*Department of Pure Mathematics and Mathematical Statistics*
*University of Cambridge*

# CODES AND CRYPTOGRAPHY


*The Enigma Cryptography Machine*

**Notes Michaelmas 2013**

**T. K. Carne.**
*t.k.carne@dpmms.cam.ac.uk*

Last revised: 21 November, 2013

## CONTENTS

## 6: NOISY CHANNELS

So far we have assumed that each coded message is transmitted without error through the channel to the receiver where it can be decoded. In many practical applications, there is a small but appreciable probability of error, so a letter may be altered in the channel. We want to devise codes that detect, or even correct, such errors.

### 6.1 Memoryless Channels

A coded message $b_1b_2b_3\dots b_N \in \mathcal{B}^*$ is sent through a channel $C$ where each letter $b_n$ may be altered to another. We will assume that any changes to one letter are independent of the other letters and what happens to them. In this case we say the channel is *memoryless*. So, if we write $B_n$ for the random variable that gives the $n$th letter of the message and $B'_n$ for the random variable giving the $n$th letter received, then

$$\mathbb{P}(b'_1b'_2b'_3\dots b'_N \text{ received } | b_1b_2b_3\dots b_N \text{ sent }) = \prod_{n=1}^{N} \mathbb{P}(B'_n = b'_n | B_n = b_n) \ .$$

This gives a *transition matrix*

$$\mathbb{P}(B'_n = i | B_n = j) \qquad \text{for } i,j \in \mathcal{B}$$

describing how the $n$th letter is altered. We expect the probability of error to be small, so the probability $\mathbb{P}(B'_n = i | B_n = i)$ should be close to 1.

We will also assume that the chances of a particular alteration does not depend on the index $n$, so the channel is *time independent*. Then the transition matrices do not change with $n$.

---

**Example:**
Let $\mathcal{B} = \{0,1,2,3,4,5,6,7,8,9\}$ and the transition matrix be

$$\mathbb{P}(j \text{ received } | i \text{ sent }) = \begin{cases} \frac{3}{4} & \text{when } j = i; \\ \frac{1}{4} & \text{when } j = i+1; \\ 0 & \text{otherwise.} \end{cases}$$

If I send a word $b_1b_2b_3\dots b_N$, the probability that it is received without error is only $(\frac{3}{4})^N$.

I can improve the chances of receiving the message without error by sending each letter 3 times in succession. The probability that at least 2 of the 3 are received without error is then

$$\left(\tfrac{3}{4}\right)^3 + 3\left(\tfrac{1}{4}\right)\left(\tfrac{3}{4}\right)^2 = \tfrac{27}{32} = 0.84375$$

which is larger than $\frac{3}{4}$. So we have increased the probability of decoding the message without error to $\left(\frac{27}{32}\right)^N$ albeit at the price of sending a message three times as long.

We can do much better than this. For suppose that we first recode our message using the shorter alphabet $\mathcal{E} = \{0,2,4,6,8\}$. If a letter $i \in \mathcal{E}$ is sent, then either $i$ or $i+1$ is received. In either case we can tell **with certainty** that $i$ was sent. So we can decode the message perfectly.

---

### 6.2 The Binary Symmetric Channel

For most of this course we will use binary codes for simplicity. When $\mathcal{B} = \{0,1\}$, the *binary symmetric channel* (BSC) is a time independent, memoryless channel with transition matrix

$$\begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix} \ .$$

Here $p$ is the probability of error in a single bit. Usually it is small. Note that if $p = \frac{1}{2}$ then no information is transmitted, everything is noise. When $p = 0$ there is no noise and no errors to be corrected.

---

**Exercise:**
In a binary symmetric channel we usually take the probability $p$ of error to be less than $1/2$. Why do we not consider $1 \geqslant p \geqslant 1/2$?

---

**Example:**
Punched computer tapes traditionally had 8 spaces per row and used holes to indicate the 1 bit. Of these bits the first 7 $(x_1, x_2, \ldots, x_7)$ carried information while the last $x_8$ was a check digit satisfying:

$$x_1 + x_2 + \ldots x_7 + x_8 \equiv 0 \pmod 2 \ . \qquad *$$

We can model the chances of errors in the punched tape using a binary symmetric channel with error probability $p$. Then the probability of receiving the information $x_1x_2 \ldots x_7$ without error is $(1-p)^7$. If there is a one error in the 8 bits $x_1x_2 \ldots x_7x_8$ then $(*)$ will not hold for the received message and this will show that there was an error. In this case the information could be sent again.

Note that, when 2 errors occur, $(*)$ remains true and we do not know that there has been an error.

For instance, if $p = 0.1$, then the probability of 0 errors in $x_1x_2 \ldots x_7$ is 0.48, the probability of 1 error that is detected by the check digit is 0.33.

---

**Exercise:**
Show that the probability of detecting errors at all using $(*)$ is $\frac{1}{2}(1 - (1-2p)^8)$.

---

### 6.3 Check Digits

Check digits are used very widely to detect errors.

---

**Example:**
University Candidate Numbers are of the form

$$1234A, \ 1235B, \ 1236C, \ 1237D, \ \ldots \ .$$

The first 4 digits identify the candidate, while the final letter is a check digit. If a candidate writes one letter incorrectly, then he or she will produce an invalid candidate number. The desk number can then be used to correct the error.

---

**Exercise:**
Books are identified by the ISBN-10 code. For example:

ISBN 0-521-404568

This consists of 9 decimal digits $x_1$-$x_2x_3x_4$-$x_5x_6x_7x_8x_9$ which identify the publisher, author and title. The final digit $x_{10} \in \{0, 1, 2, \ldots, 9, X\}$ is a check digit which satisfies

$$10x_1 + 9x_2 + 8x_3 + \ldots + 2x_9 + x_{10} \equiv 0 \pmod{11} \ .$$

Check the ISBN number above is valid. Show that altering any single digit or transposing any two adjacent digits in an ISBN-10 code will always result in an invalid code. Hence such errors can be detected.

How do check digits work in ISBN-13? Do they detect all single digit errors?

---

### 6.4 The Hamming Code

In the earlier lectures we saw that, when there was redundant information in a message, we could compress it. Alternatively, we can expand a code to add redundant information. This gives us the possibility of detecting errors or even correcting them.

Hamming produced a binary code that can correct a single error. This code takes a binary word of length 4 and codes it as a binary word of length 7. When we receive a word we look for the closest possible code word. Provided that there has been no more than one error in the 7 bits, we can always find that code word correctly and hence decode it.

The integers modulo 2 form a field $\mathbb{F}_2$ and we can use these to represent binary bits. So a binary string of length $N$ is represented by a vector in the $N$-dimensional vector space $\mathbb{F}_2^N$ over $\mathbb{F}_2$. Hamming's code is given by a linear map $C : \mathbb{F}_2^4 \to \mathbb{F}_2^7$. This map is given by the matrix:

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

So a word $\boldsymbol{x} \in \mathbb{F}_2^4$ is coded as

$$C\boldsymbol{x} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_2 + x_3 + x_4 \\ x_1 + x_3 + x_4 \\ x_1 + x_2 + x_4 \end{pmatrix} .$$

The first 4 bits carry the information and the remaining 3 are check digits.

We can check that each column $\boldsymbol{y}$ of the matrix $C$ satisfies

$$\begin{aligned} y_1 + y_3 + y_5 + y_7 &= 0 \\ y_2 + y_3 + y_6 + y_7 &= 0 \\ y_4 + y_5 + y_6 + y_7 &= 0 \, . \end{aligned} \qquad \dagger$$

Hence each code word $\boldsymbol{y} = C\boldsymbol{x}$ must also satisfy these conditions (†). The converse is also true. Suppose that $\boldsymbol{y}$ satisfies (†). Then the equations determine $y_5, y_6, y_7$ in terms of $y_1, y_2, y_3, y_4$. So we see that

$$\boldsymbol{y} = C\boldsymbol{x} \qquad \text{where} \qquad \boldsymbol{x} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix} \in \mathbb{F}_2^4 \, .$$

Hence the equations (†) determine precisely when $\boldsymbol{y}$ is a code word or not.

We can rewrite the linear equations (†) in matrix terms as

$$S\boldsymbol{y} = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \boldsymbol{y} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} .$$

This matrix $S$ is called the *syndrome matrix*. We have seen that $SC\boldsymbol{x} = \boldsymbol{0}$ for every string $\boldsymbol{x} \in \mathbb{F}_2^4$. Indeed, $S \circ C = 0$ and the image $C(\mathbb{F}_2^4)$ is equal to the kernel $\ker S$.

Suppose that the string $\boldsymbol{x} \in \mathbb{F}_2^4$ is encoded as $C\boldsymbol{x}$ but an error occurs in the $j$th bit and nowhere else. Then we receive

$$\boldsymbol{y} = C\boldsymbol{x} + \boldsymbol{e}_j$$

where $\boldsymbol{e}_j$ is the vector with 1 in the $j$th place and 0s elsewhere. We know that $C\boldsymbol{x}$ satisfies (†) but $\boldsymbol{e}_j$ does not. Hence

$$S\boldsymbol{y} = SC\boldsymbol{x} + S\boldsymbol{e}_j \ ,$$

which is the $j$th column of the syndrome matrix $S$. Note that this $j$th column is simply the bits of $j$ expanded in binary and reversed. For example

$$S\boldsymbol{e}_6 = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} \qquad \text{corresponds to} \qquad 110_2 = 6 \ .$$

In this way we can tell where the error occurred. Since the error is a change from 0 to 1 or 1 to 0, we can then correct it.

If I receive a message $\boldsymbol{y}$ in which two bits have errors, say $\boldsymbol{y} = C\boldsymbol{x} + \boldsymbol{e_i} + \boldsymbol{e}_j$, then the syndrome is

$$S\boldsymbol{y} = S\boldsymbol{e}_i + S\boldsymbol{e}_j \ .$$

Since $i$ and $j$ are different, the syndromes $S\boldsymbol{e}_i$ and $S\boldsymbol{e}_j$ are also different. So their sum $S\boldsymbol{e}_i + S\boldsymbol{e}_j$ is not zero. Hence $S\boldsymbol{y}$ is non-zero and tells us that there have been errors. However, it does not tell us what those errors are. For example, if we receive the string 1000000, then the syndrome is 100. This may have arisen from the code word 0000000 with one error or the code word 1000011 with two errors.

To summarise, the Hamming code detects 2 errors and can correct 1 error.

We can measure the distance between two vectors $\boldsymbol{y}$ and $\boldsymbol{z}$ in $\mathbb{F}_2^N$ by counting the number of places where they differ. This gives

$$d(\boldsymbol{y}, \boldsymbol{z}) = |\{j : y_j \neq z_j\}| \ ,$$

which is called the *Hamming distance*. So the Hamming distance is the number of bit errors required to change $\boldsymbol{y}$ to $\boldsymbol{z}$. It is simple to see that this is a metric on $\mathbb{F}_2^N$ and we will always use this metric.

Let $\boldsymbol{x}$ and $\boldsymbol{x}'$ be two different strings in $\mathbb{F}_2^4$, and $C\boldsymbol{x}, C\boldsymbol{x}'$ the corresponding code words. Their difference

$$C\boldsymbol{x}' - C\boldsymbol{x} = C(\boldsymbol{x}' - \boldsymbol{x})$$

is the sum of some of the columns of the matrix $C$. It is easy to check that each such sum has at least 3 non-zero bits. So $d(C\boldsymbol{x}', C\boldsymbol{x})$ must be at least 3. It can be 3, for example $d(1000011, 0001111) = 3$.

Suppose that we receive a string $\boldsymbol{y} \in \mathbb{F}_2^7$ and wish to decode it. If $\boldsymbol{y}$ is a code word $C\boldsymbol{x}$ then we decode it as $\boldsymbol{x}$. If there is one error in $\boldsymbol{y}$, we can use the syndrome to find and correct that error. So we find a string $\boldsymbol{x} \in \mathbb{F}_2^4$ with $d(\boldsymbol{y}, C\boldsymbol{x}) = 1$. If there are two errors, it is natural to look for the string $\boldsymbol{x} \in \mathbb{F}_2^4$ for which $C\boldsymbol{x}$ is closest to $\boldsymbol{y}$, that is

$$d(\boldsymbol{y}, C\boldsymbol{x})$$

is minimal. We would then guess that $\boldsymbol{y}$ arose from $\boldsymbol{x}$ with $d(\boldsymbol{y}, C\boldsymbol{x})$ errors. The guess may not be unique. For example, if we receive 0000001, this is at distance 2 from both $1000011 = C(1000)$ and $0100101 = C(0100)$.

## 7: ERROR CORRECTING CODES

In this lecture we will consider only codes $c : \mathcal{A} \to \mathcal{B}^N$ of constant length $N$. We will use the discrete metric on $\mathcal{B}$:

$$d(b, v) = \begin{cases} 0 & \text{when } b = v; \\ 1 & \text{when } b \neq v \ . \end{cases}$$

The *Hamming distance* on $\mathcal{B}^N$ is then

$$d(\boldsymbol{b}, \boldsymbol{v}) = \sum_{j=1}^{N} d(b_j, v_j) \ .$$

It is clear that this is a metric on $\mathcal{B}^N$. Note that the Hamming distance $d(\boldsymbol{c}, \boldsymbol{v})$ counts the number of co-ordinates where $\boldsymbol{b}$ and $\boldsymbol{v}$ differ. When a word $\boldsymbol{b} \in \mathcal{B}^N$ is transmitted through a noisy channel, we receive an altered word $\boldsymbol{v}$. The Hamming distance $d(\boldsymbol{b}, \boldsymbol{v})$ counts the number of letters that have been altered.

### 7.1 Decoding with Errors

Suppose that a word $\boldsymbol{b} \in \mathcal{B}^N$ is sent through a noisy channel and received as $\boldsymbol{v} \in \mathcal{B}^N$. We then wish to decode this. Because of the errors introduced by the channel, the word we receive may not be a code word at all and certainly not the code word $\boldsymbol{b}$ that was sent. How should we guess $\boldsymbol{b}$ when we know $\boldsymbol{v}$?

We will consider, briefly, three possible decoding rules. Suppose that $\boldsymbol{v} \in \mathcal{B}^N$ has been received.

**Ideal Observer**
Decode $\boldsymbol{v}$ as the word $\boldsymbol{b}$ with $\mathbb{P}(\boldsymbol{b}$ sent $\mid \boldsymbol{v}$ received $)$ maximal. This is a sensible choice if we know enough to find the maximum. In most cases we do not.

**Maximum Likelihood**
Decode $\boldsymbol{v}$ as the word $\boldsymbol{b}$ with $\mathbb{P}(\boldsymbol{v}$ received $\mid \boldsymbol{b}$ sent $)$ maximal. This is usually easier to find.

**Minimum Distance**
Decode $\boldsymbol{v}$ as the word $\boldsymbol{b}$ with $d(\boldsymbol{b}, \boldsymbol{v})$ minimal.

**Proposition 7.1**
*If all the code words are equally likely, then the ideal observer and maximum likelihood rules give the same result.*

*Proof:*
We know that

$$\mathbb{P}(\boldsymbol{v} \text{ received} \mid \boldsymbol{b} \text{ sent}) = \frac{\mathbb{P}(\boldsymbol{v} \text{ received and } \boldsymbol{b} \text{ sent})}{\mathbb{P}(\boldsymbol{b} \text{ sent})} = \mathbb{P}(\boldsymbol{b} \text{ sent} \mid \boldsymbol{v} \text{ received}) \frac{\mathbb{P}(\boldsymbol{v} \text{ received})}{\mathbb{P}(\boldsymbol{b} \text{ sent})} \ .$$

So, if all the code words are equally likely, then the ideal observer rule and the maximum likelihood rule give the same result. □

**Proposition 7.2**
*If letters are transmitted through a binary symmetric channel with error probability $p < \frac{1}{2}$, then the maximum likelihood and minimum distance rules give the same result.*

*Proof:*

$$\mathbb{P}(\boldsymbol{v} \text{ received} \mid \boldsymbol{b} \text{ sent }) = \prod_{j=1}^{N} \mathbb{P}(v_j \text{ received} \mid b_j \text{ sent }) = p^d(1-p)^{N-d} = (1-p)^N \left(\frac{p}{1-p}\right)^d$$

where $d$ is the number of terms where $v_j \neq b_j$. Hence $d$ is the Hamming distance $d(\boldsymbol{v}, \boldsymbol{b})$. Since $0 \leqslant p/(1-p) < 1$ we see that $\mathbb{P}(\boldsymbol{v} \text{ received} \mid \boldsymbol{b} \text{ sent })$ is maximal when $d(\boldsymbol{b}, \boldsymbol{v})$ is minimal. □

We will generally use the minimum distance decoding rule.

## 7.2 Error Detecting and Error Correcting

Throughout this section we will use binary codes $c : \mathcal{A} \to \{0,1\}^N$. This is for simplicity since the results can easily be adapted to more general alphabets. Recall that the *code words* are the words $c(a)$ for $a \in \mathcal{A}$. We will write $K$ for the number of these $K = |\mathcal{A}|$. If a word $\boldsymbol{b} = c(a)$ is sent, then we receive a word $\boldsymbol{v} \in \{0,1\}^N$. We decode this by finding the code word $c(a')$ closest to $\boldsymbol{v}$, so $d(c(a'), \boldsymbol{v})$ is minimal.

The ball of radius $r$ centred on $\boldsymbol{b} \in \{0,1\}^N$ is

$$B(\boldsymbol{b}, r) = \{\boldsymbol{v} \in \{0,1\}^N : d(\boldsymbol{b}, \boldsymbol{v}) < r\} \ .$$

Its *volume* is the number of points that it contains:

$$|B(\boldsymbol{b}, r)| = \sum_{0 \leqslant k < r} \binom{N}{k} \ .$$

This is clearly independent of the centre $\boldsymbol{b}$ and we will denote it by $V(N, r)$. It is reasonably simple to estimate the size of $V(N, r)$ and we will do so later.

The code $c : \mathcal{A} \to \{0,1\}^N$ is *e-error detecting* if, whenever no more than $e$ letters in a code word $c(a)$ are altered, the received word is not a different code word $c(a')$. This means that we can tell if there have been errors, provided that there are at most $e$ of them.

The code $c : \mathcal{A} \to \{0,1\}^N$ is *e-error correcting* if, whenever no more than $e$ letters in a code word $c(a)$ are altered, the received word is still decoded as $a$ using the minimum distance decoding rule.

The *minimum distance* for a code $c : \mathcal{A} \to \{0,1\}^N$ is

$$\delta = \min \{d(c(a), c(a')) : a, a' \in \mathcal{A} \ \text{ with } a \neq a'\} \ .$$

For this we immediately have:

**Proposition 7.3** Minimum distance for a code
*Let the code $c : \mathcal{A} \to \{0,1\}^N$ have minimum distance $\delta > 0$.*

*(a) $c$ is $(\delta - 1)$-error detecting and not $\delta$-error detecting.*

*(b) $c$ is $\lfloor \frac{1}{2}(\delta - 1) \rfloor$-error correcting but not $\lfloor \frac{1}{2}(\delta + 1) \rfloor$-error correcting*

Note that $\lfloor \frac{1}{2}(\delta+1) \rfloor = \lfloor \frac{1}{2}(\delta-1) \rfloor \ + \ 1$.

*Proof:*

(a) Clearly no code word other than $c(a)$ can lie within a distance $\delta - 1$ of $c(a)$. However, there are two letters $a_1, a_2 \in \mathcal{A}$ with $d(c(a_1), c(a_2)) = \delta$, so $c$ is not $\delta$-error detecting.

(b) If $a' \neq a$, and $d(c(a), \boldsymbol{v}) \leqslant \lfloor \frac{1}{2}(\delta-1) \rfloor$, then the triangle inequality gives

$$d(c(a'), \boldsymbol{v}) \geqslant d(c(a'), c(a)) - d(c(a), \boldsymbol{v}) \geqslant \delta - \lfloor \tfrac{1}{2}(\delta-1) \rfloor > \lfloor \tfrac{1}{2}(\delta-1) \rfloor \ .$$

So $c$ is $\lfloor \frac{1}{2}(\delta-1) \rfloor$-error detecting. However, there are two letters $a_1, a_2 \in \mathcal{A}$ with $d(c(a_1), c(a_2)) = \delta$, so $c(a_1)$ and $c(a_2)$ differ in exactly $\delta$ places. Choose $\lfloor \frac{1}{2}(\delta+1) \rfloor$ of these places and change $c(a_1)$ in these to get a word $\boldsymbol{v}$ with

$$d(\boldsymbol{v}, c(a_1)) = \lfloor \tfrac{1}{2}(\delta+1) \rfloor \qquad \text{and} \qquad d(\boldsymbol{v}, c(a_2)) = \delta - \lfloor \tfrac{1}{2}(\delta+1) \rfloor \leqslant \lfloor \tfrac{1}{2}(\delta+1) \rfloor \ .$$

So $c$ can not be $\lfloor \frac{1}{2}(\delta+1) \rfloor$-error correcting. □

---

**Example:**
For the repetition code, where each letter is repeated $m$-times, the minimum distance is $m$, so this code is $(m-1)$-error detecting and $\lfloor \frac{1}{2}(m-1) \rfloor$-error correcting.

For a code $\mathcal{A} \to \{0,1\}^N$ where the $N$th bit is a check digit, the minimum distance is 2, so this code is 1-error detecting and 0-error correcting.

For the Hamming code $h : \{0,1\}^4 \to \{0,1\}^7$, the minimum distance is 3, so the Hamming code is 2-error detecting and 1-error correcting.

---

### 7.3 Covering Estimates

Observe that the code $c : \mathcal{A} \to \{0,1\}^N$ is $e$-error correcting precisely when the balls $B(c(a), e+1)$ for $a \in \mathcal{A}$ are disjoint. We can use this to estimate the number of code words for such a code.

**Proposition 7.4** Hamming's bound
*If $c : \mathcal{A} \to \{0,1\}^N$ is an $e$-error correcting code, then the number of code words $K = |\mathcal{A}|$ must satisfy*

$$K \leqslant \frac{2^N}{V(N, e+1)} \ .$$

*Proof:*
First note that

$$B(\boldsymbol{c}, e+1) = \{\boldsymbol{v} \in \{0,1\}^N : d(\boldsymbol{c}, \boldsymbol{v}) < e+1\} = \{\boldsymbol{v} \in \{0,1\}^N : d(\boldsymbol{c}, \boldsymbol{v}) \leqslant e\}$$

so the open ball $B(\boldsymbol{c}, e+1)$ is the closed ball of radius $e$.

When $c$ is $e$-error correcting, the balls $B(c(a), e+1)$ for $a \in \mathcal{A}$ must be disjoint. Hence their total volume $KV(N, e+1)$ is at most equal to the volume of all of $\{0,1\}^N$, which is $2^N$. □

A code is said to be *perfect* if it is $e$-error correcting for some $e$ and the balls $B(c(a), e+1)$ cover all of $\{0,1\}^N$. This is the case when there is equality in Hamming's bound. For example, Hamming's code is 1-error correcting, $K = 2^4$, $N = 7$ and

$$V(7,2) = \binom{7}{0} + \binom{7}{1} = 8 \ .$$

So $K = 2^4 = 2^7/8 = 2^N/V(7,2)$ and the code is perfect.

We can also use a similar idea to give a lower bound.

**Proposition 7.5** Gilbert – Shannon - Varshamov bound
*There is an $e$-error detecting code $c : \mathcal{A} \to \{0,1\}^N$ with the number of code words $K = |\mathcal{A}|$ satisfying*

$$K \geqslant \frac{2^N}{V(N, e+1)} \ .$$

*Proof:*

For an $e$-error detecting code, no code word other than $c(a)$ can lie within the ball $B(c(a), e+1)$. Choose a maximal set $\mathcal{C}$ of code words satisfying this condition and let $K = |\mathcal{C}|$. If there were any word $\boldsymbol{v}$ not in the union

$$\bigcup_{\boldsymbol{c}\in\mathcal{C}} B(\boldsymbol{c}, e+1)$$

then we could add it to $\mathcal{C}$, contradicting maximality. Therefore,

$$\bigcup_{\boldsymbol{c}\in\mathcal{C}} B(\boldsymbol{c}, e+1) = \{0,1\}^N \ .$$

The volume of each ball is $V(N, e+1)$, so the volume of their union is no more than $KV(N, e+1)$. Hence $KV(N, e+1) \geqslant 2^N$. □

**Corollary 7.6**
*Let $c : \mathcal{A} \to \{0,1\}^N$ be a code with minimum distance $\delta$. Then*

$$K = |\mathcal{A}| \leqslant \frac{2^N}{V(N, \lfloor \frac{1}{2}(\delta+1) \rfloor)} \ .$$

*Moreover, there is a code with minimum distance $\delta$ and*

$$K = |\mathcal{A}| \geqslant \frac{2^N}{V(N, \delta)} \ .$$

□

### 7.4 Asymptotics for $V(N, r)$

We wish to use the Hamming and Gilbert – Shannon – Varshamov bounds of the last section to give asymptotic estimates on how large error correcting codes are. To do this we need to determine the asymptotic behaviour of the volumes $V(N, r)$. It will be useful to describe this in terms of the entropy of a Bernoulli random variable that takes two values with probability $q$ and $1-q$. We will denote this by

$$h(q) = -q \log_2 q - (1-q) \log_2 (1-q) \ .$$

**Lemma 7.7** Asymptotics of binomial coefficients
*For natural numbers $0 \leqslant r \leqslant N$ we have*

$$\frac{1}{N+1} 2^{Nh(q)} \leqslant \binom{N}{r} \leqslant 2^{Nh(q)}$$

*where $q = r/N$.*

*Proof:*

Let $X$ be a Bernoulli $B(N,q)$ random variable, so

$$\mathbb{P}(X=k) = \binom{N}{k} q^k (1-q)^{N-k} \ .$$

(The entropy of $X$ is $H(X) = Nh(q)$.) It is simple to check that this probability is maximal when $k = r = qN$. Therefore,

$$\mathbb{P}(X=r) \ \leqslant \ \sum_{k=0}^{N} \mathbb{P}(X=k) \ = \ 1 \ \leqslant \ (N+1)\mathbb{P}(X=r) \ .$$

This means that

$$\frac{1}{N+1} \ \leqslant \ \mathbb{P}(X=r) = \binom{N}{r} q^r (1-q)^{N-r} = \binom{N}{r} 2^{-Nh(q)} \ \leqslant 1 \ .$$

The result follows when we observe that

$$2^{-Nh(q)} = q^{Nq}(1-q)^{N(1-q)} = q^r (1-q)^{N-r} \ .$$

□

---

**Exercise:**
Use Stirling's formula $\left[ n! \sim \sqrt{2\pi n}\frac{n^n}{e^n} \right]$ to describe the behaviour of $\binom{N}{r}$ in terms of the entropy $h(r/N)$.

---

**Proposition 7.8** Asymptotics of $V(N,r)$
*For a natural number $N$ and $0 < r < \frac{1}{2}N$*

$$V(N,r) \ \leqslant \ 2^{Nh(q)}$$

*for $q = r/N$ and*

$$\frac{1}{N+1} 2^{Nh(q')} \ \leqslant \ V(N,r)$$

*for $q' = (\lceil r \rceil - 1)/N$.*

*Proof:*

$V(N,r)$ is the number of points in an open ball of radius $r$ in $\{0,1\}^N$, so

$$V(N,r) = \sum_{0 \leqslant k < r} \binom{N}{k} \ .$$

Since $q = r/N \leqslant \frac{1}{2}$, we have $q^k(1-q)^{N-k} \leqslant q^r(1-q)^{N-r}$ for $0 \leqslant k < r$. So

$$\begin{aligned} 1 = \sum_{0 \leqslant k \leqslant N} \binom{N}{k} q^k (1-q)^{N-k} &\geqslant \sum_{0 \leqslant k < r} \binom{N}{k} q^k (1-q)^{N-k} \\ &\geqslant \left( \sum_{0 \leqslant k < r} \binom{N}{k} \right) q^r (1-q)^{N-r} = V(N,r) q^r (1-q)^{N-r} \ . \end{aligned}$$

This shows that $V(N,r) \ \leqslant \ 2^{Nh(q)}$.

For the other inequality, let $k = \lceil r \rceil - 1$ so $k$ is the largest integer strictly smaller than $r$. Then

$$V(N,r) \geqslant \binom{N}{k} \geqslant \frac{1}{N+1} 2^{Nh(q')}$$

because of Lemma 7.7 . □

Consider a code $c : \mathcal{A} \to \{0,1\}^N$. If $A$ is a a random variable taking values in the alphabet $\mathcal{A}$ then information is transmitted through the channel at a rate

$$\frac{H(A)}{N} \ .$$

This is largest when $A$ is equally distributed over the $K$ possible values. Then it is

$$\frac{\log_2 K}{N} \ .$$

Choose $q$ with $0 < q < \frac{1}{2}$. Hamming's bound ( Proposition 7.4 ) shows that

$$\frac{\log_2 K}{N} \leqslant 1 - \frac{\log_2 V(N, qN)}{N}$$

for $(qN-1)$-error correcting codes. The last proposition now shows that the right side tends to $1-h(q)$ as $N \to \infty$.

Similarly, the Gilbert – Shannon – Varshamov bound Proposition 7.5 shows that there are $(qN-1)$-error detecting codes with information rate

$$\frac{\log_2 K}{N} \geqslant 1 - \frac{\log_2 V(N, qN)}{N}$$

and the right side tends to $1 - h(q)$ as $N \to \infty$.

## 8: INFORMATION CAPACITY

### 8.1 Mutual Information

We wish to determine how much information can be passed successfully through a noisy channel for each bit that is transmitted. Consider a time-independent, memoryless channel. If a random letter $B \in \mathcal{B}$ is sent through the channel, then a new random letter $B'$ is received. Because the channel is noisy, $B'$ may differ from $B$.

The average amount of information given by $B$ is the entropy $H(B)$, while the average amount of information received is $H(B')$. The information received is partly due to $B$ and partly due to the noise. The conditional entropy $H(B'|B)$ is the amount of information in $B'$ conditional on the value of $B$, so it is the information due to the noise in the channel. The remaining information

$$H(B') - H(B'|B)$$

is the part that is due to the letter $B$ that was sent. It is the amount of uncertainty about $B'$ that is removed by knowing $B$. We therefore define the *mutual information of $B$ and $B'$* to be

$$I(B', B) = H(B') - H(B'|B) \ .$$

**Proposition 8.1** Mutual Information
*The mutual information satisfies:*

$$I(B', B) = H(B') + H(B) - H(B', B) = I(B, B') \ .$$

*Furthermore,*

*(a) $I(B', B) \geqslant 0$ with equality if and only if $B'$ and $B$ are independent.*

*(b) $I(B', B) \leqslant H(B')$ with equality if and only if $B'$ is a function of $B$.*

*(c) $I(B', B) \leqslant H(B)$ with equality if and only if $B$ is a function of $B'$.*

*Proof:*

Recall from Lecture 2 that the conditional entropy satisfies $H(B'|B) = H(B', B) - H(B)$, so

$$I(B', B) = H(B') - H(B'|B) = H(B') - H(B', B) + H(B)$$

and the left side is clearly symmetric.

(a) Corollary 2.2 shows that $I(B', B) \geqslant 0$ with equality if and only if $B$ and $B'$ are independent.

(b) We always have $H(B'|B) \geqslant 0$ with equality if and only if $B'$ is a function of $B$.

(c) Since $I(B', B) = I(B, B')$ part (c) follows from (b). □

In our case, the distribution $\mathbb{P}(B = b)$ is known as are the transition probabilities $\mathbb{P}(B' = b'|B = b)$. From these we can calculate both

$$\mathbb{P}(B' = b', B = b) = \mathbb{P}(B' = b'|B = b)\mathbb{P}(B = b)$$

and

$$\mathbb{P}(B' = b') = \sum_{b \in \mathcal{B}} \mathbb{P}(B' = b'|B = b)\mathbb{P}(B = b) \ .$$

Thence we can find the entropies $H(B), H(B'), H(B', B)$ and the mutual information. If the channel has no noise, then $B' = B$ and the proposition shows that the mutual information is just the entropy $H(B)$.

The *information capacity* of the channel is the supremum of the mutual information $I(B', B)$ over all probability distributions for $B$. The probability distribution of $B$ is given by a point $\boldsymbol{p} = (p(a_1), p(a_2), \ldots, p(A_K))$ in the compact set $\{\boldsymbol{p} \in [0,1]^K : \sum p_k = 1\}$. The mutual information is a continuous function of $\boldsymbol{p}$, so we know that the supremum is attained for some distribution. This information capacity depends only on the transition probabilities $\mathbb{P}(B'|B)$.

**Proposition 8.2** Information capacity of a BSC
*A binary symmetric channel with probability p of error has information capacity* $1 - h(p)$.

*Proof:*

Suppose that the letter $B$ has the distribution $\mathbb{P}(B = 1) = t \ , \ \ \mathbb{P}(B = 0) = 1 - t$. Then the entropy of $B$ is

$$H(B) = H(1 - t, t) = -t \log_2 t - (1 - t) \log_2 (1 - t) = h(t) \ .$$

The conditional entropy is

$$\begin{aligned} H(B'|B) &= \mathbb{P}(B = 1)H(B'|B = 1) + \mathbb{P}(B = 0)H(B'|B = 0) \\ &= tH(1 - p, p) + (1 - t)H(p, 1 - p) = h(p) \ . \end{aligned}$$

The distribution of $B'$ is

$$\mathbb{P}(B' = 1) = t(1 - p) + (1 - t)p = t + p - 2tp \ , \ \ \mathbb{P}(B' = 0) = tp + (1 - t)(1 - p) = 1 - t - p + 2tp$$

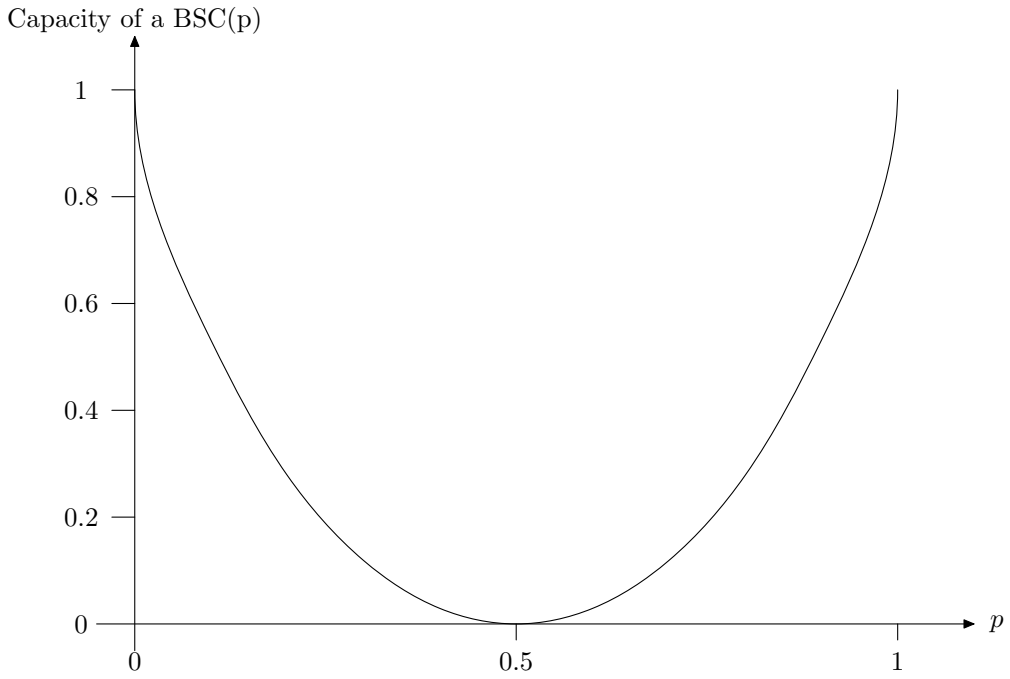
so its entropy is

$$H(B') = h(t + p - 2tp) \ .$$

Therefore the mutual information is

$$I(B', B) = H(B') - H(B'|B) = h(t + p - 2tp) - h(p) \ .$$

We can choose any distribution for $B$ and so take any $t$ between 0 and 1. The maximum value for $h(t + p - 2tp)$ is 1 which is attained when $t = \frac{1}{2}$ and $t + p - 2tp = \frac{1}{2}$. So the information capacity of the binary symmetric channel is $1 - h(p)$. □




Note that, when $p = 0$ or 1, the channel transmits perfectly and the information capacity is 1. When $p = \frac{1}{2}$, no information is transmitted and the capacity is 0.

We often wish to consider using the same channel repeatedly, for example, when we wish to transmit a word in $\{0,1\}^N$ by sending the $N$ bits one at a time through a binary symmetric channel. We can also think of this as having $N$ copies of the channel in parallel and sending the $j$th bit through the $j$th channel. It is simple to compute the capacity in such a situation.

**Proposition 8.3** The capacity of parallel channels
*Let $Q$ be a channel that transmits letters from the alphabet $\mathcal{B}$ and has capacity* $\mathrm{Cap}(Q)$. *Form a new channel $Q^N$ that transmits an $N$-tuple $(b_1, b_2, \ldots, b_N) \in \mathcal{B}^N$ one letter at a time through the channel $Q$ with each use being independent of the others. Then the capacity of $Q^N$ is* $N\mathrm{Cap}(Q)$.

*Proof:*

Let $\boldsymbol{B} = (B_1, B_2, \ldots, B_N)$ be a random variable taking values in the product alphabet $\mathcal{B}^N$ and let $\boldsymbol{B}' = (B'_1, B'_2, \ldots, B'_N)$ be the random variable we receive after sending $\boldsymbol{B}$ through the channel $Q^N$.

For each vector $\boldsymbol{b} = (b_1, b_2, \ldots, b_N)$ we know that

$$H(\boldsymbol{B}'|\boldsymbol{B} = \boldsymbol{b}) = \sum H(B'_j|B_j = b_j)$$

because, once we condition on the $\boldsymbol{B}$, each $B'_j$ is independent of all the other letters received. Therefore,

$$H(\boldsymbol{B}'|\boldsymbol{B}) = \sum H(B'_j|B_j) \ .$$

Also,

$$H(\boldsymbol{B}') \leqslant \sum H(B'_j)$$

with equality if and only if the $(B'_j)$ are independent.

Thus

$$I(\boldsymbol{B}', \boldsymbol{B}) = H(\boldsymbol{B}') - H(\boldsymbol{B}'|\boldsymbol{B}) \leqslant \sum_{j=1}^{N} H(B'_j) - H(B'_j|B_j) = \sum_{j=1}^{N} I(B'_j, B_j)$$

and there is equality if and only if the $(B'_j)$ are independent. This shows that $\mathrm{Cap}(Q^N) \leqslant N\mathrm{Cap}(Q)$.

Now choose the distribution of $B_j$ so that $\mathrm{Cap}(Q) = I(B'_j, B_j)$ and the $(B_j)$ are independent. Then the $(B'_j)$ are also independent and

$$I(\boldsymbol{B}', \boldsymbol{B}) = N\mathrm{Cap}(Q)$$

so the product channel $Q^N$ has capacity $N\mathrm{Cap}(Q)$. □

---

**Exercise:**
Use a similar argument to show that the capacity of independent channels $Q_j$ in parallel is $\sum \mathrm{Cap}(Q_j)$.

---

Suppose that we send a random variable $X$ through a channel and receive $Y$ with mutual information $I(X, Y)$. If we apply a function $f$ to $Y$ then it is intuitively clear that the information transmitted from $X$ to $f(Y)$ is no larger than that transmitted from $X$ to $Y$. The next Proposition proves this rigorously.

**Proposition 8.4** Functions do not increase mutual information
*Let $X, Y$ be random variable and $f$ a function defined on the values taken by $Y$, then*

$$I(X, f(Y)) \leqslant I(X, Y) \ .$$

*Similarly, $I(f(Y), X) \leqslant I(Y, X)$.*

*Proof:*

We know from Corollary 2.2 that $H(A, B) \leqslant H(A) + H(B)$. Applying this to the random variables conditioned on the value of $Z$ gives

$$H(X, Y|Z) \leqslant H(X|Z) + H(Y|Z) \ .$$

This is equivalent to

$$\begin{aligned} & H(X, Y, Z) - H(Z) \leqslant H(X, Z) - H(Z) + H(Y, Z) - H(Z) \\ \Leftrightarrow \quad & H(X) + H(Z) - H(X, Z) \leqslant H(X) + H(Y, Z) - H(X, Y, Z) \\ \Leftrightarrow \quad & I(X, Z) \leqslant I(X, (Y, Z)) \end{aligned}$$

If we set $Z = f(Y)$, then $H(Y, Z) = H(Y)$ and $H(X, Y, Z) = H(X, Y)$ so $I(X, (Y, Z)) = I(X, Y)$. Thus we have

$$I(X, f(Y)) \leqslant I(X, Y)$$

as required.

Since $I(X, Y)$ is symmetric ( Proposition 8.1 ), we also have $I(X, f(Y)) \leqslant I(X, Y)$. □

---

**Exercise:**
**Data Processing Inequality**
Consider two independent channels in series. A random variable $X$ is sent through channel 1 and received as $Y$. This is then sent through channel 2 and received as $Z$. Prove that $I(X, Z) \leqslant I(X, Y)$, so the further processing of the second channel can only reduce the mutual information. (The last proposition established this when the second channel has no noise.)

The independence of the channels means that, if we condition on the value of $Y$, then $(X|Y = y)$ and $(Z|Y = y)$ are independent. Deduce that

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

By writing the conditional entropies as $H(A|B) = H(A, B) - H(B)$, show that

$$H(X, Y, Z) + H(Z) = H(X, Y) + H(Y, Z) \ .$$

Define $I(X, Z|Y)$ as $H(X|Y) + H(Z|Y) - H(X, Z|Y)$ and show that

$$I(X, Z|Y) = I(X, Y) - I(X, Z) \ .$$

Deduce from this the *data processing inequality*:

$$I(X, Z) \leqslant I(X, Y) \ .$$

When is there equality?

---

## 9: FANO'S INEQUALITY

### 9.1 A Model for Noisy Coding

We will, eventually, prove Shannon's Coding Theorem which shows that we can find codes that transmit messages through a noisy channel with small probability of error and at a rate close to the information capacity of the channel. These are very good codes. Unfortunately, Shannon's argument is not constructive and it has proved very difficult to construct such good codes. In the remainder of this section we will show that the rate at which a code transmits information can not exceed the information capacity of the channel without the probability of error being large.

The situation we face is as follows. We wish to transmit a message written in the alphabet $\mathcal{A}$ of size $K$. Let $A$ be a random variable that takes values in $\mathcal{A}$. Then $H(A) \leqslant \log_2 K$ with equality when all $K$ letters in $\mathcal{A}$ are equally likely. We will assume that this is the case.

We choose a constant length code $c : \mathcal{A} \to \mathcal{B}^N$ to encode the message, one letter at a time. This gives a random variable $\boldsymbol{B} = c(A)$ taking values in $\mathcal{B}^N$. Since $c$ is assumed to be invertible, $A$ determines $\boldsymbol{B}$ and *vice versa*. Therefore $H(A) = H(\boldsymbol{B})$.

Each bit of $c(a)$ is then passed through a channel where errors may arise. This results in a new string $c(a)' \in \mathcal{B}^N$ and we set $\boldsymbol{B'} = c(A)'$. The closest code word to $c(a)'$ will be denoted by $c(a')$ for some $a' \in \mathcal{A}$ and we then decode it as $a'$. The random variable $A'$ results from decoding $\boldsymbol{B'}$. Since $A'$ is a function of $\boldsymbol{B'}$, we certainly have $H(A') \leqslant H(\boldsymbol{B'})$.

The probability of error when the letter $a$ is sent is

$$\mathbb{P}(\text{ error } \mid a \text{ sent })$$

and the total probability of error is

$$\mathbb{P}(\text{ error }) = \sum_{a \in \mathcal{A}} \mathbb{P}(\text{ error } \mid a \text{ sent })\mathbb{P}(A = a) \ .$$

We would like this to be small. Indeed, we would really like the probability to be small over all choices of $a \in \mathcal{A}$. So we want the *maximum error*

$$\widehat{e}(c) = \max\{\mathbb{P}(\text{ error } \mid a \text{ sent }) : a \in \mathcal{A}\}$$

to be small.

The *rate of transmission for* $c$ is

$$\rho(c) = \frac{H(A)}{N} = \frac{\log_2 K}{N} \ .$$

We want this to be as large as possible. Our aim is to relate this to the information capacity of the channel.

| | SENDER | | | CHANNEL | RECEIVER | | |
|---|---|---|---|---|---|---|---|
| | $\mathcal{A}$ | $\overset{c}{\longrightarrow}$ | $\mathcal{B}^N$ | - - - - - - - - - - - $\to$ | $\mathcal{B}^N$ | $\longrightarrow$ | $\mathcal{A}$ |
| Random Variables | $A$ | | $\boldsymbol{B}$ | Cap | $\boldsymbol{B'}$ | | $A'$ |
| Entropy | $H(A)$ | $=$ $\log_2 K$ | $H(\boldsymbol{B})$ | | $H(\boldsymbol{B'})$ | $\geqslant$ | $H(A')$ |

Note that the particular code $c : \mathcal{A} \to \mathcal{B}^N$ does not really matter. We can permute the code words $\{c(a) : a \in \mathcal{A}\}$ in any way we wish and not alter the rate of tranmission or the maximum error probability. So, the set of code words is really more important than the actual alphabet or code we choose. We sometimes refer to the set of code words $\{c(a) : a \in \mathcal{A}\}$ as the *code book*.

## 9.2 Fano's Inequality

**Lemma 9.1**
*Let $X$ be a random variable that takes values in a finite alphabet $\mathcal{A}$ of size $K$. Then, for any fixed letter $a \in \mathcal{A}$, we have*

$$H(X) \leqslant p \log_2(K-1) + h(p)$$

*where $p = \mathbb{P}(X \neq a)$.*

*Proof:*

Recall that

$$H(X, I) = H(X|I) + H(I) \ .$$

Let $I$ be the indicator random variable

$$I = \begin{cases} 1 & \text{when } X \neq a; \\ 0 & \text{when } X = a. \end{cases}$$

Then we see that:

$$H(X, I) = H(X)$$

because $I$ is determined by the values of $X$.

$$\begin{aligned} H(X|I) &= \mathbb{P}(I=0)H(X|I=0) + \mathbb{P}(I=1)H(X|I=1) \\ &= (1-p)0 + pH(X|I=1) \\ &\leqslant p \log_2(K-1) \end{aligned}$$

because, if $I = 0$ then $X = a$ and so $H(X|I=0) = 0$, while if $I = 1$, there are only $K-1$ possible values for $X$.

Therefore, we have

$$\begin{aligned} H(X) &= H(X|I) + H(I) \\ &\leqslant p \log_2(K-1) + h(p) \end{aligned}$$

because $H(I) = H(1-p, p) = h(p)$. □

**Theorem 9.2** Fano's inequality
*Let $X, Y$ be two random variables that take values in a finite alphabet $\mathcal{A}$ of size $K$. Then*

$$H(X|Y) \leqslant p \log_2(K-1) + h(p)$$

*where $p = \mathbb{P}(X \neq Y)$.*

We will apply this where $X$ takes values in the alphabet $\mathcal{A}$ and $Y$ is the result of passing the code word $c(X)$ through a channel and decoding it. The probability $p$ is then the probability of error.

*Proof:*

Condition the inequality in the Lemma on the value of $Y$ to obtain

$$H(X|Y=y) \leqslant \mathbb{P}(X \neq y|Y=y) \log_2(K-1) + H(I|Y=y)$$

where $I$ is given by

$$I = \begin{cases} 1 & \text{when } X \neq Y; \\ 0 & \text{when } X = Y. \end{cases}$$

If we multiply this by $\mathbb{P}(Y=y)$ and sum over all values for $y$ we obtain

$$H(X|Y) \leqslant \mathbb{P}(X \neq Y) \log_2(K-1) + H(I|Y) = p \log_2(K-1) + H(I|Y) \ .$$

Finally, recall that $H(I|Y) = H(I, Y) - H(Y) \leqslant H(I)$ by Corollary 2.2 . □

We can think of the two terms of Fano's inequality as $h(p) = H(I)$ measuring the information given by knowing whether there is an error and $p \log_2(K-1)$ measuring the information on what that error is, when it occurs.

We can use Fano's inequality to compare the rate of transmission of information to the capacity of the channel. As always we will use our model for a noisy channel. A random letter $A$ is chosen from $\mathcal{A}$ and encoded as the $\mathcal{B}^N$-valued random variable $\boldsymbol{B} = c(A)$. This is transmitted, one letter at a time, through the noisy channel and received as the $\mathcal{B}^N$-valued random variable $\boldsymbol{B}'$ which is decoded to give $A'$. The rate of transmission is $\rho(c) = \log_2 K/N$.

Choose $A$ so that each letter in $\mathcal{A}$ has probability $1/K$. Fano's inequality gives

$$H(A|A') \leqslant h(p) + p \log_2(K-1) < h(p) + p \log_2 K \ .$$

So we have

$$I(A', A) = H(A) - H(A|A') \geqslant \log_2 K - (h(p) + p \log_2 K) \ .$$

Now the code $c : \mathcal{A} \to \mathcal{B}^N$ is injective, so $A$ is a function of $\boldsymbol{B}$. Therefore, Proposition 8.4 show that

$$I(A', A) \leqslant I(A', \boldsymbol{B}) \ .$$

Similarly, $A'$ is a function of $\boldsymbol{B}'$ — the decoding function. So

$$I(A', \boldsymbol{B}) \leqslant I(\boldsymbol{B}', \boldsymbol{B}) \ .$$

Therefore, $I(A', A) \leqslant I(\boldsymbol{B}', \boldsymbol{B}) \leqslant N\text{Cap}$ and hence

$$N\text{Cap} \geqslant (1-p) \log_2 K - h(p) \ .$$

Dividing by $N$ we obtain

$$\text{Cap}(Q) + \frac{h(p)}{N} \geqslant (1-p)\rho(c) \ .$$

**Theorem 9.3** Capacity bounds the rate of transmission of information
*Let $c : \mathcal{A} \to \mathcal{B}^N$ where each letter of a code word is transmitted through a time-independent memoryless channel with capacity* Cap. *Let $p$ be the probability of decoding incorrectly. Then the rate of transmission satisfies*

$$\rho(c) \leqslant \frac{\log_2 K}{N} \leqslant \frac{\text{Cap}}{1-p} + \frac{h(p)}{N(1-p)} \ .$$

□

As we let the probability of error decrease to 0, so $1 - p \nearrow 1$ and $h(p) \searrow 0$. Therefore

$$\frac{\text{Cap}}{1-p} + \frac{h(p)}{N(1-p)} \searrow \text{Cap} \ .$$

Hence, if we find codes $c_j$ with probabilities of error $p_j$ that decrease to 0, and rates of tranmission that tend to a limit $\rho$, then $\rho$ must be at most the information capacity of the channel.

## 10: SHANNON'S NOISY CODING THEOREM

### 10.1 Introduction

In this section we will prove Shannon's Noisy Coding Theorem, which shows that we can find codes with rate of tranmission close to the capacity of a channel and small probability of error. We will do this only in the case of a binary symmetric channel. The arguments are simpler in this case but the principles apply more generally.

**Theorem 10.1** Shannon's Noisy Coding Theorem
*For $\varepsilon > 0$ and a binary symmetric channel with capacity* Cap *there are codes*

$$c_N : \mathcal{A}_N \to \{0,1\}^N$$

*for $N$ sufficiently large with rate of tranmission*

$$\rho(c_N) = \frac{\log_2 |\mathcal{A}_N|}{N} \geqslant \text{Cap} - \varepsilon$$

*and maximum error*

$$\widehat{e}(c_N) < \varepsilon \ .$$

---

**Example:**
We wish to send a message $m$ from an alphabet $\mathcal{A}$ of size $K$ through a binary symmetric channel with error probability $p = 0.1$. What rate of transmission can we achieve with small probability of error?

Cap $= 1 - h(0.1) = 0.53$, so Shannon's Theorem shows that we can achieve any rate of tranmission strictly less than this with arbitrarily small probability of error. For example, suppose that we want a rate of tranmission $\rho \geqslant 0.5$ and $\widehat{e} < 0.01$. Shannon's Theorem shows that there are codes $c_N : \mathcal{A}_N \to \{0,1\}^N$ that achieve this provided that $N$ is large enough, say $N \geqslant N_o$.

Suppose that we know such a code $c_N$. How do we encode the message $m$? First divide $m$ into blocks of length $B$ where

$$B = \left\lfloor \frac{0.5N}{\log_2 K} \right\rfloor \qquad \text{so that} \qquad |\mathcal{A}^B| = K^B \leqslant 2^{0.5N} \ .$$

Then we can embed the blocks from $\mathcal{A}^B$ in the alphabet $\mathcal{A}_N$ and so encode the blocks. The rate of tranmission is

$$\frac{\log_2 |\mathcal{A}^B|}{N} \sim 0.5 \ .$$

Unfortunately, Shannon's Theorem tells us that there are such codes but does not tell us how to find them and it is very difficult to do so.

---

### 10.2 Chebyshev's Inequality

**Theorem 10.2** Chebyshev's inequality
*Let $X$ be a real-valued, discrete random variable with mean $\mathbb{E}X$ and variance* var$(X)$*. Then*

$$\mathbb{P}\left(|X - \mathbb{E}X| \geqslant t\right) \leqslant \frac{\text{var}(X)}{t^2} \ .$$

(The inequality is true whenever the variance of $X$ exists, even if $X$ is not discrete.)

*Proof:*

We have

$$\begin{aligned}\mathrm{var}(X) = \mathbb{E}|X - \mathbb{E}X|^2 &= \sum \mathbb{P}(X = x)|x - \mathbb{E}X|^2 \\ &\geqslant \sum \left\{\mathbb{P}(X = x)t^2 : |x - \mathbb{E}X| \geqslant t\right\} = t^2\mathbb{P}(|X - \mathbb{E}X| \geqslant t) \ .\end{aligned}$$

So we obtain the result. □

### 10.3 Proof of the Noisy Coding Theorem

To prove Shannon's Theorem ( Theorem 10.1 ) we need to find code words in $\{0,1\}^N$. Choose these at random and independently of one another and of the channel. We will prove that, on average, this gives the inequalities we want. Then there must be at least some of the choices for code words that also give the inequalities.

Let the binary symmetric channel have error probability $p$ with $0 \leqslant p < \frac{1}{2}$. Then $\mathrm{Cap} = 1 - h(p)$. Set

$$K_N = \left\lfloor 2^{N(\mathrm{Cap}-\varepsilon)} \right\rfloor \ \ll \ 2^N$$

and let $\mathcal{A}_N$ be an alphabet with $K_N$ letters.

Choose a letter letter $a_o \in \mathcal{A}_N$ and then choose a random code word as $c_N(a_o)$ uniformly from the $2^N$ words in $\{0,1\}^N$. These choices must be independent for each different letter and also independent of the channel. When a fixed word $\boldsymbol{c}_o = c_N(a_o)$ is sent through the channel, it is corrupted and a new word $\boldsymbol{c}'_o$ is received. We want this received word to be close to $\boldsymbol{c}_o$.

The Hamming distance $d(\boldsymbol{c}, \boldsymbol{c}')$ is a Bernoulli $B(N,p)$ random variable, so it has mean $Np$ and variance $Np(1-p)$. Therefore Chebyshev's inequality gives

$$\mathbb{P}(d(\boldsymbol{c}_o, \boldsymbol{c}'_o) \geqslant r) \leqslant \frac{Np(1-p)}{(r - Np)^2}$$

for $r > Np$. Take $q > p$ and set $r = Nq$, then

$$\mathbb{P}(d(\boldsymbol{c}_o, \boldsymbol{c}'_o) \geqslant r) \leqslant \frac{p(1-p)}{N(q-p)^2} \qquad (1).$$

Now consider another code word $\boldsymbol{c} = c_N(a)$ for a letter $a \neq a_o$. This is chosen uniformly and randomly from the $2^N$ words in $\{0,1\}^N$, so Proposition 7.8 shows that

$$\mathbb{P}(d(\boldsymbol{c}, \boldsymbol{c}'_o) < r) = \mathbb{P}(\boldsymbol{c} \in B(\boldsymbol{c}'_o, r)) = \frac{V(N,r)}{2^N} \leqslant \frac{2^{Nh(q)}}{2^N} = 2^{-N(1-h(q))} \ .$$

The probability that $d(\boldsymbol{c}, \boldsymbol{c}'_o) < r$ for at least one of the $K_N - 1$ code words $\boldsymbol{c}$ not equal to $\boldsymbol{c}_o$ is therefore at most

$$K_N 2^{-N(1-h(q))} = 2^{N(\mathrm{Cap}-\varepsilon-(1-h(q)))} \ .$$

Now $\mathrm{Cap} = 1 - h(p)$, so this gives

$$\mathbb{P}(\text{ there exists } a \neq a_o \text{ with } d(c_N(a), \boldsymbol{c}'_o) < r) \leqslant 2^{N(h(q)-h(p)-\varepsilon)} \qquad (2).$$

We will use the minimum distance decoding rule. So we make an error and decode $\boldsymbol{c}_o = c_N(a_o)$ as another letter $a \neq a_o$ only when either $d(\boldsymbol{c}_o, \boldsymbol{c}'_o) \geqslant r$ or $d(c_N(a), \boldsymbol{c}'_o) < r$ for some $a \neq a_o$. Hence the probability of an error is

$$\mathbb{P}(\text{ error }) \leqslant \frac{p(1-p)}{N(q-p)^2} \;+\; 2^{N(h(q)-h(p)-\varepsilon)}$$

because of (1) and (2). Choose $q$ with

$$p < q < \tfrac{1}{2} \qquad \text{and} \qquad h(q) < h(p) + \varepsilon \ .$$

This is certainly possible for $0 \leqslant p < \frac{1}{2}$. Then

$$\frac{p(1-p)}{N(q-p)^2} \leqslant \tfrac{1}{2}\varepsilon \qquad \text{and} \qquad 2^{N(h(q)-h(p)-\varepsilon)} \leqslant \tfrac{1}{2}\varepsilon$$

for all sufficiently large $N$. For such $N$ we see that the probability of error is strictly less than $\varepsilon$.

This is the error averaged over all of the random choices of code words. This means that there must be at least one choice of the code words that also has $\mathbb{P}(\text{ error }) < \varepsilon$. Take this as the code $c_N$.

We have almost proved the result we sought except that we have proved that the mean error $\mathbb{P}(\text{ error })$ rather than that the maximum error $\widehat{e}(c_N) = \max\{\mathbb{P}(\text{ error } \mid a \text{ sent }) : a \in \mathcal{A}\}$ is less than $\varepsilon$.

For our code $c_N$ the average error probability satisfies

$$\frac{1}{|\mathcal{A}_N|} \sum_{a \in \mathcal{A}_N} \mathbb{P}(\text{ error } \mid a \text{ sent }) \;< \varepsilon$$

so, for at least half of the letters in $\mathcal{A}_N$ we must have

$$\mathbb{P}(\text{ error } \mid a \text{ sent }) < 2\varepsilon \ .$$

Choose a new alphabet $\mathcal{A}'_N$ that consists just of these $\frac{1}{2}K_N$ letters. Then we certainly have

$$\mathbb{P}(\text{ error } \mid a \text{ sent }) < 2\varepsilon \qquad \text{for } a \in \mathcal{A}'_N$$

so the maximum error is at most $2\varepsilon$. Furthermore,

$$|\mathcal{A}'_N| = \tfrac{1}{2}|\mathcal{A}_N| = 2^{N(\mathrm{Cap}-\varepsilon)-1},$$

so the rate of transmission is

$$\frac{\log_2 |\mathcal{A}'_N|}{N} = \mathrm{Cap} - \varepsilon - \frac{1}{N} \ .$$

□

### 10.4 * Typical Sequences *

In this section we want to reinterpret the entropy $H(A)$ in terms of the probability of a "typical sequence" of values arising from a sequence of independent random variables $(A_n)$ all with the same distribution as $A$.

As a motivating example, consider tossing an unfair coin with probability $p$ of heads. Let $A, A_n$ be the results of independent coin tosses. If we toss this coin a large number $N$ times, we expect to get approximately $pN$ heads and $(1-p)N$ tails. The probability of any particular sequence of $pN$ heads and $(1-p)N$ tails is

$$p^{pN}(1-p)^{(1-p)N} = 2^{N(p\log_2 p+(1-p)\log_2(1-p))} = 2^{-NH(A)} \ .$$

Of course not every sequence of coin tosses will be like this. It is quite possible to get extraordinary sequences like all $N$ heads. However, there is only a small probability of getting such an atypical sequence. With high probability we will get a typical sequence and its probability will be close to $2^{-NH(A)}$.

To make this precise we need to recall the weak law of large numbers from the probability course. This follows from Chebyshev's inequality.

We say that a sequence of random variables $S_n$ *converge to a random variable* $L$ *in probability* if

$$\mathbb{P}\left(|S_n - L| \geqslant \varepsilon\right) \to 0 \qquad \text{as} \qquad n \to \infty$$

for every $\varepsilon > 0$. This means that $S_n$ and $L$ can still take very different values for large $n$ but only on a set with small probability.

**Theorem 10.3** The weak law of large numbers
*Let $X$ be a discrete real-valued random variable. Let $(X_n)$ be independent random variables each with the same distribution as $X$. The averages:*

$$S_n = \frac{X_1 + X_2 + \ldots + X_n}{n}$$

*then converge in probability to the constant $\mathbb{E}X$ as $n \to \infty$.*

*Proof:*
We need to show that

$$\mathbb{P}\left(|S_n - \mathbb{E}X| \geqslant t\right) \to 0 \qquad \text{as} \qquad n \to \infty \ .$$

Note that

$$\mathbb{E}S_n = \frac{\mathbb{E}X_1 + \mathbb{E}X_2 + \ldots + \mathbb{E}X_n}{n} = \mathbb{E}X$$

and

$$\mathrm{var}(S_n) = \frac{\mathrm{var}(X_1) + \mathrm{var}(X_2) + \ldots + \mathrm{var}(X_n)}{n^2} = \frac{\mathrm{var}(X)}{n} \ .$$

Chebyshev's inequality gives

$$\mathbb{P}\left(|S_n - \mathbb{E}S_n| \geqslant t\right) \leqslant \frac{\mathrm{var}(S_n)}{t^2}$$

so we have

$$\mathbb{P}\left(|S_n - \mathbb{E}X| \geqslant t\right) \leqslant \frac{\mathrm{var}\, X}{nt^2}$$

which certainly tends to 0 as $n \to \infty$. □

Let $A$ be a random variable that takes values in the finite set $\mathcal{A}$. Let $(A_n)$ be a sequence of independent random variables each with the same distribution as $A$. Recall that the entropy of $A$ is

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

This is the expectation of the real-valued random variable:

$$X(\omega) = -\log_2 \mathbb{P}(A = A(\omega))\ .$$

(In other words, if $A$ takes the values $a_k$ with probabilities $p_k$, then $X$ takes the values $-\log_2 p_k$ with the same probabilities.) In a similar way we define $X_n$ from $A_n$, so the random variables $X_n$ are independent and each has the same distribution as $X$.

The weak law of large numbers Theorem 10.3 shows that

$$\frac{X_1 + X_2 + \ldots X_n}{n} \to \mathbb{E}X \qquad \text{in probability as } n\to\infty\ .$$

Hence, for any $\varepsilon > 0$, there is an $N(\varepsilon)$ with

$$\mathbb{P}\left(\left|\frac{X_1 + X_2 + \ldots X_n}{n} - \mathbb{E}X\right| \geqslant \varepsilon\right) \leqslant \varepsilon$$

for $n \geqslant N(\varepsilon)$.

Consider a particular sequence of values $(a_j)_{j=1}^n$ from $\mathcal{A}$. The probability of obtaining this sequence as $(A_j)$ is

$$\mathbb{P}\left(A_j = a_j \text{ for } j = 1,2,\ldots,n\right) = \prod_{j=1}^n \mathbb{P}(A=a_j)\ .$$

Also, if we do have $A_j = a_j$, then

$$\frac{X_1 + X_2 + \ldots X_n}{n} = -\sum_{j=1}^n \log_2 \mathbb{P}(A=a_j) = -\log_2 \prod_{j=1}^n \mathbb{P}(A=a_j)$$

and therefore,

$$\left|\frac{X_1 + X_2 + \ldots X_n}{n} - \mathbb{E}X\right| < \varepsilon$$

is equivalent to

$$2^{-n(H(A)+\varepsilon)} < \mathbb{P}\left(A_j = a_j \text{ for } j = 1,2,\ldots,n\right) < 2^{-n(H(A)-\varepsilon)}\ . \qquad *$$

Thus we see that the probability of obtaining a sequence $(a_j)$ for which $(*)$ holds is at least $1-\varepsilon$ for $n \geqslant N(\varepsilon)$. These are the "typical sequences". Thus we have proved:

**Theorem 10.4** Asymptotic equipartition property
*Let $A$ be a random variable taking values in the finite set $\mathcal{A}$ and with entropy $H(A)$. Let $(A_n)$ be a sequence of independent random variables each with the same distribution as $A$. For each $\varepsilon > 0$, there is a natural number $N(\varepsilon)$ such that the following conditions hold for any $n \geqslant N(\varepsilon)$.*

*There is a set $T$ of sequences in $\mathcal{A}^n$ with*

$$\mathbb{P}((A_j)_{j=1}^n \in T) > 1-\varepsilon$$

*and, for each sequence $(a_j) \in T$,*

$$2^{-n(H(A)+\varepsilon)} < \mathbb{P}\left(A_j = a_j \text{ for } j = 1,2,\ldots,n\right) < 2^{-n(H(A)-\varepsilon)}\ . \qquad *$$

We call the sequences in $T$ the "typical ones" while those outside are "atypical". The sequences in $T$ all have approximately the same probability $2^{-nH(A)}$ of arising, so we say that the sequence of random variable $(A_n)$ has the *asymptotic equipartition property.*

## 11: LINEAR CODES

It is useful to use the algebra you have learnt in earlier courses to describe codes. In particular, if we are dealing with an alphabet of size $q = p^k$ for some prime number $p$, then there is a field $\mathbb{F}_q$ with exactly $q$ elements and we can use this to represent the alphabet. The most important case is when $q = 2$, or a power of 2, and we have seen this already when we looked at Hamming's code.

### 11.1 Finite Fields

In this section we will recall results from the Numbers and Sets, Linear Algebra, and Groups, Rings and Modules courses. For this course, it will be sufficient to use only the integers modulo 2, $\mathbb{F}_2$ or occasionally the fields $\mathbb{F}_{2^K}$ for $K > 1$.

There are fields with a finite number $q$ of elements if and only if $q$ is a prime power $q = p^r$. These fields are unique, up to isomorphism, and will be denoted by $\mathbb{F}_q$.

---

**Example:**
For each prime number $p$, the integers modulo $p$ form a field. So $\mathbb{F}_p = \mathbb{Z}/p\mathbb{Z}$.

---

The non-zero elements of a finite field $\mathbb{F}_q$ form a commutative group denoted by $\mathbb{F}_q^\times$. This is a cyclic group of order $q-1$. Any generator of $\mathbb{F}_q^\times$ is called a *primitive element* for $\mathbb{F}_q$. Let $\alpha$ be a primitive element. Then the other elements of the group $\mathbb{F}_q^\times$ are the powers $\alpha^k$ for $k = 1, 2, 3, \ldots, q-1$. These are all distinct. The order of the element $\alpha^k$ is $\dfrac{q-1}{(q-1,k)}$, so the number of primitive elements is given by Euler's totient function

$$\varphi(q-1) = |\{k : k = 1, 2, 3, \ldots, q-1 \text{ and } (q-1, k) = 1\}| \ .$$

---

**Example:**
For the integers modulo 7 we have 3 and 5 as the only primitive elements.

---

We will want to work with alphabets that are finite dimensional vector spaces over the finite fields $\mathbb{F}_q$. For example, $\mathbb{F}_q^N$ is an $N$ dimensional vector space over $\mathbb{F}_q$. Any $N$-dimensional vector space has $q^N$ vectors. The *scalar product* on $\mathbb{F}_q^N$ is given by

$$\boldsymbol{x} \cdot \boldsymbol{y} = \sum_{n=1}^{N} x_n y_n \ .$$

(Every linear map $A : \mathbb{F}_q^N \to \mathbb{F}_q$ is of the form $\boldsymbol{x} \mapsto \boldsymbol{x} \cdot \boldsymbol{y}$ for some $\boldsymbol{y} \in \mathbb{F}_q^N$. So the scalar product gives us a way to identify the dual space $(\mathbb{F}_q^N)^*$ with $\mathbb{F}_q^N$.) For any finite set $S$, the set of all functions $V = \mathbb{F}_q^S = \{f : S \to \mathbb{F}_q\}$ is a vector space over $\mathbb{F}_q$ with dimension $|S|$.

A *polynomial* over $\mathbb{F}_q$ is a formal expression:

$$A(X) = a_0 + a_1 X + a_2 X^2 + \ldots + a_D X^D$$

with the coefficients $a_n \in \mathbb{F}_q$. In this expression, $X$ is an indeterminate or formal parameter rather than an element of the field. The *degree* $\deg(A)$ of the polynomial $A$ is the largest $n$ with $a_n \neq 0$. The degree of the 0 polynomial is defined to be $-\infty$. Define addition and multiplication of polynomials as usual. This makes the set $\mathbb{F}_q[X]$ of all polynomials over $\mathbb{F}_q$ a ring. The degree is an Euclidean function for this ring and makes it an Euclidean domain: For polynomials $A, D \in \mathbb{F}_q[X]$ with $D \neq 0$, there are uniquely determined polynomials $Q$ and $R$ with

$$A = Q.D + R \qquad \text{and} \qquad \deg(R) < \deg(D) \ .$$

This has many useful consequences. We have the division criterion:

$$(X-\alpha)|A \qquad \Leftrightarrow \qquad A(\alpha)=0$$

for an element $\alpha \in \mathbb{F}_q$. Furthermore, the polynomial ring is a principal ideal domain. For, if $I$ is an ideal in $\mathbb{F}_q[X]$ (written $I \lhd \mathbb{F}_q[X]$), then either $I=\{0\}$ or there is a non-zero polynomial $D \in I$ of smallest degree. For any other polynomial $A \in I$ we can write

$$A = Q.D + R \qquad \text{with} \qquad \deg(R) < \deg(D)\ .$$

However, $R \in I$, so $R$ must be the 0 polynomial. Therefore $A = Q.D$. Hence every ideal is of the form $D\mathbb{F}_q[X]$ for some polynomial $D$. We will denote this ideal by $(D)$.

For any ideal $I = D\mathbb{F}_q[X]$, the quotient $\mathbb{F}_q[X]/I$ is a ring and there is a quotient ring homomorphism $\mathbb{F}_q[X] \to \mathbb{F}_q[X]/I$. The quotient is a vector space over $\mathbb{F}_q$ with dimension $\deg(D)$.

A very important example of this is when $D$ is the polynomial $X^n-1$ and we consider the quotient $\mathbb{F}_q[X]/(X^n-1)$. If we divide any polynomial $A \in \mathbb{F}_q[X]$ by $X^n-1$ we obtain a remainder of degree strictly less than $n$. So each coset $A+(X^n-1)\mathbb{F}_q[X]$ contains an unique polynomial $R$ of degree at most $n-1$. Hence we can represent the quotient $\mathbb{F}_q[X]/(X^n-1)$ by

$$\{a_0 + a_1X + \ldots + a_{n-1}X^n : a_0, a_1, \ldots, a_{n-1} \in \mathbb{F}_q\}\ .$$

Multiplication in the quotient $\mathbb{F}_q[X]/(X^n-1)$ then corresponds to multiplying the polynomials and reducing the result modulo $X^n-1$, that is replacing any power $X^k$ for $k \geqslant n$ by $X^m$ where $k \equiv m \pmod{n}$. Note that, in this case, the quotient $\mathbb{F}_q[X]/(X^n-1)$ is a vector space of degree $n$.

## 11.2 Linear Codes

Suppose that we are looking at codes $c : \mathcal{A} \to \mathcal{B}^N$. If both of the alphabets $\mathcal{A}$ and $\mathcal{B}^N$ have orders that are powers of the same prime power $q$, then we can represent both as vector spaces over $\mathbb{F}_q$. So $c$ is a map $\mathbb{F}_q^K \to \mathbb{F}_q^N$ for suitable dimensions $K, N$. Such a map is much easier to specify if it is linear, for then we need only give the images of a basis for $\mathbb{F}_q^K$. This is in itself a considerable advantage and makes it much simpler to implement the encoding and decoding of messages.

We will write $\mathbb{F}$ for any finite field, so $\mathbb{F} = \mathbb{F}_q$ for some prime power $q$. A code is *linear* if it is an injective linear map $c : \mathbb{F}^K \to \mathbb{F}^N$. The image of $c$ is then a vector subspace of $\mathbb{F}^N$, called the *code book* of $c$. Its dimension is the rank of $c$.

---

**Example:**
Hamming's code was a linear map $\mathbb{F}_2^4 \to \mathbb{F}_2^7$.

---

A linear code $c : \mathbb{F}^K \to \mathbb{F}^N$ is given by an $N \times K$ matrix $C$ with entries in $\mathbb{F}$. So $c : \boldsymbol{x} \mapsto C\boldsymbol{x}$. Since $c$ is injective, the matrix $C$ has nullity 0 and rank $K$. As usual, we are really only interested in the code book. This can be specified by giving any basis for the image.

The columns of the matrix $C$ are one basis of code words for the code book. We can apply column operations to these to reduce the matrix to echelon form. For example

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

where $*$ denotes an arbitrary but unimportant value. Applying column operations to this we can obtain a matrix:

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

The columns of this matrix are still a basis for the code book. If we re-order the rows of this matrix, which corresponds to re-ordering the components of $\mathbb{F}^N$, then we obtain a matrix of the form

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

In general, when we apply these operations to our code matrix $C$ we change it to the form

$$C = \begin{pmatrix} I \\ A \end{pmatrix}$$

where $I$ is the $K \times K$ identity matrix and $A$ is an arbitrary $(N-K) \times K$ matrix. Whenever we need to we can reduce our linear code to this standard form. When we have a matrix of this form, the code is

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_K \end{pmatrix} \mapsto \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_K \\ \sum a_{1j}x_j \\ \vdots \\ \sum a_{(N-K)j}x_j \end{pmatrix} \qquad \text{or, more concisely,} \qquad \boldsymbol{x} \mapsto \begin{pmatrix} \boldsymbol{x} \\ A\boldsymbol{x} \end{pmatrix} .$$

So the first $K$ terms carry the information about the vector $\boldsymbol{x}$ while the remainder are check digits.

### 11.3 The Syndrome

Let $c : \mathbb{F}^K \to \mathbb{F}^N$ be a linear code. Its image is a vector subspace of $\mathbb{F}^N$ with dimension $K$. So we can construct a linear map

$$s : \mathbb{F}^N \to \mathbb{F}^{N-K} \qquad \text{with } \ker(s) = \mathrm{Im}(c) .$$

Such a linear map is called a *syndrome of c*.

For example, if we write the matrix $C$ for $c$ in the standard form $C = \begin{pmatrix} I \\ A \end{pmatrix}$, then the matrix $S = \begin{pmatrix} -A & I \end{pmatrix}$ satisfies

$$\ker(S) = \left\{ \begin{pmatrix} \boldsymbol{u} \\ \boldsymbol{v} \end{pmatrix} : -A\boldsymbol{u} + \boldsymbol{v} = \boldsymbol{0} \right\} = \left\{ \begin{pmatrix} \boldsymbol{u} \\ A\boldsymbol{u} \end{pmatrix} : \boldsymbol{u} \in \mathbb{F}^K \right\} = \mathrm{Im}(C) .$$

So $S$ is a syndrome for the code.

The syndrome gives an easy way to check whether a vector $\boldsymbol{x} \in \mathbb{F}^N$ is a code word, for we simply compute $S\boldsymbol{x}$ and see whether it is $\mathbf{0}$. We can also define a code by giving a syndrome, for the code book is then $\ker(s)$.

---

**Exercise:**
For a natural number $d$ there are $N = 2^d - 1$ non-zero vectors in $\mathbb{F}_2^d$. Take these as the columns of an $d \times N$ matrix $S$. The kernel of $S$ is then the code book for a Hamming code

$$c : \mathbb{F}_2^{N-d} \to \mathbb{F}_2^N \ .$$

Check that the case $d = 3$ gives the Hamming code we constructed earlier.

Show that each of these Hamming codes is a perfect 1-error correcting code.

---

(The code $c : \mathbb{F}^K \to \mathbb{F}^N$ and the syndrome $s : \mathbb{F}^N \to \mathbb{F}^{N-K}$ are really dual to one another. For the dual maps are

$$s^* : \mathbb{F}^{N-K} \to \mathbb{F}^N \qquad \text{and} \qquad c^* : \mathbb{F}^N \to \mathbb{F}^K$$

with $\ker(c^*) = \operatorname{Im} c^\perp = \ker(s)^\perp = \operatorname{Im}(s^*)$. So $s^*$ is a linear code with $c^*$ as its syndrome. These correspond to transposing the matrices. So $S^\top$ is the matrix for a linear code with $C^\top$ its syndrome.)

The minimum distance for a code is the minimum Hamming distance between two different code words. When the code is linear we have $d(\boldsymbol{c}', \boldsymbol{c}) = d(\mathbf{0}, \boldsymbol{c} - \boldsymbol{c}')$. So the minimum distance is

$$\min \{ d(\mathbf{0}, \boldsymbol{c}) : \boldsymbol{c} \text{ is a non-zero code word } \} \ .$$

The *weight of a code word* $\boldsymbol{x}$ is the Hamming distance of $\boldsymbol{x}$ from $\mathbf{0}$. So the minimum distance for a linear code is the minimum weight for non-zero code words.

We will use minimum distance decoding. Suppose that we receive a word $\boldsymbol{x} \in \mathbb{F}^N$. We compute its syndrome $s(\boldsymbol{x})$. If this is $\mathbf{0}$ then we know that $\boldsymbol{x}$ is a code word. Otherwise there have been errors in transmission and we need to find the code word closest to $\boldsymbol{x}$.

We proceed as follows. For each vector $\boldsymbol{y} \in \mathbb{F}^{N-K}$, find a vector $\boldsymbol{z} \in s^{-1}(\boldsymbol{y})$ with minimum weight and call this $u(\boldsymbol{y})$. Note that we can do this for every vector $\boldsymbol{y} \in \mathbb{F}^{N-K}$ in advance of receiving any message, at least when $\mathbb{F}^{N-K}$ is not too large. It is clear that $s(u(\boldsymbol{y})) = \boldsymbol{y}$.

Now we decode the received vector $\boldsymbol{x}$ as $\boldsymbol{c}_o = \boldsymbol{x} - u(s(\boldsymbol{x}))$. Since

$$s(\boldsymbol{x} - u(s(\boldsymbol{x}))) = s(\boldsymbol{x}) - s(u(s(\boldsymbol{x}))) = \mathbf{0}$$

we see that $\boldsymbol{c}_o$ is indeed a code word. For any other code word $\boldsymbol{c}$, we have $d(\boldsymbol{c}, \boldsymbol{x}) = d(\mathbf{0}, \boldsymbol{x} - \boldsymbol{c})$. However, $s(\boldsymbol{x} - \boldsymbol{c}) = s(\boldsymbol{x})$, so the definition of $u$ ensures that

$$d(\mathbf{0}, \boldsymbol{x} - \boldsymbol{c}) \geqslant d(\mathbf{0}, u(s(\boldsymbol{x}))) = d(\mathbf{0}, \boldsymbol{x} - \boldsymbol{c}_o)$$

and hence that

$$d(\boldsymbol{c}, \boldsymbol{x}) \geqslant d(\boldsymbol{c}_o, \boldsymbol{x}) \ .$$

So $\boldsymbol{c}_o$ is the code word closest to $\boldsymbol{x}$.

## 11.4 Reed – Muller Codes

In this section we will construct linear codes that have proved useful in dealing with very noisy channels.

Let $S$ be the finite dimensional vector space $\mathbb{F}_2^M$ over the field $\mathbb{F}_2$ of integers modulo 2. This is itself a finite set with $2^M$ elements. Let $V$ be the set of all functions from $S$ to $\mathbb{F}_2$:

$$V = \mathbb{F}_2^S = \{f : S \to \mathbb{F}_2\} \ .$$

The functions

$$1_{\boldsymbol{y}}(\boldsymbol{x}) = \begin{cases} 1 & \text{when } \boldsymbol{x} = \boldsymbol{y}; \\ 0 & \text{otherwise} \end{cases}$$

are the standard basis for $V$, so $V$ has dimension $2^M$. We will use this basis to determine the Hamming distance in $V$, so it is

$$d(f, g) = |\{\boldsymbol{x} \in S : f(\boldsymbol{x}) \neq g(\boldsymbol{x})\}| \ .$$

Let $\pi_i$ be the function $\boldsymbol{x} \mapsto x_i$ that maps each vector in $S$ to its $i$th component. Let $I$ be a subset $\{i(1), i(2), \ldots, i(r)\}$ of $\{1, 2, 3, \ldots, M\}$ and write $\pi_I$ as an abbreviation for the function

$$\boldsymbol{x} \mapsto \pi_{i(1)}(\boldsymbol{x})\pi_{i(2)}(\boldsymbol{x}) \ldots \pi_{i(r)}(\boldsymbol{x}) = \prod_{i \in I} x_i \ .$$

This is a function in $V$. In particular the empty sequence $\emptyset$ has $\pi_\emptyset$ equal to the constant function 1.

**Lemma 11.1**
*The functions $\pi_I$ for $I \subset \{1, 2, 3, \ldots, M\}$ are a basis for $V$.*

*Proof:*

We can write the value of $1_{\boldsymbol{y}}(\boldsymbol{x})$ as the product

$$1_{\boldsymbol{y}}(\boldsymbol{x}) = \prod_{i=1}^{M} (x_i - y_i + 1) \ .$$

Now expand this product to get

$$1_{\boldsymbol{y}}(\boldsymbol{x}) = \sum_I \alpha_I \left( \prod_{i \in I} x_i \right)$$

for some scalar coefficients $\alpha_I \in \mathbb{F}_2$ that depend on $\boldsymbol{y}$. This shows that $1_{\boldsymbol{y}}$ is the linear combination $1_{\boldsymbol{y}} = \sum_I \alpha_I \pi_I$ of the functions $x_I$.

Since $(1_{\boldsymbol{y}})$ is a basis for $V$, we see that the functions $\pi_I$ span $V$. There are $2^M = \dim V$ of the $\pi_I$, so they must form a basis for $V$. □

The lemma shows that we can write any function $f \in V$ as a linear combination

$$f = \sum_I a_I \pi_I \ .$$

The *Reed – Muller code* $RM(M, r)$ has a code book equal to

$$\left\{ \sum (\alpha_I \pi_I : |I| \leqslant r) : \alpha_I \in \mathbb{F}_2 \right\} \ .$$

This code book is a vector subspace with dimension

$$\binom{M}{0} + \binom{M}{1} + \ldots \binom{M}{r}$$

so it gives a linear code of this rank and length $\dim V = 2^M$.

---

**Example:**
The $RM(M,0)$ code has only two code words 0 and 1 so it is the repetition code that repeats each bit $2^M$ times. The Hamming distance $d(0,1)$ is $2^M$, so this is also the minimum distance for $RM(M,0)$.

$RM(M,1)$ has dimension $1+M$ and has the functions $\pi_j$ for $j = 1,2,3,\ldots,M$ as a basis. The Hamming distance satisfies

$$d(0,\pi_j) = 2^{M-1}$$

and it is easy to see that this is the minimum distance for $RM(M,1)$.

---

**Proposition 11.2** Reed – Muller codes
*The Reed – Muller code $RM(M,r)$ has minimum distance $2^{M-r}$ for $0 \leqslant r \leqslant M$.*

*Proof:*

We need to show that the minimum weight of a non-zero code word in $RM(M,r)$ is $2^{M-r}$. We will do this by induction on $M$. We already know the result for $M = 1$.

Suppose that we have already established the result for $M-1$ and all $r$.

Note first that, for the set $J = \{1,2,\ldots,r\}$ we have $d(0,\pi_J) = 2^{M-r}$ so the minimum distance is at most this large. Let $f = \sum_I \alpha_I \pi_I$ be a non-zero function in $RM(M,r)$. We split this into two parts depending on whether $I$ contains $M$ or not. So

$$f = \left(\sum_{M\notin I} \alpha_I \pi_I\right) + \left(\sum_{M\in I} \alpha_I \pi_{I\setminus\{M\}}\right)\pi_M = f_0 + f_1.\pi_M \ .$$

The functions $f_0$ and $f_1$ do not depend on $x_M$ so we think of them as functions on $\mathbb{F}_2^{M-1}$. The sum $f_0$ is in $RM(M-1,r)$ while $f_1 \in RM(M-1,r-1)$.

Denote by $d_M$ the Hamming distance in $V = \mathbb{F}_2^M$ and $d_{M-1}$ the Hamming distance in $\mathbb{F}_2^{M-1}$. Observe that

$$\begin{array}{ll} f(x_1,x_2,\ldots,x_{M-1},0) = f_0(x_1,x_2,\ldots,x_{M-1}) & \text{on the set where } x_M = 0; \\ f(x_1,x_2,\ldots,x_{M-1},1) = (f_0+f_1)(x_1,x_2,\ldots,x_{M-1}) & \text{on the set where } x_M = 1. \end{array}$$

Since $f$ is non-zero, either $f_0$ or $f_0+f_1$ must be non-zero. If $f_0 = 0$, then $f_1$ is non-zero and

$$d_M(0,f) = d_{M-1}(0,f_1) \geqslant 2^{(M-1)-(r-1)} \ .$$

If $f_0 \neq 0$ and $f_1 = -f_0$, then $f_0 = -f_1 \in RM(M-1,r-1)$ so we have

$$d_M(0,f) = d_{M-1}(0,f_0) \geqslant 2^{(M-1)-(r-1)} \ .$$

Finally, if $f_0 \neq 0$ and $f_1 \neq -f_0$, then

$$d_M(0,f) = d_{M-1}(0,f_0) + d_{M-1}(0,f_0+f_1) \geqslant 2\times 2^{M-1-r}$$

since both $f_0$ and $f_0+f_1$ are non-zero elements of $RM(M-1,r)$. In all cases we have

$$d_M(0,f) \geqslant 2^{M-r}$$

which completes the inductive step. □

The Reed – Muller $RM(5,1)$ code was used by NASA in the Mariner mission to Mars. This code has rank $1+5 = 6$ and length $2^5 = 32$. So its rate of tranmission is $\frac{6}{32} = \frac{3}{16}$. The last proposition show that it has minimum distance $2^4 = 16$. So it is 15-error detecting and 7-error correcting.

## 12: CYCLIC CODES

A linear code is *cyclic* if $c_N c_1 c_2 \dots c_{N-2} c_{N-1}$ is a code word whenever $c_1 c_2 c_3 \dots c_{N-1} c_N$ is a code word. This means that the *shift map*:

$$T : \mathbb{F}^N \to \mathbb{F}^N \; ; \quad \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_N \end{pmatrix} \mapsto \begin{pmatrix} x_N \\ x_1 \\ x_2 \\ \vdots \\ x_{N-1} \end{pmatrix}$$

maps the code book into itself.

---

**Example:**
The matrix

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

defines a cyclic code over $\mathbb{F}_2$ since, if we label the columns of $C$ as $\boldsymbol{c}_1, \boldsymbol{c}_2, \boldsymbol{c}_3$ and $\boldsymbol{c}_4$, then

$$T(\boldsymbol{c}_4) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix} = \boldsymbol{c}_1 + \boldsymbol{c}_2 + \boldsymbol{c}_3 \; .$$

---

It is easier to describe cyclic codes if we identify $\mathbb{F}^N$ with the quotient $\mathbb{F}[X]/(X^N - 1)$. Recall that $\mathbb{F}[X]/(X^N - 1)$ is the set of equivalence classes of polynomials modulo $X^N - 1$. Given a polynomial $P(X)$, divide it by $X^N - 1$ to get $P(X) = Q(X)(X^N - 1) + R(X)$ with $\deg R(X) < N$. Then $P(X)$ is equivalent modulo $X^N - 1$ to

$$R(X) = r_0 + r_1 X + r_2 X^2 + \ldots + r_{N-1} X^{N-1} \in \{A(X) \in \mathbb{F}[x] : \deg A(X) < N\} \; .$$

We will write $[P(X)]$ for the equivalence class of $P(X)$ modulo $X^N - 1$. So we have seen how to identify $[P(X)]$ with $[R(X)]$ and hence with the vector

$$\begin{pmatrix} r_0 \\ r_1 \\ r_2 \\ \vdots \\ r_{N-2} \\ r_{N-1} \end{pmatrix} \in \mathbb{F}^N \; .$$

The shift map corresponds to

$$T : \mathbb{F}[X]/(X^N - 1) \to \mathbb{F}[X]/(X^N - 1) \; ; \quad [P(X)] \mapsto [X.P(X)]$$

because $X.X^{N-1} = X^N \equiv 1 \pmod{X^N - 1}$. The quotient homomorphism will be denoted by

$$q : \mathbb{F}[X] \to \mathbb{F}[X]/(X^N - 1) \; ; \quad P(X) \mapsto [P(X)] \; .$$

**Proposition 12.1** Cyclic codes
*Let $W$ be a vector subspace of $\mathbb{F}[X]/(X^N - 1)$ and let $J = q^{-1}(W)$. Then $W$ is the code book for a cyclic code if and only if $J$ is an ideal of $\mathbb{F}[X]$.*

*Proof:*

The quotient map $q$ is linear, so $J$ is certainly a vector subspace of $\mathbb{F}[X]$. Suppose that $W$ is cyclic. Then $[P(X)] \in W \Rightarrow [X.P(X)] \in W$ for any polynomial $P(X)$. This means that

$$P(X) \in J \Rightarrow X.P(X) \in J \ .$$

Therefore, if $P(X) \in J$, then

$$A(X)P(X) = \left(\sum a_k X^k\right) P(X) = \sum a_k X^K.P(X) \in J$$

for any polynomial $A(X)$. Hence $J$ is an ideal of $\mathbb{F}[X]$: $J \lhd \mathbb{F}[X]$.

Conversely, suppose that $J \lhd \mathbb{F}[X]$ and that $[P(X)] \in W$. Then $P(X) \in J$ and so $X.P(X) \in J$. This means that $[X.P(X)] \in W$. So $W$ is cyclic. □

**Corollary 12.2** Generators of cyclic codes
*Let $W$ be the code book for a cyclic code in $\mathbb{F}[X]/(X^N - 1)$. Then*

$$W = \{[A(X)G(X)] : A(X) \in \mathbb{F}[X]\}$$

*for a generator polynomial $G(X) \in \mathbb{F}[X]$. Moreover, $G(X)$ is a divisor of $X^N - 1$.*

*Proof:*

The proposition shows that $J = q^{-1}(W)$ is an ideal of $\mathbb{F}[X]$. Since $\mathbb{F}[X]$ is a principal ideal domain, this ideal must be of the form $G(X)\mathbb{F}[X]$ for some polynomial $G(X)$. Therefore, $W = \{[A(X)G(X)] : A(X) \in \mathbb{F}[X]\}$.

Now $0 = q(X^N - 1)$ is in $W$, so $X^N - 1 \in J$. Hence $G(X)$ must divide $X^N - 1$. □

We may always multiply the generator polynomial $G(X)$ by a scalar to ensure that it has leading coefficient 1. Then the generator polynomial is uniquely determined by the cyclic code book $W$. Since $G(X)$ divides $X^N - 1$, we may write $X^N - 1 = G(X)H(X)$. We call $H(X)$ the *check polynomial* for the code. Once again, it is unique up to a scalar multiple.

**Proposition 12.3**
*Let $G(X)$ be the generator of a cyclic code of length $N$ over $\mathbb{F}$. Then the vectors*

$$[G(X)],\ [X.G(X)],\ [X^2.G(X)],\ \ldots,\ [X^{N-D-2}.G(X)],\ [X^{N-D-1}.G(X)]$$

*are a basis for the code book in $\mathbb{F}[X]/(X^N - 1)$. Hence the code book has rank $N - D$ where $D = \deg G(X)$.*

*Proof:*

We know that each of the products $X^k.G(X)$ is in the ideal $J$, so the vectors $[X^k.G(X)]$ must lie in the code book $W$ for the cyclic code.

Every vector in $W$ is of the form $[G(X)P(X)]$ for some polynomial $P(X) \in \mathbb{F}[X]$. By reducing $G(X).P(X)$ modulo $X^N - 1$ we may ensure that $\deg P(X) < N - D$. So any vector in $W$ is of the form

$$p_0[G(X)] + p_1[X.G(X)] + p_2[X^2.G(X)] + \ldots + p_{N-D-1}[X^{N-D-1}.G(X)] \ .$$

Thus the vectors $[G(X)],\ [X.G(X)],\ [X^2.G(X)],\ \ldots,\ [X^{N-D-2}.G(X)],\ [X^{N-D-1}.G(X)]$ span $W$.

Suppose that these vectors were linearly dependent, say

$$p_0[G(X)] + p_1[X.G(X)] + p_2[X^2.G(X)] + \ldots + p_{N-D-1}[X^{N-D-1}.G(X)] = 0 \ .$$

Then

$$[(p_0 + p_1X + p_2X^2 + \ldots + p_{N-D-1}X^{N-D-1})G(X)] = [P(X)G(X)] = 0$$

so $(X^N - 1)|P(X)G(X)$. This means that $H(X)|P(X)$. Since $\deg P(X) < N - D = \deg H(X)$, this implies that $P(X) = 0$. Hence the vectors $[G(X)],\ [X.G(X)],\ [X^2.G(X)],\ \ldots,\ [X^{N-D-2}.G(X)]$, $[X^{N-D-1}.G(X)]$ are linearly independent. □

Using the basis for the code book in this proposition, we see that a matrix for the code is the $N \times (N-D)$ matrix:

$$C = \begin{pmatrix} g_0 & 0 & 0 & \dots & 0 \\ g_1 & g_0 & 0 & \dots & 0 \\ g_2 & g_1 & g_0 & \dots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ g_D & g_{D-1} & g_{D-2} & \dots & \\ 0 & g_D & g_{D-1} & \dots & \\ 0 & 0 & g_D & \dots & \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \dots & g_D \end{pmatrix}$$

where $G(X) = g_0 + g_1 X + g_2 X^2 + \ldots + g_D X^D$.

Let $S$ be the $D \times N$ matrix:

$$S = \begin{pmatrix} 0 & 0 & 0 & \dots & h_2 & h_1 & h_0 \\ 0 & 0 & 0 & \dots & h_1 & h_0 & 0 \\ 0 & 0 & 0 & \dots & h_0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \\ 0 & h_{N-D} & h_{N-D-1} & \dots & 0 & 0 & \\ h_{N-D} & h_{N-D-1} & h_{N-D-2} & \dots & 0 & 0 & \end{pmatrix}$$

where $H(X) = h_0 + h_1 X + h_2 X^2 + \ldots + h_{N-D} X^{N-D}$. Then $h_{N-D} \neq 0$, so the rows of $S$ are linearly independent. So its rank is $D$ and its nullity must be $N-D$. However, it is easy to check that each column of the matrix $C$ lies in the kernel of $S$, because $G(X)H(X) = X^N - 1$ so

$$\sum_j h_{m-j} g_j = 0 \qquad \text{for } \ 0 < m < N \ .$$

This means that the kernel of $S$ is spanned by the columns of $C$. So $\ker S = W = \mathrm{Im}(C)$ and $S$ is a syndrome matrix for the cyclic code.

---

**Example:**
Over the field $\mathbb{F}_2$ of integers modulo 2 we have

$$X^7 - 1 = ((1 + X + X^3)(1 + X + X^2 + X^4) \ .$$

Let $G(X) = 1 + X + X^3$. This generates a cyclic binary code of length 7. A syndrome matrix is

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

All of the non-zero vectors in $\mathbb{F}_2^3$ occur as columns in this matrix, so the code is Hamming's code with the terms of the code re-ordered.

---

1

## 13: BCH CODES

### 13.1 The Characteristic of a Field

Let $K$ be a finite field and consider the sequence

$$0_K,\ 1_K,\ 1_K + 1_K,\ 1_K + 1_K + 1_K,\ \ 1_K + 1_K + 1_K + 1_K,\ \ldots\ .$$

Eventually this sequence must repeat, so we obtain a first natural number $p$ with the sum of $p$ terms $1_K + 1_K + \ldots + 1_K$ equal to 0. If $p$ factorised, then one of the factors would also have this property, so $p$ must be a prime number. It is the *characteristic of* $K$. The set of $p$ elements $0_K, 1_K, 1_K + 1_K$, $1_K + 1_K + 1_K$, ... is then a subfield of $K$ which is isomorphic to the integers modulo $p$. Hence $K$ contains $\mathbb{F}_p$. ($K$ is then a vector space over $\mathbb{F}_p$ of some dimension $r$, so it has $p^r$ elements. Thus $K$ has characteristic $p$ when it is one of the fields $\mathbb{F}_{p^r}$.)

We will only consider fields with characteristic 2 in this section, so they are all fields of order $2^r$ and will give us binary codes. The arguments readily extend to other characteristics.

We know that any cyclic code has a generator polynomial $G(X)$ with $G(X)|(X^N - 1)$ where $N$ is the length of the code. We will assume that $N$ is odd. It can be shown (in the Galois Theory course) that there is some finite field $K$ containing $\mathbb{F}$ in which $X^N - 1$ factorises completely into linear factors (or "splits"). The next lemma shows when these factors are all distinct.

**Lemma 13.1** Separable fields
*Let $N$ be an odd natural number and $K$ a field with characteristic 2 in which $X^N - 1$ factorises completely into $N$ linear factors. Then $X^N - 1$ has $N$ distinct roots in $K$. These form a cyclic group.*

*Proof:*
Suppose that $X^N - 1$ had a repeated root $\alpha$ in $K$. Then

$$X^N - 1 = (X - \alpha)^2 P(X)$$

for some polynomial $P(X) \in K[X]$. Now if we differentiate this formally we obtain

$$NX^{N-1} = 2(X - \alpha)P(X) + (X - \alpha)^2 P'(X) = (X - \alpha)\left(2P(X) + (X - \alpha)P'(X)\right)\ .$$

So $X - \alpha$ divides both $X^N - 1$ and $NX^{N-1}$. Since $N$ is odd, $N1_K \neq 0_K$. So $X - \alpha$ must divide the highest common factor of $X^N - 1$ and $X^{N-1}$. This highest common factor is 1, so we get a contradiction.

We have now shown that the set $S$ of roots of $X^N - 1$ in $K$ contains $N$ distinct elements. It is clearly a subgroup of the group $K^\times$. We know that $K^\times$ is cyclic, so the subgroup $S$ must be also. □

The lemma shows that we can find a *primitive root $\alpha$ of* $X^N - 1$ in the field $K$ with the other roots being

$$\alpha, \alpha^2, \alpha^3, \ldots, \alpha^{N-1}\ .$$

The lemma is not true if $N$ is even. For example, $X^2 - 1 = (X - 1)^2$.

### 13.2 BCH codes

Let $\mathbb{F}$ be a finite field with characteristic 2. Choose an odd integer $N$ and let $K$ be a field containing $\mathbb{F}$ in which $X^N - 1$ factorises completely into linear factors. Take $\alpha$ as a primitive root of $X^N - 1$ in the field $K$. The Bose - Ray Chaudhuri – Hocquenghem (BCH) code with design distance $\delta$ is the cyclic code of length $N$ over $\mathbb{F}$ defined by the ideal

$$J = \{P(X) \in \mathbb{F}[X] : P(\alpha) = P(\alpha^2) = P(\alpha^3) = \ldots = P(\alpha^{\delta-1}) = 0\} \ .$$

Here $\delta$ is a natural number with $1 \leqslant \delta \leqslant N$.

The generating polynomial $G(X)$ for this code is the minimal polynomial in $\mathbb{F}[X]$ that is 0 at each of the points $\alpha, \alpha^2, \ldots, \alpha^{\delta-1}$ in $K$. It is thus the least common multiple of the minimal polynomials for these points. It is clearly a factor of $X^N - 1$. So the BCH code is a cyclic linear code of length $N$.

---

**Example:**
Consider the polynomial $X^7 - 1$ over $\mathbb{F}_2$. The roots of this form a cyclic group of order 7, so every root is a primitive root. We can factorise $X^7 - 1$ over $\mathbb{F}_2$ as

$$X^7 - 1 = (X - 1)(X^3 + X^2 + 1)(X^3 + X + 1) \ .$$

The cubic factors are irreducible over $\mathbb{F}_2$ since, if they factorized further, then one of the factors would be linear and so give a root in $\mathbb{F}_2$.

Let $\alpha$ be one of the roots of $X^3 + X + 1$ in $K$. Then $\alpha^7 = 1$ and so

$$(\alpha^2)^3 + (\alpha^2) + 1 = \alpha^6 + \alpha^2 + 1 = \alpha^6(1 + \alpha^3 + \alpha) = 0 \ .$$

This shows that $\alpha^2$ is also a root of $X^3 + X + 1$. Repeating the argument shows that $\alpha^4 = (\alpha^2)^2$ is a root. These roots are distinct, so

$$X^3 + X + 1 = (X - \alpha)(X - \alpha^2)(X - \alpha^4) \ .$$

Similarly we see that

$$X^3 + X^2 + 1 = (X - \alpha^3)(X - \alpha^5)(X - \alpha^6) \ .$$

The BCH code with design distance 3 is given by the ideal

$$J = \{P(X) \in \mathbb{F}_2[X] : P(\alpha) = P(\alpha^2) = 0\} \ .$$

The generating polynomial for this ideal is $X^3 + X + 1$. We saw at the end of lecture 12 that this gave a rearrangement of Hamming's code.

---

We want to prove that the minimum distance for a BCH code is at least as big as the design distance $\delta$. To do this we need to recall a result about determinants.

**Lemma 13.2** van der Monde determinants
*The determinant*

$$\Delta(X_1, X_2, \ldots, X_K) = \begin{vmatrix} X_1 & X_2 & X_3 & \ldots & X_K \\ X_1^2 & X_2^2 & X_3^2 & \ldots & X_K^2 \\ X_1^3 & X_2^3 & X_3^3 & \ldots & X_K^3 \\ \vdots & \vdots & \vdots & & \vdots \\ X_1^K & X_2^K & X_3^K & \ldots & X_K^K \end{vmatrix}$$

*satisfies*

$$\Delta(X_1, X_2, \ldots, X_K) = \left( \prod_{k=0}^{K} X_k \right) \left( \prod_{i<j} (X_j - X_i) \right) \ .$$

*Proof:*

We will treat the $(X_k)$ as independent indeterminants and prove the lemma by induction on $K$. The result is clearly true for $K = 1$.

Consider $\Delta(X_1, X_2, \ldots, X_K)$ as a polynomial of degree $K$ in $X_K$ with coefficients that are polynomials in $X_1, X_2, \ldots, X_{K-1}$. When $X_K = 0$ or $X_1$ or $X_2$ or $\ldots$ or $X_{K-1}$, then the determinant is obviously 0, so

$$X_K \left( \prod_{i<K} (X_K - X_i) \right)$$

must divide the determinant. Hence

$$\Delta(X_1, X_2, \ldots, X_K) = c(X_1, X_2, \ldots, X_{K-1}) X_K \left( \prod_{i<K} (X_K - X_i) \right)$$

for some polynomial $c(X_1, X_2, \ldots, X_{K-1})$. The leading coefficients of both sides of this equation are

$$\Delta(X_1, X_2, \ldots, X_{K-1}) \qquad \text{and} \qquad c(X_1, X_2, \ldots, X_{K-1}) \; .$$

So we see that

$$\Delta(X_1, X_2, \ldots, X_K) = \Delta(X_1, X_2, \ldots, X_{K-1}) X_K \left( \prod_{i<K} (X_K - X_i) \right)$$

which completes the induction. □

**Theorem 13.3** Minimum distance for BCH codes
*The minimum distance for a BCH code with design distance $\delta$ is at least $\delta$.*

*Proof:*

Let $P(X) = p_0 + p_1 X + \ldots + p_{N-1} X^{N-1}$ be a non-zero polynomial in the code book. Then we have

$$\begin{pmatrix} 1 & \alpha & \alpha^2 & \alpha^3 & \ldots & \alpha^{N-1} \\ 1 & \alpha^2 & \alpha^4 & \alpha^6 & \ldots & \alpha^{2(N-1)} \\ 1 & \alpha^3 & \alpha^6 & \alpha^9 & \ldots & \alpha^{3(N-1)} \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 1 & \alpha^{\delta-1} & \alpha^{2(\delta-1)} & \alpha^{3(\delta-1)} & \ldots & \alpha^{(N-1)(\delta-1)} \end{pmatrix} \begin{pmatrix} p_0 \\ p_1 \\ p_2 \\ \vdots \\ p_{N-2} \\ p_{N-1} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \; .$$

Call the matrix above $A$. Lemma 13.1 shows that any $\delta - 1$ columns of $A$ are linearly independent because no two of the $N$ powers of $\alpha$ are equal or 0. This means that there must be at least $\delta$ non-zero coefficients $(p_k)$ in order for $A\boldsymbol{p}$ to be $\mathbf{0}$. Hence the minimum distance is at least $\delta$. □

It remains for us to consider how to decode a message. Since our BCH code is a linear code, we can use the general methods for finding the closest code word to any message we receive. However, we can also exploit the algebraic definition of BCH codes to find more efficient ways to do this.

We know from Proposition 7.3 that our BCH code is $t$-error correcting, where $t = \lfloor \frac{1}{2}(\delta - 1) \rfloor$. So suppose that a code word $\boldsymbol{c}$ is sent and we receive $\boldsymbol{r} = \boldsymbol{c} + \boldsymbol{e}$ where the error vector $\boldsymbol{e}$ has at most $t$ non-zero entries. How do we find $\boldsymbol{e}$ and thence recover the sent code word $\boldsymbol{c}$?

Let $C(X), R(X)$ and $E(X)$ be the polynomials with degree less than $N$ that correspond to the vectors $\boldsymbol{c}, \boldsymbol{r}$ and $\boldsymbol{e}$. We know that the code word $C(X)$ satisfies

$$C(\alpha) = C(\alpha^2) = \ldots = C(\alpha^{\delta-1}) = 0 \; .$$

So

$$R(\alpha) = E(\alpha) \ , \quad R(\alpha^2) = E(\alpha^2) \ , \quad \ldots, R(\alpha^{\delta-1}) = E(\alpha^{\delta-1}) \ .$$

First calculate $R(\alpha^j)$ for $j = 1, 2, \ldots, \delta - 1$. If these are all 0 then $R(X)$ is a code word and there have been no errors (or at least $\delta + 1$ of them).

Otherwise let $\mathcal{E} = \{i : e_i \neq 0\}$ be the set of indices at which errors occur and assume that $0 < |\mathcal{E}| \leqslant t$. The *error locator polynomial* is

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

This is a polynomial (over $K$) of degree $|\mathcal{E}|$ with constant term 1. If we know $\sigma(X)$ then we can easily find which powers $\alpha^{-i}$ are roots of $\sigma(X)$ and hence find the indices where errors have occurred. We could then correct the errors by changing those entries. So we need to calculate $\sigma(X)$.

Consider the formal power series

$$\eta(X) = \sum_{j=1}^{\infty} E(\alpha^j) X^j \ .$$

(Since $\alpha^N = 1$, the coefficients of this power series repeat.) Note that, for the first few coefficients, $E(\alpha^j) = R(\alpha^j)$ for $j = 1, 2, \ldots, \delta - 1$. So we can calculate these coefficients in terms of the received word $\boldsymbol{r}$. These are the only coefficients we will use.

$$\eta(X) = \sum_{j=1}^{\infty} E(\alpha^j) X^j = \sum_{j=1}^{\infty} \sum_{i \in \mathcal{E}} \alpha^{ij} X^j = \sum_{i \in \mathcal{E}} \sum_{j=1}^{\infty} \alpha^{ij} X^j = \sum_{i \in \mathcal{E}} \frac{\alpha^i X}{1 - \alpha^i X}$$

Write this as $\dfrac{\omega(X)}{\sigma(X)}$ for the polynomial

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

Note that both $\sigma(X)$ and $\omega(X)$ have degree $|\mathcal{E}| \leqslant t$.

We have shown that $\sigma(X)\eta(X) = \omega(X)$. Writing this out explicitly and using $E(\alpha^k) = R(\alpha^k)$ for $k = 1, 2, \ldots, 2t$ we obtain

$$\begin{aligned} \left(\sigma_0 + \sigma_1 X + \ldots + \sigma_t X^t\right) \left(R(\alpha) X + R(\alpha^2) X^2 + \ldots + R(\alpha^{2t}) X^{2t} + E(\alpha^{2t+1}) X^{2t+1} + \ldots\right) = \\ = \omega_0 + \omega_1 X + \ldots + \omega_t X^t \ . \end{aligned}$$

The coefficients of $X^n$ for $t < n \leqslant 2t$ are

$$\sum_{j=0}^{t} \sigma_j R(\alpha^{n-j}) = 0$$

which do not involve any of the terms $E(\alpha^k) X^k$. So we get equations

$$\begin{pmatrix} R(\alpha^{t+1}) & R(\alpha^t) & \ldots & R(\alpha) \\ R(\alpha^{t+2}) & R(\alpha^{t+1}) & \ldots & R(\alpha^2) \\ \vdots & \vdots & & \vdots \\ R(\alpha^{2t}) & R(\alpha^{2t-1}) & \ldots & R(\alpha^t) \end{pmatrix} \begin{pmatrix} \sigma_0 \\ \sigma_1 \\ \sigma_0 \\ \vdots \\ \sigma_t \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \ .$$

The matrix above is a $t \times (t + 1)$ matrix, so there is always a vector $\sigma \neq \mathbf{0}$ in its kernel. This tells us what the error locator polynomial $\sigma(X)$ is and hence what the errors are that we should correct.

The special case of a BCH code where the field $\mathbb{F}$ is $\mathbb{F}_q$ and $N = q - 1$ is also called a Reed – Solomon code. These are very widely used. In particular, CDs are encoded using two interleaved Reed – Solomon codes. Both are defined over $\mathbb{F}_{2^8} = \mathbb{F}_{256}$ with $\delta = 5$. They have lengths $N = 32$ and 28 respectively. The interleaving spreads the information physically on the disc so that bursts of successive errors, as caused by a scratch, can be corrected more reliably. A CD can correct a burst of about $4,000$ binary errors.

## 14: LINEAR FEEDBACK SHIFT REGISTERS

### 14.1 Definitions

Suppose that we have $K$ registers each of which can contain one value from the finite field $\mathbb{F} = \mathbb{F}_q$ with $q$ elements. Initially these contain the values $x_0, x_1, \ldots, x_{K-1}$, which is called the *initial fill*. At each time step, the values in the registers are shifted down and the final register is filled by a new value determined by the old values in all of the registers.

Denote the values in the registers at time $t = 0, 1, 2, \ldots$ by $X_0(t), X_1(t), \ldots, X_{K-1}(t)$. Then we have

$$\begin{aligned} X_0(t+1) &= X_1(t) \\ X_1(t+1) &= X_2(t) \\ &\vdots \\ X_{K-2}(t+1) &= X_{K-1}(t) \\ X_{K-1}(t+1) &= f(X_0(t), X_1(t), \ldots, X_{K-1}(t)) \ . \end{aligned}$$

The function $f$ is called the *feedback function*. The system is called a *linear feedback shift register* (LFSR) when the feedback function is linear, say $f(X_0, X_1, \ldots, X_{K-1}) = -c_0X_0 - c_1X_1 - \ldots - c_{K-1}X_{K-1}$ so that

$$c_0X_0(t) + c_1X_1(t) + \ldots c_{K-1}X_{K-1}(t) + X_K(t) = 0$$

for each $t$.

Such a linear feedback shift register gives an infinite stream of values

$$X_0(0),\ X_0(1),\ X_0(2),\ ,\ldots,\ X_0(t),\ \ldots\ .$$

The values in register $r$ are just these values with the first $r-1$ discarded and the others shifted back. This stream of values begins with the initial fill

$$X_0(0) = x_0,\ X_0(1) = x_1,\ X_0(2) = x_2,\ \ldots,\ X_0(K-1) = x_{K-1}$$

and satisfies the recurrence relation

$$c_0X_0(t) + c_1X_0(t+1) + c_2X_0(t+2) + \ldots c_{K-1}X_0(t+K-1) + X_0(t+K) = 0\ .$$

The polynomial $C(X) = c_0 + c_1X + c_2X^2 + \ldots + c_{K-1}X^{K-1} + X^K$ is the *feedback polynomial*. It is the characteristic polynomial for the recurrence relation and so determines the solutions.

It is simple to produce such feedback shift registers using computers and so to produce such streams. In superficial ways the stream of values appears random but we will see that this is not the case.

We will always assume that the first coefficient $c_0$ for the feedback function is not zero. For, if $c_0 = 0$, then the value in the 0th register does not affect any later values and we should consider the remaining $K-1$ registers as a simpler LFSR.

**Proposition 14.1** Periodicity for LFSR
*A linear feedback shift register is periodic.*

This means that there is an integer $N$ with $X_0(t+N) = X_0(t)$ for each time $t$. The smallest such integer $N > 0$ is the *period* for the LFSR.

*Proof:*

For each time $t$, let $\boldsymbol{V}(t)$ be the vector

$$\boldsymbol{V}(t) = \begin{pmatrix} X_0(t) \\ X_0(t+1) \\ X_0(t+2) \\ \vdots \\ X_0(t+K-1) \end{pmatrix} \in \mathbb{F}^K \ .$$

There are only a finite number $q^K$ of vectors in $\mathbb{F}^K$ so the sequence $(\boldsymbol{V}(t))_{t\in\mathbb{N}}$ must eventually repeat a value taken earlier. Suppose that this first occurs at $\boldsymbol{V}(N)$ with $\boldsymbol{V}(N) = \boldsymbol{V}(j)$ for some $0 \leqslant j < N$.

The definition of the LFSR shows that

$$\boldsymbol{V}(t+1) = \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ -c_0 & -c_1 & -c_2 & \dots & -c_{K-1} \end{pmatrix} \boldsymbol{V}(t) \ .$$

Write this as $\boldsymbol{V}(t+1) = M\boldsymbol{V}(t)$. The matrix $M$ has determinant $\pm c_0$, which we are assuming to be non-zero. So $M$ is invertible.

We have $\boldsymbol{V}(t) = M^t\boldsymbol{V}(0)$, so the first repeat gives

$$M^N\boldsymbol{V}(0) = M^j\boldsymbol{V}(0) \ .$$

If $j$ were not 0, then we could multiply by $M^{-1}$ to get an earlier repeat. Hence we must have $j = 0$ and so $M^N\boldsymbol{V}(0) = \boldsymbol{V}(0)$. Consequently, the sequence of vectors $(\boldsymbol{V}(t))$ is periodic with period $N$. Looking at the first entry in these vectors shows that $(X_0(t))$ is also periodic. □

---

**Exercise:**
What happens when the coefficient $c_0$ is allowed to be 0? Do we still get periodicity when the feedback function $f$ is not assumed to be linear? Is the matrix $M$ in the proof periodic? What is its characteristic polynomial?

---

The matrix $M$ in the proof above clearly maps $\mathbf{0}$ to itself. Hence, the largest the period can be is $N = q^K - 1$. In this case the sequence $\boldsymbol{V}(0), \boldsymbol{V}(1), \dots, \boldsymbol{V}(N-2), \boldsymbol{V}(N-1)$ takes every value in $\mathbb{F}^K \setminus \{0\}$ exactly once. In one period of length $q^K - 1$, we then see that 0 occurs $q^{K-1} - 1$ times while every other number in $\mathbb{F}$ occurs $q^{K-1}$ times. In this, rather weak, sense the sequence is random. However, the sequence is periodic so it becomes entirely predictable once we have at least one period of data.

Let $N$ be the period for a linear feedback shift register and set

$$\boldsymbol{W}(t) = \begin{pmatrix} X_0(t) \\ X_0(t+1) \\ X_0(t+2) \\ \vdots \\ X_0(t+N-1) \end{pmatrix} \in \mathbb{F}^N \ .$$

Then the vectors $\boldsymbol{W}(t+1)$ is a cyclic shift of $\boldsymbol{W}(t)$ because $X_0(t+N) = X_0(t)$. Also,

$$c_0\boldsymbol{W}(0) + c_1\boldsymbol{W}(1) + \ldots + c_{K-1}\boldsymbol{W}(K-1) + \boldsymbol{W}(K) = \mathbf{0} \ .$$

So we see that the vectors $\boldsymbol{W}(0), \boldsymbol{W}(1), \dots, \boldsymbol{W}(K-1)$ span the code book for a cyclic code of length $N$.

## 14.2 The Berlekamp – Massey Algorithm

Suppose that you receive a sequence of values $(a_t)_{t\in\mathbb{N}}$ from the finite field $\mathbb{F}$ that you believe has come from a linear feedback shift register. Can you determine which linear feedback shift register has produced it?

This is simply a matter of solving linear equations to find the coefficients in the feedback polynomial. Suppose that the sequence came from a linear feedback shift register with $K$ registers and feedback polynomial

$$C(X) = c_0 + c_1X + c_2X^2 + \ldots + c_{K-1}X^{K-1} + X^K \ .$$

Then we would have

$$c_0a_t + c_1a_{t+1} + \ldots + c_{K-1}a_{t+K-1} + a_{t+K} = 0$$

for every $t$. Consequently,

$$\begin{pmatrix} a_0 & a_1 & a_2 & \ldots & a_{K-2} & a_{K-1} \\ a_1 & a_2 & a_3 & \ldots & a_{K-1} & a_K \\ a_2 & a_3 & a_4 & \ldots & a_K & a_{K+1} \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ a_{K-2} & a_{K-1} & a_K & \ldots & a_{2K-4} & a_{2K-3} \\ a_{K-1} & a_K & a_{K+1} & \ldots & a_{2K-3} & a_{2K-2} \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{K-1} \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix} \ . \qquad (*)$$

Hence the matrix above must have determinant 0.

Berlekmap and Massey formalised this as an algorithm to find the linear feedback shift register that generated $(a_t)$. Begin with the smallest value of $K$ that might be possible for the number of registers and compute the determinant

$$\begin{vmatrix} a_0 & a_1 & a_2 & \ldots & a_{K-2} & a_{K-1} \\ a_1 & a_2 & a_3 & \ldots & a_{K-1} & a_K \\ a_2 & a_3 & a_4 & \ldots & a_K & a_{K+1} \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ a_{K-2} & a_{K-1} & a_K & \ldots & a_{2K-4} & a_{2K-3} \\ a_{K-1} & a_K & a_{K+1} & \ldots & a_{2K-3} & a_{2K-2} \end{vmatrix} \ .$$

If this is non-zero, then we can not solve the problem using $K$ registers. So increase $K$ by 1 and repeat the process. If the determinant is 0, then solve $(*)$ above to find the coefficients $c_j$. Then set up a linear feedback shift register with initial fill $a_0, a_1, \ldots, a_{K-1}$ and feedback polynomial $C(X) = c_0 + c_1X + c_2X^2 + \ldots + c_{K-1}X^{K-1} + X^K$. check if the stream produced by this agrees with the original stream. If it does we are finished. If it does not, then consider other solutions $(c_j)$ or increase $K$.

## 14.3 Power Series

Let $A(X) = a_0 + a_1X + \ldots + a_DX^D$ and $B(X) = 1 + b_1X + \ldots + b_KX^K$ be two polynomials in $\mathbb{F}[X]$. If we write $B(X) = 1 - \beta(X)$, then we have

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

Expanding the powers of $\beta(X)$ gives a formal power series for $1/B(X)$. Multiplying this by $A(X)$ we obtain a formal power series for $A(X)/B(X)$, say

$$\frac{A(X)}{B(X)} = \sum_{j=0}^{\infty} u_jX^j \ .$$

In these formal power series we are not concerned about convergence but merely wish the coefficients of each power of $X$ on each side of the equation to match. Hence we need

$$A(X) = B(X) \left( \sum_{j=0}^{\infty} u_j X^j \right)$$

which is equivalent to

$$a_n = \sum_{j=0}^{n} b_j u_{n-j} \qquad \text{for } n = 0, 1, 2, \ldots \ .$$

Thus the sequence $(u_j)$ satisfies

$$u_n = a_n - \sum_{j=1}^{n} b_j u_{n-j} \qquad \text{for } n = 0, 1, \ldots, D;$$

$$u_n = \quad - \sum_{j=1}^{n} b_j u_{n-j} \qquad \text{for } n > D \ .$$

This shows that the sequence $(u_j)$ is the stream produced by a linear feedback shift register with feedback polynomial

$$C(X) = b_K + b_{K-1}X + b_{K-2}X^2 + \ldots + b_1 X^{K-1} + X^K \ .$$

This shows that determining whether a stream of numbers from $\mathbb{F}$ is the output of a LFSR is the same as determining whether a formal power series is the quotient of two polynomials. You should compare this to the method used to decode a BCH code in the previous lecture.

---

**Exercise:**
You learnt at school that a decimal repeats if and only if it represents a rational number. How does this relate to LFSRs and the periodicity proved in Proposition 14.1?

---