency .5 and the letters B and C both have frequency .25. In order to disguise this, the Abacan Navy uses codes in which the $3r+i$th number is $x_{3r+i} + y_i$ modulo 3 $[0 \leq i \leq 2]$ where $x_j = 0$ if the jth letter of the message is A, $x_j = 1$ if the jth letter of the message is B, $x_j = 2$ if the jth letter of the message is C and y_0, y_1 and y_2 are the numbers 0, 1, 2 in some order.

Radio interception has picked up the following message.

120022010211121001001021002021

Although nobody in Naval Intelligence reads Abacan, it is believed that the last letter of the message will be B if the Abacan fleet is at sea. The Admiralty are desperate to know the last letter and send a representative to your rooms in Baker Street to ask your advice. Give it.

Q 27.12. Consider the bit exchange scheme proposed at the end of Section 19. Suppose that we replace STEP 5 by:- Alice sends Bob r_1 and r_2 and Bob checks that

$$r_1^2 \equiv r_2^2 \equiv m \mod n.$$

Suppose further that Alice cheats by choosing 3 primes p_1, p_2, p_3, and sending Bob $p = p_1$ and $q = p_2 p_3$. Explain how Alice can shift the odds of heads to 3/4. (She has other ways of cheating, but you are only asked to consider this one.)

Q 27.13. (i) Consider the Fermat code given by the following procedure. 'Choose N a large prime. Choose e and d so that $a^{de} \equiv a \mod N$, encrypt using the publicly known N and e, decrypt using the secret d.' Why is this not a good code?

(ii) In textbook examples of the RSA code we frequently see $e = 65537$. How many multiplications are needed to compute a^e modulo N?

(iii) Why is it unwise to choose primes p and q with $p - q$ small when forming $N = pq$ for the RSA method? Factorise 1763.

Q 27.14. The University of Camford is proud of the excellence of its privacy system CAMSEC. To advertise this fact to the world, the Vice-Chancellor decrees that the university telephone directory should bear on its cover a number N (a product of two very large secret primes) and each name in the University Directory should be followed by their personal encryption number e_i. The Vice-Chancellor knows all the secret decryption numbers d_i but gives these out on a need to know basis only. (Of course each member of staff must know their personal decryption number but they are instructed to keep it secret.) Messages a from the Vice-Chancellor to members of staff are encrypted in the standard manner as a^{e_i} modulo N and decrypted as b^{d_i} modulo N.

(i) The Vice-Chancellor sends a message to all members of the University. An outsider intercepts the encrypted message to individuals i and j where e_i and e_j are coprime. How can the outsider read the message? Can she read other messages sent from the Vice-Chancellor to the ith member of staff only?

(ii) By means of a phone tapping device, the Professor of Applied Numismatics (number u in the University Directory) has intercepted messages from the Vice-Chancellor to her hated rival, the Professor of Pure Numismatics (number v in the University Directory). Explain why she can decode them.

What moral should be drawn?

Q 27.15. The Poldovian Embassy uses a one-time pad to communicate with the notorious international spy Ivanovich Smith. The messages are coded in the obvious way. (If the pad has C the 3rd letter of the alphabet and the message has I the 9th then the encrypted message has L the $3 + 9$th. Work modulo 26.) Unknown to them, the person whom they employ to carry the messages is actually the MI5 agent 'Union' Jack Caruthers in disguise. MI5 are on the verge of arresting Ivanovich when 'Union' Jack is given the message

$$LRPFOJQLCUD.$$

Caruthers knows that the actual message is

$$FLYXATXONCE$$

and suggests that 'the boffins change things a little' so that Ivanovich deciphers the message as

$$REMAINXHERE.$$

The only boffin available is you. Advise MI5.

Q 27.16. Suppose that X and Y are independent random variables taking values in $\mathbb{Z}_n$. Show that

$$H(X + Y) \geq \max\{H(X), H(Y)\}.$$

Why is this remark of interest in the context of one-time pads?

Does this result remain true if X and Y need not be independent? Give a proof or counterexample.

Q 27.17. I use the Elgamal signature scheme described on page 77. Instead of choosing k at random, I increase the value used by 2 each time I use it. Show that it will often be possible to find my privacy key u from two successive messages.

Q 27.18. Confident in the unbreakability of RSA, I write the following. What mistakes have I made?

0000001 0000000 0002048 0000001 1391142
0000000 0177147 1033288 1391142 1174371.

Advise me on how to increase the security of messages.

Q 27.19. Let K be the finite field with 2^d elements and primitive root α. (Recall that α is a generator of the cyclic group $K \setminus \{0\}$ under multiplication.) Let $T : K \to \mathbb{F}_2$ be a non-zero linear map. (Here we treat K as a vector space over $\mathbb{F}_2$.)

(i) Show that the map $S : K \times K \to \mathbb{F}_2$ given by $S(x, y) = T(xy)$ is a symmetric bilinear form. Show further that S is non-degenerate (that is to say $S(x, y) = 0$ for all x implies $y = 0$).

(ii) Show that the sequence $x_n = T(\alpha^n)$ is the output from a linear feedback register of length at most d. (Part (iii) shows that it must be exactly d.)

(iii) Show that the period of the system (that is to say the minimum period of T) is $2^d - 1$. Explain briefly why this is best possible.