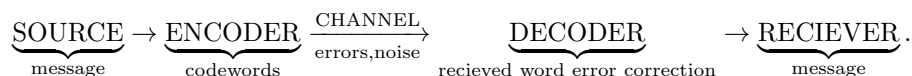


Introduction

We model communication:



Examples: optical signals, electrical telegraph, SMS (compression), postcodes, CDs (error correction), zip/gz files (compression).

Given a source and a channel, modelled probabilistically, the basic problem is to design an encoder and decoder to transmit messages economically (noiseless coding; compression) and reliably (noisy coding).

Examples:

- Noiseless coding: Morse code: common letters are assigned shorter code-words, e.g $A \mapsto \bullet-$, $E \mapsto \bullet$, $Q \mapsto --\bullet-$, $S \mapsto \bullet\bullet\bullet$, $O \mapsto --$, $Z \mapsto --\bullet\bullet$. Noiseless coding is adapted to source.
- Noisy coding: Every book has an ISBN $a_1, a_2, \dots, a_9, a_{10}$, $a_i \in \{0, 1, \dots, 9\}$ for $1 \leq i \leq 9$ and $a_{10} \in \{0, 1, \dots, 9, X\}$ with $\sum_{j=1}^{10} ja_j \equiv 0 \pmod{11}$. This detects common errors - e.g one incorrect digit, transposition of two digits. Noisy coding is adapted to the channel.

Plan:

- (I) Noiseless coding - entropy
- (II) Error correcting codes - noisy channels
- (III) Information theory - Shannon's theorems
- (IV) Examples of codes
- (V) Cryptography

Books: [GP], [W], [CT], [TW], Buchmann, Körner. Online notes: Carne, Körner.

Basic Definitions

Definition (Communication channel). A *communication channel* accepts symbols from a alphabet $\mathcal{A} = \{a_1, \dots, a_r\}$ and it outputs symbols from alphabet $\mathcal{B} = \{b_1, \dots, b_s\}$. Channel modelled by the probabilities $\mathbb{P}(y_1 \dots y_n \text{ recieved} | x_1 \dots x_n \text{ sent})$. A *discrete memoryless channel* (DMC) is a channel with

$$p_{ij} = \mathbb{P}(b_j \text{ recieved} | a_i \text{ sent})$$

the same for each channel use and independent of all past and future uses. The channel matrix is $P = (b_{ij})$, a $r \times s$ stochastic matrix.

Definition (Binary symmetric channel). The *binary symmetric channel* (BSC) with error probability $p \in [0, 1)$ from $\mathcal{A} = \mathcal{B} = \{0, 1\}$. The channel matrix is

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

A symbol is transmitted correctly with probability $1 - p$. Usually assume $p < 1/2$.

The *binary erasure channel* (BEC) has $\mathcal{A} = \{0, 1\}$, $\mathcal{B} = \{0, 1, *\}$. The channel matrix is

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

So $p = \mathbb{P}(\text{symbol can't be read})$.

Definition. We model n uses of a channel by the n th extension, with input alphabet \mathcal{A}^n and output alphabet \mathcal{B}^n . A *code* C of length n is a function $\mathcal{M} \rightarrow \mathcal{A}^n$ where \mathcal{M} is the set of possible messages. Implicitly we also have a decoding rule $\mathcal{B}^n \rightarrow \mathcal{M}$. The *size* of C is $m = |\mathcal{M}|$. The *information rate* is $\rho(C) = \frac{1}{n} \log_2 m$. The *error rate* is $\hat{e}(C) = \max_{x \in \mathcal{M}} \mathbb{P}(\text{error} | x \text{ sent})$.

Remark. For the remainder of the course we write \log instead of \log_2 .

Definition. A channel can *transmit reliably at rate* R if there exists $(C_n)_{n=1}^\infty$ with each C_n a code of length n such that

$$\lim_{n \rightarrow \infty} \rho(C_n) = R \text{ \& } \lim_{n \rightarrow \infty} \hat{e}(C_n) = 0.$$

The *capacity* is the supremum of all reliable transmission rates. We'll see in Chapter 9 that a BSC with error probability $p < 1/2$ has non-zero capacity.

1 Noiseless coding

1.1 Prefix-free codes

For an alphabet \mathcal{A} , $|\mathcal{A}| < \infty$, let $\mathcal{A}^* = \bigcup_{n \geq 0} \mathcal{A}^n$, the set of all finite strings from \mathcal{A} . The *concatenation* of strings $x = x_1 \dots x_r$ and $y = y_1 \dots y_s$ is $xy = x_1 \dots x_r y_1 \dots y_s$.

Definition. Let \mathcal{A}, \mathcal{B} be alphabets. A code is a function $c : \mathcal{A} \rightarrow \mathcal{B}^*$. The strings $c(a)$ for $a \in \mathcal{A}$ are called *codewords* or *words* (CWS).

Example 1.1 (Greek fire code). $\mathcal{A} = \{\alpha, \beta, \dots, \omega\}$ (greek alphabet), $\mathcal{B} = \{1, 2, 3, 4, 5\}$, $c : \alpha \mapsto 11, \beta \mapsto 12, \dots, \psi \mapsto 53, \omega \mapsto 54$. xy means hold up x torches and another y torches nearby.

Example 1.2. \mathcal{A} = words in a dictionary, $\mathcal{B} = \{A, B, \dots, Z, \omega\}$. $c : \mathcal{A} \rightarrow \mathcal{B}$ splits the word and follows with a space. Send message $x_1 \dots x_n \in \mathcal{A}^*$ as $c(x_1) \dots c(x_n) \in \mathcal{B}^*$. So c extends to a function $c^* : \mathcal{A}^* \rightarrow \mathcal{B}^*$.

Definition. c is said to be *decipherable* if the induced map c^* (as in the previous example) is injective. In other words, each string from \mathcal{B} corresponds to at most one message.

Clearly if c is decipherable, it is necessary for c to be injective. However it is not sufficient:

Example 1.3. $\mathcal{A} = \{1, 2, 3, 4\}$, $\mathcal{B} = \{0, 1\}$. Define $c : 1 \mapsto 0, 2 \mapsto 1, 3 \mapsto 00, 4 \mapsto 01$. Then $c^*(114) = 0001 = c^*(312) = c^*(144)$ yet c is injective.

Notation: $|\mathcal{A}| = m$, $|\mathcal{B}| = a$, call c an a -ary code of size m . For example a 2-ary code is a binary one, and a 3-ary code is a ternary code.

Our aim is to construct decipherable codes with short word lengths. Assuming c is injective, the following codes are always decipherable:

- (i) A block code has all codewords of the same length (e.g Greek fire code);
- (ii) A comma code reserves a letter from \mathcal{B} to signal the end of a word (e.g Example 1.2);
- (iii) A prefix-free code is a code where no codeword is a prefix of any other distinct word (if $x, y \in \mathcal{B}^*$ then x is a prefix of y if $y = xz$ for some string $z \in \mathcal{B}^*$).

(i) and (ii) are special cases of (iii). As we can decode the message as it is recieved, prefix-free codes are sometimes called *instantaneous*.

Exercise: find a decipherable code which is not prefix-free.

Definition (Kraft's inequality). $|\mathcal{A}| = m$, $|\mathcal{B}| = a$, $c : \mathcal{A} \rightarrow \mathcal{B}^*$ has word lengths l_1, \dots, l_m . Then Kraft's inequality is

$$\sum_{i=1}^m a^{-l_i} \leq 1. \quad (*)$$

Theorem 1.1. A prefix-free code exists if and only if Kraft's inequality $(*)$ holds.

Proof. Rewrite $(*)$ as

$$\sum_{l=1}^s n_l a^{-l} \leq 1, \quad (**)$$

where n_l is the number of codewords with length l , and $s = \max_{1 \leq i \leq m} l_i$.

Now if $c : \mathcal{A} \rightarrow \mathcal{B}^*$ is prefix-free,

$$n_1 a^{s-1} + n_2 a^{s-2} + \dots + n_{s-1} a + n_s \leq a^s.$$

Indeed the LHS is the number of strings of length s in B with some codeword of c as a prefix, and the RHS is the total number of strings of length S . Dividing through by a^s we get (**).

Now given n_1, \dots, n_s satisfying (**), we try to construct a prefix-free code c with n_l codewords of length l , $\forall l \leq s$. Proceed by induction on s , $s = 1$ is clear (since (**) gives $n_1 \leq a$ so can construct code).

By the induction hypothesis, there exists a prefix-code \hat{c} with n_l codewords of length l for all $l \leq s - 1$. Then (**) implies

$$n_1 a^{s-1} + n_2 a^{s-2} + \dots + n_{s-1} a + n_s \leq a^s.$$

The first $s - 1$ terms on the LHS sum to the number of strings of length s with a codeword of \hat{c} as a prefix and the RHS is the number of strings of length s . Hence we can add at least n_s new codewords of length s to \hat{c} and maintain the prefix-free property. □

Remark. This proof is constructive: just choose codewords in order of increasing length, ensuring that no previous codeword is a prefix.

Theorem 1.2 (McMillan). *Any decipherable code satisfies Kraft's inequality.*

Proof (Karush, 1961). Let $c : \mathcal{A} \rightarrow \mathcal{B}^*$ be a decipherable code with word lengths l_1, \dots, l_m . Set $s = \max_{1 \leq i \leq m} l_i$. For $R \in \mathbb{N}$

$$\left(\sum_{i=1}^m a^{-l_i} \right)^R = \sum_{l=1}^{Rs} b_l a^{-l}, \quad (\dagger)$$

where b_l is the number of ways of choosing R codewords of total length l . Since c is decipherable, any string of length l formed from codewords must correspond to at most one sequence of codewords, i.e $b_l \leq |\mathcal{B}^l| = a^l$. Subbing this into (†)

$$\left(\sum_{i=1}^m a^{-l_i} \right)^R \leq \sum_{l=1}^{Rs} a^l a^{-l} = Rs,$$

so

$$\sum_{i=1}^m a^{-l_i} \leq (Rs)^{1/R} \rightarrow 1 \text{ as } R \rightarrow \infty.$$

Hence $\sum_{i=1}^m a^{-l_i} \leq 1$. □

Corollary 1.3. *A decipherable code with prescribed word lengths exists if and only if a prefix-free code with the same word lengths exists.*

Proof. Combine previous two theorems. □

Therefore we can restrict our attention to prefix-free codes.

2 Shannon's Noiseless Coding Theorem

Entropy is a measure of 'randomness' or 'uncertainty'. Suppose we have a random variable X taking a finite set of values x_1, \dots, x_n with probabilities p_1, \dots, p_n respectively. The *entropy* $H(X)$ of X is the expected number of fair coin tosses needed to simulate X (roughly speaking).

Example 2.1. Suppose $p_1 = p_2 = p_3 = p_4 = 1/4$. Identify (x_1, x_2, x_3, x_4) with (HH, HT, TH, TT) . Then the entropy is 2.

Example 2.2. Suppose $(p_1, p_2, p_3, p_4) = (1/2, 1/4, 1/8, 1/8)$. Identify (x_1, x_2, x_3, x_4) with (H, TH, TTH, TTT) . Then the entropy is

$$\frac{1}{2} \times 1 + \frac{1}{4} \times 2 + \frac{1}{8} \times 3 + \frac{1}{8} \times 3 = \frac{7}{4}.$$

In a sense, the previous example (2.1) was 'more random' than this.

Definition (Entropy). The *entropy* of X is

$$H(X) = - \sum_{i=1}^b p_i \log p_i.$$

(Recall that $\log =: \log_2$ here.) Note $H(X) \geq 0$. It is measured in *bits* (binary digits). Conventionally, we take $0 \log 0 = 0$.

Example 2.3. Take a biased coin $\mathbb{P}(H) = p, \mathbb{P}(T) = 1 - p$. Write $H(p, 1 - p) := H(p)$. Then

$$H(p) = -p \log p - (1 - p) \log(1 - p).$$

Note that $H'(p) = \log \frac{1-p}{p}$. Hence the entropy is maximised for $p = 1/2$ (giving entropy 1).

Proposition 2.1 (Gibbs' inequality). *Let $(p_1, \dots, p_n), (q_1, \dots, q_n)$ be probability distributions. Then*

$$- \sum_{i=1}^n p_i \log p_i \leq - \sum_{i=1}^n p_i \log q_i.$$

(The RHS is sometimes called the *cross entropy* or *mixed entropy*) Furthermore we have equality iff $p_i = q_i$ for all i .

Proof. Since $\log x = \frac{\ln x}{\ln 2}$, we may replace \log with \ln . Put $I = \{1 \leq i \leq n : p_i \neq 0\}$. Now $\ln x = x - 1$ for all $x > 0$ with equality iff $x = 1$. Hence $\ln \frac{q_i}{p_i} \leq \frac{q_i}{p_i} - 1$ for all $i \in I$. So

$$\begin{aligned} \sum_{i \in I} p_i \ln \frac{q_i}{p_i} &\leq \underbrace{\sum_{i \in I} q_i}_{\leq 1} - \underbrace{\sum_{i \in I} p_i}_{=1} \leq 0 \\ \implies - \sum_{i \in I} p_i \ln p_i &\leq - \sum_{i \in I} p_i \ln q_i \end{aligned}$$

$$\implies -\sum_{i=1}^n p_i \ln p_i \leq -\sum_{i=1}^n p_i \ln q_i.$$

If equality holds, then $\sum_{i \in I} q_i = 1$ and $\frac{p_i}{q_i} = 1$ for all $i \in I$. So $q_i = p_i$ for all $1 \leq i \leq n$. \square

Corollary 2.2. $H(p_1, p_2, \dots, p_n) \leq \log n$ with equality iff $p_1 = p_2 = \dots = p_n = 1/n$.

Proof. Take $q_1 = q_2 = \dots = q_n = 1/n$ in Gibbs' inequality. \square

Let $\mathcal{A} = \{\mu_1, \dots, \mu_m\}$, $|\mathcal{B}| = a$ ($m, n \geq 2$). The random variable X takes values μ_1, \dots, μ_m with probabilities p_1, \dots, p_m .

Definition. If $c : \mathcal{A} \rightarrow \mathcal{B}^*$ is a code, we say it is *optimal* if it has the smallest possible expected word length. i.e. $\mathbb{E}S := \sum_{i=1}^m p_i l_i$ is minimal amongst all decipherable codes.

Theorem 2.3 (Shannon's Noiseless Coding Theorem). *The expected word length $\mathbb{E}S$ of an optimal code satisfies*

$$\frac{H(X)}{\log a} \leq \mathbb{E}S < \frac{H(X)}{\log a} + 1.$$

Remark. The lower bound is actually true for any decipherable code.

Proof. We first get the lower bound. Let $c : \mathcal{A} \rightarrow \mathcal{B}^*$ be decipherable with word lengths l_1, \dots, l_m . Let $q_i = \frac{a^{-l_i}}{D}$ where $D = \sum_{i=1}^m a^{-l_i}$. Note $\sum_{i=1}^m q_i = 1$. By Gibbs' inequality

$$\begin{aligned} H(X) &\leq -\sum_{i=1}^m p_i \log q_i \\ &= -\sum_{i=1}^m p_i (-l_i \log a - \log D) \\ &= \left(\sum_{i=1}^m p_i l_i \right) \log a + \log D. \end{aligned}$$

By McMillan, $D \leq 1$ so $\log D \leq 0$. Hence

$$H(X) \leq \left(\sum_{i=1}^m p_i l_i \right) \log a \implies \frac{H(X)}{\log a} \leq \mathbb{E}S.$$

And we have equality iff $p_i = a^{-l_i}$ for some integers l_1, \dots, l_m . Note we have only used decipherability so far.

Now we get the upper bound. Take $l_i = \lceil -\log_a p_i \rceil$. Then

$$-\log_a p_i \leq l_i < -\log_a p_i + 1.$$

Hence $\log_a p_i \geq -l_i$, so $p_i \geq a^{-l_i}$. Therefore $\sum_{i=1}^m a^{-l_i} \leq \sum_{i=1}^m p_i = 1$. By Kraft's inequality, there exists a prefix-free code c with word lengths l_1, \dots, l_m . c has expected word length

$$\mathbb{E}S = \sum_{i=1}^m p_i l_i < \sum_{i=1}^m p_i (-\log_a p_i + 1) = \frac{H(X)}{\log a} + 1.$$

□

Example 2.4 (Shannon-Fano Coding). We mimic the above proof: given p_1, \dots, p_m , set $l_i = \lceil -\log_a p_i \rceil$. Construct a prefix-free code with word lengths l_i by choosing codewords in order of increasing length, ensuring any new codeword has no previous codeword as a prefix (Kraft's inequality ensures we can do this).

Example 2.5. Take $a = 2, m = 5$.

i	p_i	$\lceil -\log_2 p_i \rceil$	code
1	0.4	2	00
2	0.2	3	010
3	0.2	3	011
4	0.1	4	1000
5	0.1	4	1001

Then $\mathbb{E}S = \sum_{i=1}^m p_i l_i = 2.8$, $H = H/\log a = 2.12$. [See also Carne p13.]

3 Huffman Coding

How to construct an optimal code? Take $\mathcal{A} = \{\mu_1, \dots, \mu_m\}$, $p_i = \mathbb{P}(X = \mu_i)$. For simplicity take $|\mathcal{B}| = a = 2$. Without loss of generality $p_1 \geq p_2 \geq \dots \geq p_m$. Huffman gave an inductive definition of codes that we can prove are optimal. If $m = 2$, we take codewords 0, 1. If $m > 2$, first take the Huffman code for messages $\mu_1, \dots, \mu_{m-2}, \nu$ with probabilities $p_1, \dots, p_{m-2}, p_{m-1} + p_m$. Then append 0 (respectively 1) to the codeword for ν to give a codeword for μ_{m-1} (respectively μ_m).

Notes.

- Huffman codes are prefix-free;
- Huffman codes are not unique: choice is needed if some of the p_i are equal.

Example 3.1. Revisit Example 2.5. We have

i	p_i	$c^{(1)}$	$p_i^{(2)}$	$c^{(2)}$	$p_i^{(3)}$	$c^{(3)}$	$p_i^{(4)}$	$c^{(4)}$
1	0.4	1	0.4	1	0.4	1	0.6	0
2	0.2	01	0.2	01	0.4	00	0.4	1
3	0.2	000	0.2	000	0.2	01		
4	0.1	0010	0.2	001				
5	0.1	0011						

Theorem 3.1. *Huffman codes are optimal (Huffman, 1952).*

Proof. We show by induction on m that Huffman codes of size $m = |\mathcal{A}|$ are optimal.

$m = 2$: codewords are 0, 1 - clearly optimal.

$m > 2$: let c_m be a Huffman code for X_m , which takes values μ_1, \dots, μ_m with probabilities $p_1 \geq p_2 \geq \dots \geq p_m$; each c_m is constructed from Huffman code c_{m-1} for X_{m-1} which takes values $\mu_1, \dots, \mu_{m-2}, \nu$ with probabilities $p_1, \dots, p_{m-2}, p_{m-1} + p_m$. Then the expected word length is

$$\mathbb{E}S_m = \mathbb{E}S_{m-1} + p_{m-1} + p_m. \quad (*)$$

Let c'_m be an optimal code for X_m . Wlog c'_m is still prefix-free. Wlog the last two codewords of c'_m have maximal length and differ only in the final position (see next lemma). Say

$$c'_m(\mu_{m-1}) = y0, \quad c'_m(\mu_m) = y1 \text{ for some } y \in \{0, 1\}^*.$$

Let c'_{m-1} be some prefix-free code for X_{m-1} , given by

$$c'_{m-1}(\mu_i) = \begin{cases} c'_m(\mu_i) & 1 \leq i \leq m-2 \\ c'_{m-1}(\nu) = y & \end{cases}.$$

Then the expected word length satisfies

$$\mathbb{E}S'_m = \mathbb{E}S'_{m-1} + p_{m-1} + p_m. \quad (**)$$

By the inductive hypothesis, c_{m-1} is optimal, so $\mathbb{E}S_{m-1} \leq \mathbb{E}S'_{m-1}$. By (*) and (**) this implies $\mathbb{E}S_m \leq \mathbb{E}S'_m$. \square

Lemma 3.2. *Suppose letters μ_1, \dots, μ_m in \mathcal{A} are sent with probabilities p_1, p_2, \dots, p_m . Let c be an optimal (prefix-free) code with word lengths l_1, \dots, l_m . Then*

- (i) *If $p_i > p + j$, then $l_i \leq l_j$;*
- (ii) *Amongst all codewords of maximal length there exist two that differ only in the final digit.*

Proof. (i) is obvious. For (ii), could otherwise just delete the final digit of the codeword of maximal length (since prefix-free). \square

Remark. Note not all optimal codes are Huffman (look at the case $m = 4$).

Our main result says that if we have a prefix-free optimal code with word lengths l_1, \dots, l_m and associated probabilities p_1, \dots, p_m , then there is a Huffman code with these word lengths.

4 Joint Entropy

If X, Y are random variables with values in \mathcal{A} and \mathcal{B} respectively, then (X, Y) is a random variable with values in $\mathcal{A} \times \mathcal{B}$, and the *entropy* $H(X, Y)$ is called the joint entropy, given by

$$H(X, Y) = - \sum_{x \in \mathcal{A}} \sum_{y \in \mathcal{B}} \mathbb{P}(X = x, Y = y) \log \mathbb{P}(X = x, Y = y).$$

This generalises to any finite number of random variables.

Lemma 4.1. *Let X, Y be random variables taking values in \mathcal{A} and \mathcal{B} respectively. Then*

$$H(X, Y) \leq H(X) + H(Y),$$

with equality if and only if X and Y are independent.

Proof. Write $\mathcal{A} = \{x_1, \dots, x_m\}$, $\mathcal{B} = \{y_1, \dots, y_n\}$. Let

$$p_{ij} = \mathbb{P}(X = x_i, Y = y_j), \quad p_i = \mathbb{P}(X = x_i), \quad q_j = \mathbb{P}(Y = y_j).$$

Apply Gibbs' inequality to the probability distributions $\{p_{ij}\}$ and $\{p_i q_j\}$ to obtain

$$\begin{aligned} - \sum_{i,j} p_{ij} \log p_{ij} &\leq - \sum_{i,j} p_{ij} \log(p_i q_j) \\ &= - \sum_i \left(\sum_j p_{ij} \right) \log p_i - \sum_j \left(\sum_i p_{ij} \right) \log q_j \\ &= - \sum_i p_i \log p_i - \sum_j q_j \log q_j \\ &= H(X) + H(Y). \end{aligned}$$

With equality if and only if $p_{ij} = p_i q_j$ for all i, j .

□

Error-correcting codes

5 Noisy channels and Hamming's code

Definition. A *binary* $[m, n]$ -code is a subset C of $\{0, 1\}^n$ of size $m = |C|$. n is the length of the code and the elements of C are called codewords.

We use an $[n, m]$ -code to send one of m messages through a BSC (binary symmetric channel) making n uses of the channel. Clearly $1 \leq m \leq 2^n$, so $0 \leq \frac{1}{n} \log m \leq 1$.

Definition. For any $x, y \in \{0, 1\}^n$ the *Hamming distance* is

$$d(x, y) = |\{i : 1 \leq i \leq n, x_i \neq y_i\}|.$$

Definition.

- (i) The *ideal observer* decoding rule decodes $x \in \{0, 1\}^n$ as $c \in C$ maximising $\mathbb{P}(c \text{ sent} | x \text{ recieved})$.
- (ii) The *maximum likelihood* decoding rule decodes $x \in \{0, 1\}^n$ as $c \in C$ maximising $\mathbb{P}(x \text{ recieved} | c \text{ sent})$
- (iii) The *minimum distance* decoding rule decodes $x \in \{0, 1\}^n$ as $c \in C$ minimising $d(x, C)$.

Lemma 5.1.

- (a) If all the messages are equally likely, then (i) and (ii) above are equivalent.
- (b) If $p < 1/2$ (error probability) then (ii) and (iii) are equivalent.

Remark. If $p = 1/2$ the code is called *useless*. If $p = 0$ the code is called *lossless*.

Proof.

- (a) We have

$$\mathbb{P}(c \text{ sent} | x \text{ recieved}) = \frac{\mathbb{P}(c \text{ sent}, x \text{ recieved})}{\mathbb{P}(x \text{ recieved})} = \frac{\mathbb{P}(c \text{ sent})}{\mathbb{P}(x \text{ recieved})} \mathbb{P}(x \text{ recieved} | c \text{ sent}).$$

So by hypothesis, $\mathbb{P}(c \text{ sent})$ is independent of $c \in C$. So for fixed x , maximising $\mathbb{P}(c \text{ sent} | x \text{ recieved})$ is the same as maximising $\mathbb{P}(x \text{ recieved} | c \text{ sent})$.

- (b) Let $r = d(x, c)$. Then $\mathbb{P}(x \text{ recieved} | c \text{ sent}) = p^r (1-p)^{n-r} = (1-p)^n \left(\frac{p}{1-p}\right)^r$. Since $p < 1/2$, $\frac{p}{1-p} < 1$. So maximising $\mathbb{P}(x \text{ recieved} | c \text{ sent})$ is the same as minimising r .

□

We choose to use minimum distance decoding from now on.

Example 5.1. Suppose 000, 111 are sent with probabilities $\alpha = 9/10$, $\beta = 1/10$ respectively through a BSC with error probability $p = 1/4$. Suppose 110 is recieved. Then

$$\mathbb{P}(000 \text{ sent} | 110 \text{ recieved}) = \frac{\alpha p^2 (1-p)}{\alpha p^2 (1-p) + (1-\alpha) p (1-p)^2} = \frac{3}{4},$$

$$\text{similarly } \mathbb{P}(111 \text{ sent} | 110 \text{ recieved}) = \frac{1}{4}.$$

So the ideal observer decodes as 000. But the maximum likelihood/minimum distance rules decode as 111.

Remarks.

- Minimum distance decoding may be expensive in terms of time and storage if $|C|$ is large.
- Need to specify a convention in case there is no unique maximiser (e.g make a random choice, or request the message is sent again).

We aim to detect, or even correct errors.

Definition. A code C is

- *d-error* detecting if changing up to d digits in each codeword can never produce another codeword. In other words, each codeword is of Hamming distance greater than d from every other codeword.
- *e-error* correcting if knowing that $x \in \{0, 1\}^n$ differs from a codeword in at most e places we can deduce the codeword.

Examples.

- A *repetition code* of length n has codewords $\underbrace{00 \dots 0}_{n \text{ times}}, \underbrace{11 \dots 1}_{n \text{ times}}$. This is a $[n, 2]$ -code. It is $(n-1)$ -error detecting and $\lfloor \frac{n-1}{2} \rfloor$ -error correcting. But the information rate is only $1/n$.
- A *simple parity check code* or *paper tape code*: identify $\{0, 1\}$ with \mathbb{F}_2 and let $C = \{(x_1, \dots, x_n) \in \{0, 1\}^n : \sum_{i=1}^n x_i = 0\}$. This is a $[n, 2^{n-1}]$ -code, 1-error detecting but cannot correct errors. The information rate is $\frac{n-1}{n}$.
- Hamming's original code (1950): a 1-error correcting binary $[7, 16]$ -code. Take $C \subseteq \mathbb{F}_2^7$ where

$$C = \{c \in \mathbb{F}_2^7 : c_1 + c_3 + c_5 + c_7 = 0, c_2 + c_3 + c_6 + c_7 = 0, c_4 + c_5 + c_6 + c_7 = 0\}.$$

The bits c_3, c_5, c_6, c_7 are arbitrary and c_1, c_2, c_4 are forced (called the check digits) so $|C| = 2^4$. To decode: suppose we receive $x \in \mathbb{F}_2^7$. We form the *syndrome*: $z = z_x = (z_1, z_2, z_4) \in \mathbb{F}_2^3$ where

$$z_1 = x_1 + x_3 + x_5 + x_7$$

$$z_2 = x_2 + x_3 + x_6 + x_7$$

$$z_4 = x_4 + x_5 + x_6 + x_7.$$

If $x \in C$, then $z_x = (0, 0, 0)$. If $d(x, c) = 1$ for some $c \in C$, then place where x and c differ is given by $z_1 + 2z_2 + 4z_4$ (not mod 2). Check: if $x = c + e_i$ where e_i has all 0's except a 1 in the i th position, then $z_x = z_{e_i}$, so check for each $1 \leq i \leq 7$.