Coding Theory

J.W.P. Hirschfeld

Spring 2014

Contents

Chapter 1

Introduction to Error-Correcting Codes

Motivation This theory shows how to solve a practical problem using the well-established mathematical tools of Linear Algebra and Finite Fields.

Difference from Cryptography Coding Theory and Cryptography are two important parts of the modern theory of Information Science.

Cryptography, which is some 2000 years old, is the mathematical theory of sending secret messages.

Coding Theory, which only dates from 1948, is the mathematical theory of sending messages that arrive with the same content as when they were sent.

Example 1.1. To send just the two messages YES and NO, the following encoding suffices:

$$\text{YES} = 1, \quad \text{NO} = 0.$$

If there is an error, say 1 is sent and 0 arrives, this will go undetected. So, add some redundancy:

$$\text{YES} = 11, \quad \text{NO} = 00.$$

Now, if 11 is sent and 01 arrives, then an error has been detected, but not corrected, since the original messages 11 and 00 are equally plausible.

So, add further redundancy:

$$\text{YES} = 111, \quad \text{NO} = 000.$$

Now, if 010 arrives, and it is supposed that there was at most one error, we know that 000 was sent: the original message was NO.

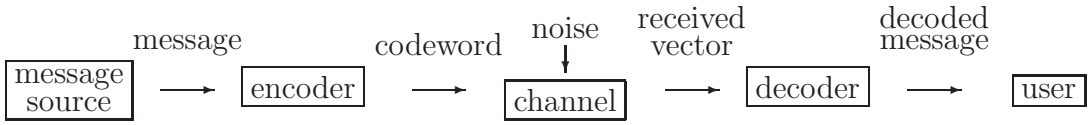
Note that the information is still in the first symbol; the other two are purely for error-correction!

The philosophy Error correction codes are used to correct errors when messages are transmitted through a noisy communication channel.

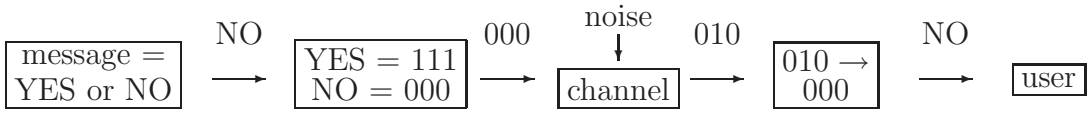
The *channel* may be a telephone line, a high frequency radio link or a satellite communication link.

The *noise* may be human error, lightning, equipment faults, etc.

The object of a code is to encode the data by adding a certain amount of redundancy so that the original message can be recovered if not too many errors occur in the transmission.



In Example 1.1,



Definition 1.2. A *binary code* is a set of sequences of 0's and 1's; each sequence is a *codeword*.

In Example 1.1, the code is $\{000,\ 111\}$. This is a *binary repetition code of length* 3.

Definition 1.3. (1) A q-ary code C is a set of sequences where each symbol is from a set $\mathbf{F}_q = \{\lambda_1, \ldots, \lambda_q\}$. Usually, $\mathbf{F}_q$ is a finite field.

(2) A 2-ary code is a *binary* code; a 3-ary code is a *ternary* code.

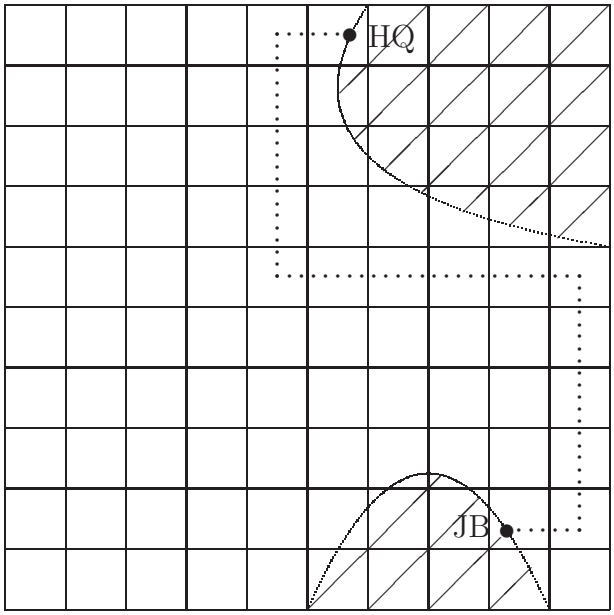
Example 1.4. The set of all words in the English language is a code over the 26-letter alphabet $\{A, B, \ldots, Z\}$. The codewords are not all the same length.

Definition 1.5. A *q-ary code C of length n* is a subset of $(\mathbf{F}_q)^n$.

Example 1.6. The set of all 11-digit telephone numbers in the UK is a 10-ary code of length 11. It is not designed for error-correcting, with area codes being important. However, it would be possible to allow for a single misdial to be corrected.

Example 1.7. On a map laid out as a grid, HQ and JB have identical maps. For JB to return to HQ, a message is transmitted in terms of the instructions $N,\ E,\ W,\ S$. The message is

$$E\,N\,N\,N\,N\,W\,W\,W\,W\,W\,N\,N\,N\,N\,E.$$



The shortest binary code is

$$C_1 = \begin{array}{c}\{\ 00,\ 01,\ 10,\ 11\ \}\\ \ N\ \ \ E\ \ \ W\ \ \ S\end{array}.$$

This code does not allow for errors; so add an extra digit, namely a *parity check*:

$$C_2 = \begin{array}{c}\{\ 000,\ 011,\ 101,\ 110\ \}\\ \ N\ \ \ \ E\ \ \ \ W\ \ \ \ S\end{array}.$$

This will *detect* an error. So JB could ask for another transmission if he received, say, 010. However, this code would not correct the error, since N, E, S are all messages with only one digit different from 010. So use

$$C_3 = \begin{array}{c}\{\ 00000,\ 01101,\ 10110,\ 11011\ \}\\ \ N\ \ \ \ \ E\ \ \ \ \ W\ \ \ \ \ S\end{array}.$$

If the received message is 01100, then it has two digits different from N, one digit different from E, three digits different from W, and four digits different from S; so it is decoded as $E = 01101$.

Definition 1.8. For $x, y \in (\mathbf{F}_q)^n$, the *(Hamming) distance* $d(x, y)$ is the number of coordinates in which they differ: that is, if

$$\begin{aligned} x &= x_1 x_2 \cdots x_n, \\ y &= y_1 y_2 \cdots y_n, \end{aligned}$$

then

$$d(x, y) = |\{i \mid x_i \neq y_i\}|.$$

Example 1.9. In C_3,

$$\begin{aligned} &d(N, E) = d(N, W) = 3,\ d(N, S) = 4, \\ &d(E, W) = 4,\ d(E, S) = d(W, S) = 3. \end{aligned}$$

Theorem 1.10. *The Hamming distance is a metric.*

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

By definition, $d(x,y) = 0 \iff x_i = y_i$ all $i \iff x = y$.

(ii) $d(x,y) = d(y,x)$ by definition.

(iii) $d(x,y) \leq d(x,z) + d(z,y)$.

Here, $d(x,y)$ is the minimum number of changes of coordinates of x to make it y. But a change from x to z requires $d(x,z)$ changes of coordinates and changing z to y requires $d(z,y)$ coordinate changes. Hence $d(x,y) \leq d(x,z) + d(z,y)$. □

Definition 1.11. The *minimum distance* of a q-ary code C of length n is

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

In Example 1.7, $d(C_1) = 1,\ d(C_2) = 2,\ d(C_3) = 3$.

Definition 1.12. *Nearest neighbour decoding*
Send x, receive y. Then, choose $x' \in C$ such that $d(x',y)$ is minimum.

This strategy depends on two assumptions:

(1) each symbol has the same probability $t\,(< \frac{1}{2})$ of being wrongly received;

(2) if a symbol is wrongly received, then each of the $q-1$ errors is equally likely.

Example 1.13. In a binary code of length n,

$$P(\text{exactly } i \text{ errors in specified positions}) = t^i(1-t)^{n-i}.$$

Since this is greatest for $i = 0$, so nearest neighbour decoding is also *maximum likelihood decoding*.

Example 1.14. $C = \{000, 111\}$, the binary repetition code of length 3.
Suppose 111 is transmitted. Then the received words decoded as 111 are

$$111, 011, 101, 110.$$

So

$$P(\text{decoding as } 111) = (1-t)^3 + 3t(1-t)^2.$$

Suppose that $t = 0.1$, that is, one symbol in 10 is wrong. So

$$\begin{aligned} P(\text{correct decoding}) &= 0.9^3 + 3 \times 0.1 \times 0.9^2 \\ &= 0.729 + 0.243 = 0.972; \\ P(\text{incorrect coding}) &= 0.028. \end{aligned}$$

It will be shown, for linear codes, that P(incorrect coding), that is, the *word error probability*, is independent of the codeword sent.

Definition 1.15. A code is *e-error correcting* if it can correct e errors.

Definition 1.16. A *q-ary* (n, M, d) *code* or $(n, M, d)_q$ *code* is a code C of length n, cardinality $M = |C|$ and minimum distance d over the alphabet $\mathbf{F}_q$.

In Example 1.7,

C_1 is a binary $(2, 4, 1)$ code or $(2, 4, 1)_2$ code;
C_2 is a binary $(3, 4, 2)$ code or $(3, 4, 2)_2$ code;
C_3 is a binary $(5, 4, 3)$ code or $(5, 4, 3)_2$ code.

Definition 1.17. For $x_0 \in (\mathbf{F}_q)^n$ and $r \in \mathbf{Z}$, $r \geq 0$, the *ball* of *centre* x_0 and *radius* r is

$$S(x_0, r) = \{x \in (\mathbf{F}_q)^n \mid d(x_0, x) \leq r\}.$$

Example 1.18. (1) In $(\mathbf{F}_2)^3$ with $\mathbf{F}_2 = \{0, 1\}$,

$$S(000, 1) = \{000, 100, 010, 001\}.$$

(2) In $(\mathbf{F}_3)^3$, with $\mathbf{F}_3 = \{0, 1, 2\}$,

$$S(000, 1) = \{000, 100, 200, 010, 020, 001, 002\}.$$

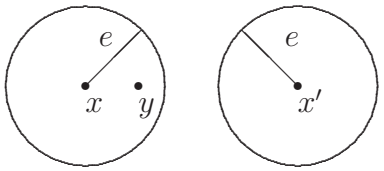
Theorem 1.19. *Let C be a code in $(\mathbf{F}_q)^n$.*

(i) *If $d(C) \geq s + 1$, then C can detect up to s errors.*

(ii) *If $d(C) \geq 2e + 1$, then C can correct up to e errors.*

Proof (i) Let $d(C) = s+1$. If $x \in C$ is sent and s mistakes occur in transmission, then the received vector cannot be a codeword. So the mistakes are detected.

(ii)



Let $d(C) = 2e + 1$.

If $x \in C$ is sent and y received with at most e errors, then $d(x, y) \leq e$. If $x' \in C$ with $x' \neq x$, then $d(x, x') \geq 2e + 1$.

Suppose that $d(x', y) \leq e$, then $d(x, x') \leq d(x, y) + d(x', y) \leq 2e$. Hence $d(x', y) \geq e+1$. So $y \to x$. Hence C can correct e errors. □

Corollary 1.20. *If C has minimum distance d, then it can detect $d - 1$ errors and correct $e = \lfloor (d - 1)/2 \rfloor$ errors, where $\lfloor m \rfloor$ denotes the integer part of m:*

d	1	2	3	4	5	6	7	8
e	0	0	1	1	2	2	3	3

Definition 1.21. The *q-ary repetition code of length n* on $\mathbf{F}_q = \{\lambda_1, ..., \lambda_q\}$ is

$$\{\lambda_1 \ldots \lambda_1,\ \lambda_2 \ldots \lambda_2,\ \ldots,\ \lambda_q \ldots \lambda_q\}.$$

This is an (n, q, n) code.

Example 1.22. (1) To send back photographs from the 1972 Mariner to Mars, a binary $(32, 64, 16)$ code was used. Here, $32 = 6 + 26$, with 6 information symbols and 26 redundancy symbols. So each part of each photograph was coded in one of $2^6 = 64$ shades of grey; 7 errors for each part could be corrected.

(2) For the 1979 Voyager spaceship to Jupiter, a binary $(24, 64^2, 8)$ code was used. This time, $24 = 12 + 12$, with 12 information symbols and 12 redundancy symbols. So each part of each photograph was coded in one of $2^{12} = 64^2 = 4096$ shades to send back colour photographs, with 3 errors able to be corrected.

Example 1.23 (Morse Code as used by the British Army)**.**

$$0 = .$$
$$1 = -$$

A	01	W	011
B	1000	X	1001
C	1010	Y	1011
D	100	Z	1100
E	0	0	1
F	0010	1	01
G	110	2	001
H	0000	3	00011
I	00	4	00001
J	0111	5	0
K	101	6	1000
L	0100	7	11000
M	11	8	100
N	10	9	10
O	111		
P	0110		
Q	1101		
R	010		
S	000		
T	1		
U	001		
V	0001		