Chapter 4

Linear Block Error-Correcting Codes [ECC]

Definitions

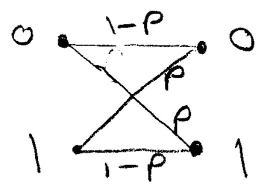
$$M \longrightarrow \boxed{ECC} \longrightarrow C$$

- The Error-Correcting Codes are functions that maps the input symbols M into a code alphabet C. In this course, we will focus on Binary source alphabets and in this case, the ECC is called Biary error-correcting codes.
- These codes are called linear since the encoder could be described by a linear function $\Rightarrow$ the codewords are linear combination of the inputs. And the mapping could be described by a Matrix called "Generator Matrix"

- Assume we have k information bits, $\bar{m} = (m_0, m_1, \cdots, m_{k-1})$
  The ECC encoder maps the block $\bar{m}$ into a codeword $\bar{c} = (c_0, c_1, \cdots, c_{n-1})$ of length n, with $n > k$.
- $\bar{m}$ is called the message and $\bar{c}$ is the codeword.
- The encoder adds $r = n - k$ redundant bits.
- As we saw in chapter one, the addition of redundant bits to the message bits doesn't change the entropy $\Rightarrow$ information lossless.
- However, this addition will reduce the entropy "information" rate.
  The entropy rate will be $R = \frac{k}{n} \Rightarrow$ Code Rate $\leq$ channel capacity

4.1.2 Error Rates and Error Distributions for the Binary Symmetric Channels.

Let the BSC be



where p is the bit error probability.

In a codeword of length n. If p is the bit probability, what is the probability of having t errors in one block of length n? Since the channel is memoryless, the bit errors are independent,

$$\Longrightarrow \quad P_r(t;p,n) = \binom{n}{t} p^t (1-p)^{n-t}$$

where $\binom{n}{t}$ (n choose t) $= \dfrac{n!}{(n-t)!\,t!} \Rightarrow$ Binomial Coefficient.

— The Probability that a block of n bits will have fewer than t errors is

$$P_r(<t) = \sum_{j=0}^{t-1} \binom{n}{j} p^j (1-p)^{n-j}$$

— The average number of errors in a block of n bits is

$$\bar{t} = \sum_{j=0}^{n} j \binom{n}{j} p^j (1-p)^{n-j} = np .$$

— The error variance is

$$\sigma_t^2 = E\{(t-\bar{t})^2\} = \sum_{j=0}^{n} (j-\bar{t})^2 \binom{n}{j} p^j (1-p)^{n-j}$$

$$= np(1-p) .$$

4.1.3 Error Detection and Correction

The goal of channel encoders are to detect and/or correct errors. Some codes can do both. Some are design for detection or correction only.

Example: Repetition Codes

$$G(0) \rightarrow 000, \quad G(1) \rightarrow 111 \quad \text{\} two code words}$$

Thus, for this code, $n = 3$, $k = 1$ and redundancy $r = 2$.

The code Rate $R = \frac{1}{3}$.

List all possible Receive code words

Received word	Error Detection: Decoded	Error Correction: Decoded
0 0 0	0	0
0 0 1	Error	0
0 1 0	Error	0
0 1 1	·	1
1 0 0	·	0
1 0 1	·	1
1 1 0	Error	1
1 1 1	1	1

Error Detection: this code can detect up to two errors.

Error Correction: This code can correct one error. Thus, it is a single-error correction code.

The decoding Rule is to select the codeword with closest distance to the received word.

- Hamming Distance:

We measure distance by the number of different bits between the codewords. For example, the distance between $C_0 = 000$ and Received word 011 is two bits.

We can make the repetition code with $n=4$ correct single-bit errors and detect two-bit errors. this is an error correction and detection capabilities. The rule is to decode received words with Hamming distance ≤ 1 and declare error for Hamming distance $=2$

Example

Received Word	Decode
0000	0
0001	0
0010	0
0011	Error
0100	0
0101	Error
0110	Error
0111	1
1000	0
1001	Error
1010	Error
1011	1
1100	Error
1101	1
1110	1
1111	1

~~The~~

— Hamming Distance and Code Capability

Let $\bar{v}$ be the received word,
Let $\bar{c}$ be the transmitted codeword.
The Hamming Distance is the number of different bits between $\bar{v}$ and $\bar{c}$. Denoted by

$$d_H(\bar{v}, \bar{c}) \longrightarrow \text{Hamming distance.}$$

So, we are receiving $\bar{v}$, the decoding Rule for BSC with independent errors is to select $\bar{c}_1$ such that

$$d_H(\bar{v}, \bar{c}_1) < d_H(\bar{v}, \bar{c}_i) \text{ for all } \bar{c}_i \in C$$

— The error detection and Correction Capabilities of codes are determined by the minimum Hamming Distance of the code.

Let G be an encoder that maps $\bar{m}_i \in M$ to $\bar{c}_i \in C$.
this mapping is one-to-one $\Rightarrow |M| = |C|$

$$\xRightarrow{\text{Mutual information}} I(M;C) = H(M)$$

- For every pair $\bar{c}_i, \bar{c}_j \in C \ (i \neq j)$, we can calculate a non-zero Hamming Distance $d_H(\bar{c}_i, \bar{c}_j)$.
- The Minimum Hamming Distance

$$d_{min} = \min_{\text{All codewords}} \{ d_H(\bar{c}_i, \bar{c}_j) \}$$

Now, the ability of the code is

① It can detect up to t errors
If and only if $d_{min} \geq t+1$

② It can correct up to t errors
If and only if $d_{min} \geq 2t+1$

③ It can correct up to t_c errors and detect up to $t_d > t_c$ errors if and only if
$d_{min} > 2t_c+1$ and $d_{min} \geq t_c+t_d+1$

Looking at it on the other direction,
If a code has a minimum Hamming Distance of d_{min},
then
① it can detect up to $t \leq d_{min}-1$ errors
② it can correct up to $t \leq \left\lfloor \frac{d_{min}-1}{2} \right\rfloor$ errors
③ it can correct $t_c \leq \left\lfloor \frac{d_{min}-1}{2} \right\rfloor$ and detect $t_d \leq d_{min}-t_c-1$

Lower Floor.

Example Repitition code with $n=4$ and $r=3$ redundant bits.

$d_{min} = d_H(0000, 1111) = 4$

$\Rightarrow$ can correct $t_c = \left\lfloor \frac{4-1}{2} \right\rfloor = 1$ error

can detect $t_d = 4-1-1 = 2$ errors

Singleton bound

The minimum Hamming distance of a code is upperbounded by the releation

$$d_{min} \leq r+1 \Rightarrow r_{min} = d_{min} - 1$$

where r is the number of redundant bits.

Example 4.1.6

Design Example

Suppose we wish to transmit seven-bit code words over a BSC with crossover probabilitiy $p = 0.05$, and we wish the Prob. of error in a block at the receiving end to be less than 10^{-3}. what is the maximum possible code rate we could achieve?

Assume that the seven-bit code that we like to design can correct up to t_c errors. The probability of having an uncorrectable error is

$$P_u = \sum_{j=t_c+1}^{7} \binom{7}{j} p^j (1-p)^{7-j} < 10^{-3}$$

By ploting this function for different t_c, we find that

$t_c = 2 \Rightarrow P_u = 0.0038 > 10^{-3}$

$t_c = 3 \Rightarrow P_u = 1.936 \times 10^{-4} < 10^{-3}$

$\therefore$ to achieve this goal, we need to correct three errors. The minimum Hamming distance will be

$$d_{min} \geq 2(3)+1 = 7$$

So, we can use a repetition code with $n=7$

$G(0) \rightarrow 0000000$

$G(1) \rightarrow 1111111$

the rate of this code is $R_1 = \frac{1}{7}$ bits per channel use $= 0.143$

Assume we want to use the seven-bit repetition code to correct one bit and detect five errors.

$$\Rightarrow P_u = 1.05 \times 10^{-7}$$

and Assume that we ask for retransmission when we detect an error. Thus, the probability of retransmission is

$$P_{rx} = \sum_{j=t_c+1}^{t_d} \binom{7}{j} p^j (1-p)^{7-j} = 0.0444.$$

Retransmission slows down information rate per channel use. and on the average, we must send each block $\frac{1}{1-P_{rx}}$ times.

Thus, the rate for this code will be

$$R_2 = \frac{1}{7}(1-P_{rx}) = 0.1365 \text{ bit per channel use}$$

The capacity and cutoff rate are

$$C_c = 0.7136 \text{ and } R_0 = 0.4781$$

$\therefore$ R_1 and R_2 are much less than R_0.

4.2 Binary Fields and Binary Vector Spaces

Binary Field.

To describe codes, we use a mathematical structure called a field.

A field is defined to be a set of elements A and two arithmetic operations called addition (+) and multiplication (·), such that,

1- Closure

for all $a, b \in A \Rightarrow a+b \in A$ and $a \cdot b \in A$

2- Associative properties of addition and multiplication

$$a+b+c = (a+b)+c = a+(b+c)$$

and $$a \cdot b \cdot c = (a \cdot b) \cdot c = a \cdot (b \cdot c)$$

3- Identity elements:

$a + 0 = 0 + a = a$; 0 is the additive identity

and $a \cdot 1 = 1 \cdot a = a$; 1 is the multiplicative identity

4- Additive inverse:

for any $a \in A$, there exist b such that

$a + b = 0 \Rightarrow b$ is the additive inverse.

$b \Rightarrow -a$

5- Multiplicative Inverses:

For any element $a \in A$, except 0, there is a multiplicative inverse of a such that $a \cdot b = 1$

$b \Rightarrow a^{-1}$

6- Addition is commutative : $a+b = b+a$

7- Distributive property: $a \cdot (b+c) = a \cdot b + a \cdot c$

$(a+b) \cdot c = a \cdot c + b \cdot c$

- Boolean arithmetic

For Binary Source, $A=\{0,1\}$

- the Field is called Galois field with two elements, $GF(2)$.

- Addition and multiplication are defined as:

$$0+0=0;\quad 0+1=1+0=1;\quad 1+1=0$$

$$0.1=1.0=0.0=0;\quad 1.1=1$$

$\Rightarrow$ Addition is the XOR "Exclusive or"

Multiplication is the AND function

Binary Vectors

- We can construct Binary vectors from the binary Field $GF(2)$.
- Define A^n to be a set with elements $\bar{a}=(a_0,a_1,\ldots,a_{n-1})$ with each $a_i \in A=\{0,1\}$.
- Define two arithmetic operations

1- Vector addition: if $\bar{a},\bar{b} \in A^n$ then vector addition $(+)$ is defined as

$$\bar{a}+\bar{b} \equiv ((a_0+b_0),(a_1+b_1),\ldots,(a_{n-1}+b_{n-1}))$$

2- Scalar multiplication:

If $\bar{a} \in A^n$ and $b \in A$ is a binary scalar, then

$$b.\bar{a}=\bar{a}.b=(ba_0, ba_1, \ldots, ba_{n-1})$$

— Vector Space

A vector space is a structure made ~~A; vectors (A^n),~~
~~a set A~~ — Vectors A^n
— Scalers A
— and two arithmetic operations.

The sets and the arithmetic operations satisfy the following constraints:

1. Closure: For every $\bar{a}, \bar{b} \in A^n$, the sum $\bar{a} + \bar{b} \in A^n$
2- Addition is commutative
3- Addition is associative
4- A^n contains a vector $\bar{0}$ such that $\bar{a} + \bar{0} = \bar{a}$
5- Additive Inverses: For every $\bar{a} \in A^n$, there is some vector $\bar{b} \in A^n$ such that $\bar{a} + \bar{b} = \bar{0}$.
6- For every scalar $a \in A$ and every vector $\bar{b} \in A^n$ there is a vector $a\bar{b} \in A^n$ $\Rightarrow$ closure of scalar Multiplication
7- scalar multiplication is associative
8- Scalar multiplication is distributive with respect to Vector addition
9- scaler multiplication is distributive with respect to Scalar addition
10 - if $1 \in A$ is the scalar multiplicative identity, then for every $\bar{a} \in A^n$, $1\bar{a} = \bar{a}$

Example A 2-D Code

A 2-D code is a simple error correcting code with additional error detection capability. The rate in this example is $9/16$.

Let the 9 bit message be

$$\bar{m} = [m_0 m_1 \cdots m_8], \quad m_i \in \{0,1\}$$

Let m_8 be the first bit transmitted, then the codeword is

$$\bar{c} = [c_0 c_1 c_2 m_0 c_4 m_1 c_6 m_2 c_8 m_3 m_5 c_{12} m_6 m_7 m_8]$$

where

$$\begin{array}{ccc|c} m_8 & m_7 & m_6 & c_{12} \\ m_5 & m_4 & m_3 & c_8 \\ m_2 & m_1 & m_0 & c_2 \\ \hline c_6 & c_4 & c_1 & c_0 \end{array}$$

Parity bits.

Parity bits

where $c_{12} = m_8 + m_7 + m_6$, similarly c_1, c_2, c_4, c_6 and c_8.

$c_0 = m_8 + m_7 + \cdots + m_1 + m_0$

During transmission, bit errors may occur,

the received word is

$$\bar{r} = \bar{c} + \bar{e}$$

where $\bar{e} = [e_0 e_1 \cdots e_{15}]$; if $e_i = 1 \Rightarrow$ error.

$e_i = 0 \Rightarrow$ no error

At the receiver, we calculate seven error check bits, which are called <u>syndrome bits</u>.

v_{15}	v_{14}	v_{13}	v_{12}	s_6
v_{11}	v_{10}	v_9	v_8	s_5
v_7	v_5	v_3	v_2	s_4
v_6	v_4	v_1	v_0	
s_3	s_2	s_1		s_0

s_6, s_5, s_4: syndrom bits → $S_6 = v_{15} + v_{14} + v_{13} + v_{12}$, sum over their respective rows

s_3, s_2, s_1: syndrom bits, sum over columns

$S_0 = \sum_{i=0}^{15} v_i$

this code can correct single-bit errors, and can detect two-bit errors.

Notice that:

1. If no errors occur, all s_i are zero
2. if an odd number of errors occur, $S_0 = 1$
3. If a single error occurs in one of the m_i message bits, One of the row and one of the column syndrome bits will equal to 1. In this case, we can correct the erroneous bit by adding 1 to the v_i bit corresponding to the intersection of the row and column.
4. If a single error occurs in one of the parity bits (c_i), then, either one row or one column syndrome bit (but not both) will be equal to one and the S_0 bit will be 1. In this case, no correction is necessary because none of the message bits are in error.
5. If two errors occur, $S_0 = 0$, and multiple synbrome bits in a row or a column will be 1. ⇒ we can detect two errors. But we can't guarantee to correct these errors.

For example, assume the errors are in v_{14} and v_{13}, then S_2 and $S_1 = 1$ but $S_6 = 0$. So, we can't tell the error location. However, if v_{14} and v_3 are in error, then S_2 and $S_1 = 1$ and S_6 and $S_4 = 1$, in this case, we can know that v_4 and V_3 are in errors.

6) If the number of errors are greater than ~~t~~two;

Odd number $\Rightarrow$ we can't guarantee that we will not miscorrect the error.

Even number $\Rightarrow$ we can't guarantee to detect the error.