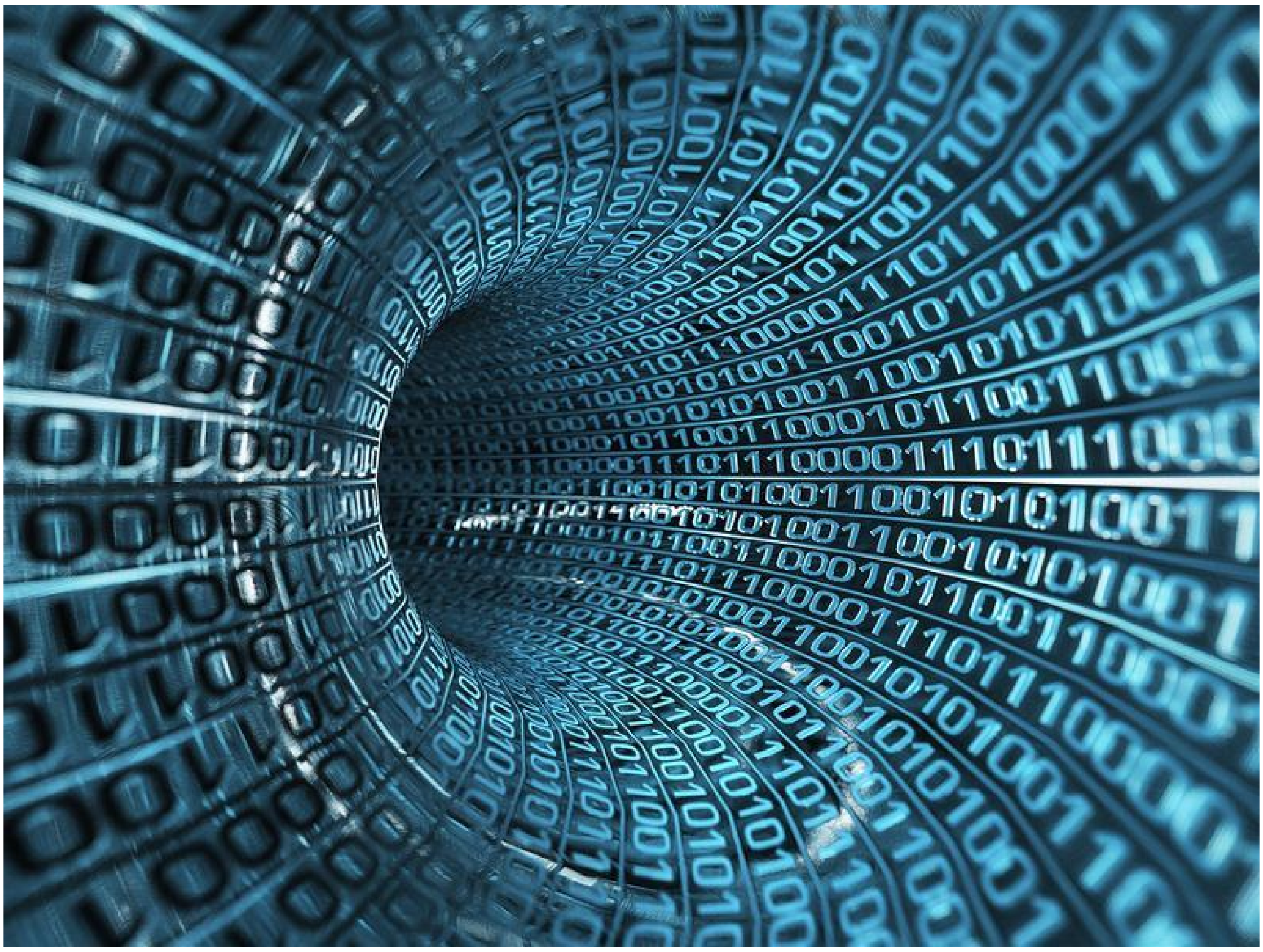
Summer of Science, 2020
MnP Club, IIT Bombay



Coding Theory

- Final Report -

Abhinav Gupta [1]
June 9, 2020

[1]Mentored by Shourya Pandey

Preface

In this report, we will try to understand how a message/data is sent across noisy channels. Messages sent through such channels undergo some aberration due to various kinds of noises. Our motto is to have a high probability of decoding a recieved message to its original meaning. This report goes through such encoding and decoding procedures, from fairly trivial to some advanced methods that enable us with fast and accurate decoding schemes. We will be following the text *A first Course in Coding Theory*[1] throughout this report, with some help from *Introoduction to Coding Theory* [2].

Contents

Notation

The following notations will be followed in this report. These are basic notations, same as those done in high school or above, by most of the people.

Sets

A *set* is a collection of objects. The following sets (among others) will be used in this report:
$\mathbb{R}$: the set of real numbers.
$\mathbb{Z}$: the set of integers (positive, negative, or zero).
$\mathbb{Z}_n$: $\{0, 1, 2, \ldots, n-1\}$

The symbols $\varnothing$, $\in$, $\notin$, $\cup$, $\cap$, $\subseteq$ and $\supseteq$ have their usual meanings. If S and T are sets and, $S \cap T = \varnothing$, then S and T are said to be *disjoint*.
If S is a set and P a property (or a combination of properties), we can define a new set with the notation

$$\{x \in S \mid P(x)\}$$

which denotes 'set of all elements of S which have property P'.
The *order* or *cardinality* of a finite set S is the number of elements in S and is denoted by $|S|$. For example, $|\mathbb{Z}_n| = n$.
The *Cartesian Prooduct* of two sets S and T is given by

$$S \times T = \{(s, t) \mid s \in S,\ t \in T\}.$$

If S and T are finite sets, then $|S \times T| = |S| \cdot |T|$.
In general,

$$S_1 \times S_2 \times \cdots \times S_n = \{(s_1, s_2, \ldots, s_n) \mid s_i \in S_i,\ i = 1, 2, \ldots, n\},$$

is the *Cartesian Product* (a set of *ordered n-tuples*) of n sets $S_1, S_2, \ldots, S_n$.
In this report, an ordered n-tuple $(x_1, x_2, \ldots, x_n)$ will be denoted simply as $x_1 x_2 \cdots x_n$.

Combinatorics

Number of ways of choosing m distinct objects from n distinct objects
or
the coefficient of x^m in $(1+x)^n$
are both given by

$$\binom{n}{m} = \frac{n!}{m!\,(n-m)!}$$

where $p! = p(p-1)\cdots 3.2.1$ for $m > 0$ and $0! = 1$.
This bracket notation will be used throughout the report.

A *permutation* of a set $S = \{x_1, x_2, \ldots, x_n\}$ is a one-to-one mapping from set S to itself. It is denoted by

$$\begin{pmatrix} x_1 & x_2 & \ldots & x_n \\ \downarrow & \downarrow & & \downarrow \\ f(x_1) & f(x_2) & \ldots & f(x_n) \end{pmatrix}$$

Modular Arithmatic

Let m be a fixed positve integer. Two integers a and b are written as

$$a \equiv b \pmod{m}$$

if $a - b$ is divisible by m.
It can be noted that if $a \equiv a' \pmod{m}$ and $b \equiv b' \pmod{m}$ then

(i) $a + b \equiv a' + b' \pmod{m}$
(ii) $ab \equiv a'b' \pmod{m}$

Fermat's Little Theorem: Let p be a prime, and a be any integer, then $a^p \equiv a \pmod{p}$

§1 Introduction to error-correcting codes

Error-correcting codes are used to reduce errors when data is transmitted in noisy communication channels, like a telephone line, a satellite communication link, etc. The objective of error-correcting codes is to add a certain amount of redundancy to the message, so that even if some errors occur during transmission, we have high probability to recover the original message.

Definition 1.1. A ***q-ary code*** is a given set of sequences of symbols where each symbol is chosen from a set $F_q = \{\lambda_1, \lambda_2, \dots, \lambda_q\}$ of q distinct elements. The set F_q is called the ***alphabet***.

In particular, the 2-ary codes are called *binary codes.*

Definition 1.2. A code in which every codeword is of fixed length n is called ***block code of length n***.

We will deal with such codes only, so from now on, 'code' will mean 'block code'. A code C with M codewords of length n is often written as $M \times n$ array. For example, the *binary repitition code* of length 3 is $\begin{bmatrix} 000 \\ 111 \end{bmatrix}$.

The elements of the set $(F_q)^n$ are called *words* or *vectors* and if C is a q-ary code of length n then $C \subseteq (F_q)^n$.

$$(F_q)^n = \{\mathbf{a} \mid \mathbf{a} = a_1 a_2 \cdots a_n,\ a_i \in F_q\}$$

Definition 1.3 (Hamming distance). The ***distance*** between two vectors $\mathbf{x}$ and $\mathbf{y}$ of $(F_q)^n$, denoted by $d(\mathbf{x}, \mathbf{y})$, is the number of places in which they differ.

Theorem 1.4. The Hamming distance satisfies:

(i) $d(\mathbf{x}, \mathbf{y}) = 0 \iff \mathbf{x} = \mathbf{y}$.
(ii) $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})\ \forall\ \mathbf{x}, \mathbf{y} \in (F_q)^n$.
(iii) $d(\mathbf{x}, \mathbf{y}) \leq d(\mathbf{x}, \mathbf{z}) + d(\mathbf{y}, \mathbf{z})\ \forall\ \mathbf{x}, \mathbf{y}, \mathbf{z} \in (F_q)^n$. (Triangle inequality)

Proof. First two statements are trivial. In third, we note that $d(\mathbf{x}, \mathbf{y})$ is the minimum number of changes required to change $\mathbf{x}$ to $\mathbf{y}$, but we can also do this change by first changing $\mathbf{x}$ to $\mathbf{z}$ in $d(\mathbf{x}, \mathbf{z})$ changes and then from $\mathbf{z}$ to $\mathbf{y}$ in $d(\mathbf{z}, \mathbf{y})$ changes. Thus, $d(\mathbf{x}, \mathbf{y}) \leq d(\mathbf{x}, \mathbf{z}) + d(\mathbf{y}, \mathbf{z})$. □

Suppose a codeword $\mathbf{x}$ is being transmitted, and we receive a distorted vector $\mathbf{y}$. If we decode it as $\mathbf{x}'$ such that $d(\mathbf{y}, \mathbf{x}')$ is minimum, then this decoding scheme is called ***nearest neighbour decoding***.

Definition 1.5. A transmitting channel is said to be ***q-ary symmetric channel*** if

1. Each symbol transmitted has the same probability $p(< \frac{1}{2})$ of being recieved in error.
2. If a symbol is received with error, then all $q - 1$ errors are equally likely.

Theorem 1.6. For a binary symmetric channel, *nearest neighbour decoding* is the *maximum likelihood decoding*, i.e. *nearest neighbour decoding* gives the highest probability of error detection in these types of channels.

Proof. The probability that codeword has errors at given i positions is $p^i(1-p)^{n-i}$, which decreases as i increases (as $p < \frac{1}{2}$), hence the least distant word is the most probable of being the original word. □

Definition 1.7. For a code C, we define ***minimum distance***, denoted by $d(C)$, as the smallest of all the distances between different codewords of C, i.e.

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

Theorem 1.8.

(i) A code C can detect upto s errors if $d(C) \geq s + 1$.

(ii) A code C can correct upto t errors if $d(C) \geq 2t + 1$.

Proof.

(i) Suppose $d(C) \geq s + 1$. Now if the recieved codeword has less than or equal to s errors, it will be a different codeword than any present in C. Hence, the error will be detected.

(ii) Suppose $d(C) \geq 2t+1$. Let the original codeword be $\mathbf{x}$ and received one be $\mathbf{y}$ with $d(\mathbf{x},\mathbf{y}) \leq t$. Let $\mathbf{x}$' be any codeword in C other than $\mathbf{x}$, then by triangle inequality,

$$d(\mathbf{x},\mathbf{x}') \leq d(\mathbf{x},\mathbf{y}) + d(\mathbf{x}',\mathbf{y}) \geq 2t+1$$

$$d(\mathbf{x}',\mathbf{y}) \geq t+1$$

Therefore, $\mathbf{x}$ is the nearest neighbour to $\mathbf{y}$, and nearest neighbour decoding corrects the error.

□

Corollary 1.8.1. If code C has minimum distance d then it can be used either:

(i) to detect upto $d-1$ errors; or

(ii) to correct upto $\left\lfloor \dfrac{d-1}{2} \right\rfloor$ errors in any coedword.
($\lfloor x \rfloor$ denotes the greatest integer leass than or equal to x)

Proof. Simple rearrangment of terms in the Theorem 1.8 will give the result. □

Notation: A (n, M, d)-*code* is a code with M words of length n, having minimum distance d.

§2 The main coding theory problem

A good (n, M, d)-code should allow:

1. fast transmission.
2. transmission of wide variety of messages.
3. to correct many errors.

These will be possible if we have small n, large M and d. But these are, quite intutively also, conflicting aims. We generally aim for maximizing M, given n and d.
We denote by $A_q(n, d)$ the largest value of M such that q-ary (n, M, d)-code exists.

Theorem 2.1. (i) $A_q(n, 1) = q^n$. (ii) $A_q(n, n) = q$.
Proof. (i) $d = 1$ reqiures the words to be *distinct* only. Therefore, $(n, M, 1)$-code $= (F_q)^n \implies M = A_q(n, 1) = q^n$
(ii) $d = n$ implies symbols appearing at any fixed position in the code must be all different $\implies A_q(n, d) \le q$. But there exists q-ary repetition code of length n, $\begin{bmatrix} 00\ldots0 \\ 11\ldots1 \\ \vdots \\ qq\ldots q \end{bmatrix}$, hence $A_q(n, n) = q$. □

Definition 2.2. Two q-ary codes are said to be ***equivalent*** if one can be obtained from another by combinations of operations of the types, i.e.(i) permuations of the positions of the code. (ii) permutations of the symbols appearing at a fixed position.

The second point means assigning a permutation function f on the *alphabet*, and applying f in a particular column of the code. Distances between operators remain same in these operations, so *equivalent* codes have the same parameteres (n, M, d) and therefore, will correct the same number of errors.

Lemma 2.3. Any q-ary (n, m, d)-code is equivalent to a (n, M, d)-code containing the vector $\mathbf{0} = 00\ldots0$.

Proof. For any codeword $\mathbf{x} = x_1, x_2 \cdots x_n$ in the code, for each $x_i \neq 0$ applying the permutation

$$\begin{pmatrix} 0 & x_i & j & \\ \downarrow & \downarrow & \downarrow & \forall\, j \neq 0, x_i \\ x_i & 0 & j & \end{pmatrix}$$

to the symbols of position i, will give the desired (n, M, d)-code. □

Now taking F_2 to be the set $\{0, 1\}$. Let $\mathbf{x} = x_1x_2\cdots x_n$, $\mathbf{y} = y_1y_2\cdots y_n \in (F_2)^n$, we define the following operations, namely, the ***sum*** $\mathbf{x} + \mathbf{y}$ is the vector in $(F_2)^n$ defined as

$$\mathbf{x} + \mathbf{y} = (x_1 + y_1, x_2 + y_2, \ldots, x_n + y_n)$$

while the ***intersection*** $\mathbf{x} \cap \mathbf{y}$ is a vector in $(F_2)^n$ defined as

$$\mathbf{x} \cap \mathbf{y} = (x_1y_1, x_2y_2, \ldots, x_ny_n)$$

The terms $x_i + y_i$ and x_iy_i are calculated modulo 2.
The ***weight*** of a vector $\mathbf{x}$ in $(F_2)^n$, denoted $w(\mathbf{x})$, is the number of 1s in $\mathbf{x}$.

The proofs of the following two lemmas are ommited as they are quite easy to determine once one gets the lemma.

Lemma 2.4. If $\mathbf{x}$, $\mathbf{y} \in (F_2)^n$, then $d(\mathbf{x}, \mathbf{y}) = w(\mathbf{x} + \mathbf{y})$.

Lemma 2.5. If $\mathbf{x}$, $\mathbf{y} \in (F_2)^n$, then

$$d(\mathbf{x}, \mathbf{y}) = w(\mathbf{x}) + w(\mathbf{y}) - 2w(\mathbf{x} \cap \mathbf{y}).$$

Theorem 2.6. Suppose d is odd. Then a binary (n, M, d)-code exists if and only if $(n+1, M, d+1)$-code exists.

Proof. $\Longrightarrow$: Suppose C is a binary (n, M, d)-code and $\mathbf{x} = x_1x_2\cdots x_n$ be one of it's codewords. Define $x_{n+1} = \sum_{i=1}^{n} x_i$. Let $\hat{C}$ be a $n+1$ length code, given by

$$\hat{C} = \{\hat{\mathbf{x}} \mid \hat{\mathbf{x}} = x_1x_2\cdots x_nx_{n+1} \forall\ \mathbf{x} \in C\}$$

Now $w(\hat{\mathbf{x}})$ is every codeword $\hat{\mathbf{x}}$ in $\hat{C}$, it follows from Lemma 2.5 that $d(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ will be even for all $\hat{\mathbf{x}}, \hat{\mathbf{y}}$ in $\hat{C}$. Hence, $d(\hat{C})$ is even. But $d \leq d(\hat{C}) \leq d+1$, so $d(\hat{C} = d+1$ as d is odd. Therefore, there exists a binary $(n+1, M, d+1)$-code. (This type of operation is called adding overall *parity-check* on a code C)
$\Longleftarrow$: Suppose C is a binary $(n+1, M, d+1)$-code. Let $\mathbf{x}, \mathbf{y} \in C$ such that $d(\mathbf{x}, \mathbf{y}) = d+1$. Now we remove one column where they differ from the whole code C. We are left now left with (n, M, d)-code. Therefore, binary (n, M, d)-code exists. □

Corollary 2.6.1. If d is odd, then $A_2(n+1, d+1) = A_2(n, d)$. Equivalently, if d is even, then $A_2(n, d) = A_2(n-1, d-1)$.

Proof. Follows directly from Theorem 2.6. □

Notion of a ***sphere*** in $(F_q)^n$, natural definition that follows from Hamming distance.

Definition 2.7. For any vector $\mathbf{u}$ in $(F_q)^n$, a ***sphere*** of radius r and centre $\mathbf{u}$, is given by

$$S(\mathbf{u}, r) = \{\mathbf{v} \in (F_q)^n \mid d(\mathbf{u}, \mathbf{v}) \leq r\}$$

Lemma 2.8. A sphere of radius r in $(F_Q)^n$ contains exactly

$$\binom{n}{0} + \binom{n}{1}(q-1) + \binom{n}{2}(q-1)^2 + \cdots + \binom{n}{r}(q-1)^r$$

vectors.

Proof. For finding vectors with a distance $d \in \{1, 2, \ldots, r\}$, we can choose d positions from the centre vector in $\binom{n}{d}$ ways, and we have $(q-1)$ choices for each of the positions. So, total vectors at a distance d from the the centre vector are $\binom{n}{d}(q-1)^d$. Then summing over all possible distances gives the result. □

Theorem 2.9 (**Hamming bound** or the **sphere-packing bound**). A q-ary (n, M, d)-code saisfies

$$M\left\{\binom{n}{0} + \binom{n}{1}(q-1) + \binom{n}{2}(q-1)^2 + \cdots + \binom{n}{t}(q-1)^t\right\} \leq q^n$$

where $t = \left\lfloor \dfrac{d-1}{2} \right\rfloor$.

Proof. By Corollary 1.8.1(ii), spheres of radius $\leq t$ and centered at the codewords won't overlap, therefore, the sum of elements of all these sphere will not exceed the *cardinality* of $(F_q)^n$. □

Definition 2.10. A code that achieves the *sphere-packing bound*, i.e. equality occurs in Theorem 2.9, is called a ***perfect code***.

Some *trivial* examples of *perfect codes* are binary-repition code, single word codes or the whole set $(F_q)^n$. But the *non-trivial* ones are the ones we are looking to find, as these codes will allow us to transmit/receive messages with very less probable grey region.

Remark. For binary codes, the *Hamming bound* turns out be close to the actual values of $A_2(n, d)$ when $n \geq 2d+1$. It is a weak bound for the case when $n \leq 2d$. For such cases, *Plotkin bound* is better and the following lemmas will lead to the bound.

Lemma 2.11. If there exists a binary (n, M, d)-code, then there exists a binary $(n-1, M', d)$-code such that $M' \geq \frac{M}{2}$. Further, $A_2(n-1, d) \geq \frac{A_2(n,d)}{2}$.

Proof. Let C be the existing (n, M, d)-code then if we partition the set C in two disjoint sets, those ending with 0 and 1, then atleast one of them will have *order* $\geq \frac{M}{2}$ (as the sum of orders is M). Then if we remove the last bit from this larger set then we have our binary $(n-1, M', d)$-code with $M' \geq \frac{M}{2}$. Also, therefore $A_2(n-1, d) \geq \frac{A_2(n,d)}{2}$. □

Lemma 2.12. If C is a binary (n, M, d)-code with $n < 2d$, then

$$M \leq \begin{cases} \dfrac{2d}{2d-n} & \text{if M is even} \\ \dfrac{2d}{2d-n} - 1 & \text{if M is odd} \end{cases}.$$

In paricular,

$$A_2(n, d) \leq 2 \left\lfloor \frac{d}{2d-n} \right\rfloor$$

Proof. Let $C = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ and T be the $\binom{M}{2} \times n$ matrix whose are vectors $\mathbf{x}_i + \mathbf{x}_j$ for $1 \leq i < j \leq M$. Now in T, number of non-zero entries per row is $d(\mathbf{x}_i, \mathbf{y}_j)$, i.e. greater equal to d, let the total non-zero entries of the matrix be $w(T)$. Thus,

$$\binom{M}{2} d \leq L \implies \frac{M(M-1)}{2} \cdot d \leq w(T)$$

If t_j codewords have 1 in j^{th} position then total number of 1s in j^{th} column of T are

$$t_j(M - t_j) \leq \begin{cases} \frac{M^2}{4} & \text{if M is even} \\ \frac{M^2-1}{4} & \text{if M is odd} \end{cases}$$

Thus,

$$w(T) \leq \begin{cases} \frac{M^2}{4} \cdot n & \text{if M is even} \\ \frac{M^2-1}{4} \cdot n & \text{if M is odd} \end{cases}$$

From above equations:

If M is even:

$$\begin{aligned} \frac{M(M-1)}{2} \cdot d &\leq \frac{M^2}{4} \cdot n \\ 2(M-1)d &\leq Mn \\ \frac{M}{2} &\leq \frac{d}{2d-n} \\ \frac{M}{2} &\leq \left\lfloor \frac{d}{2d-n} \right\rfloor \qquad \left(\text{as } \frac{M}{2} \in \mathbb{Z}\right) \\ M &\leq 2 \left\lfloor \frac{d}{2d-n} \right\rfloor \end{aligned}$$

If M is odd:

$$\begin{aligned} \frac{M(M-1)}{2} \cdot d &\leq \frac{M^2-1}{4} \cdot n \\ 2Md &\leq (M+1)n \\ M &\leq \frac{n}{2d-n} \\ \frac{M+1}{2} &\leq \frac{d}{2d-n} \\ \frac{M+1}{2} &\leq \left\lfloor \frac{d}{2d-n} \right\rfloor \qquad \left(\text{as } \frac{M+1}{2} \in \mathbb{Z}\right) \\ M &\leq 2 \left\lfloor \frac{d}{2d-n} \right\rfloor \end{aligned}$$

□

Theorem 2.13 (Plotkin Bound). For a binary (n, M, d)-code:
(i) if d is even and $n < 2d$, then $A_2(n,d) \leq 2\left\lfloor \frac{d}{2d-n} \right\rfloor$.
(ii) if d is odd and $n < 2d+1$, then $A_2(n,d) \leq 2\left\lfloor \frac{d+1}{2d+1-n} \right\rfloor$.
(iii) if d is even, then $A_2(2d,d) \leq 4d$.
(iv) if d is odd, then $A_2(2d+1,d) \leq 4d+4$.

Proof. (i) Directly follows from Lemma 2.12.

(ii) From Corollory 2.6.1, we have $A_2(n,d) = A_2(n+1,d+1)$, therefore, $A_2(n,d) = A_2(n+1,d+1) = 2\left\lfloor \frac{d+1}{2d+1-n} \right\rfloor$(follows from part(i) as $d+1$ is even)

(iii) From Lemma 2.11, $A_2(2d,d) \leq 2A_2(2d-1,d)$. Now, from part(i), we have $A_2(2d-1,d) \leq 2\left\lfloor \frac{d}{1} \right\rfloor \implies A_2(2d,d) \leq 4d$

(iv) Follows from Corollory 2.6.1 and part(iii).

□

Definition 2.14. A ***balanced-block design*** consists of a set S of v elements, called *points*, and a collection of b subsets of S, called *blocks*, such that for some fixed k, r and λ

1. each blocks contain exactly k points.
2. each point occurs in exactly r blocks.
3. each pair of points occurs together in exactly λ blocks.

We denote this design as a (b, v, k, r, λ)-*design.*

A *balanced block design* satisfy these two basic conditions, namely, (i) $bk = vr$. (ii) $r(k-1) = \lambda(v-1)$. First one results from counting all the points in two ways, points in each block and number of blocks each point is in. Second results from equating the pairs of a particular point in two ways.

Definition 2.15. The ***incidence-matrix*** $A = [a_{ij}]$ of a block design is a $v \times b$ matrix, where rows corresponds to points $x_1, x_2, \ldots, x_v$ of the design while the columns represent the blocks $B_1, B_2, \ldots, B_b$. The matrix is given by

$$a_{ij} = \begin{cases} 1 & \text{if } x_i \in B_j \\ 0 & \text{if } x_i \notin B_j \end{cases}$$

Incidence matrices of block codes are used to generate *perfect codes.*

§3 Finite fields and Vector spaces over finite fields

Definition 3.1. A field F is a set of with two operations $+$(addition) and $\cdot$ (multiplication) satisfiying the following conditions.

(i) $a+b,\ a\cdot b \in F\ \forall\ a,\ b\in F.$
(ii) $a+b=b+a, a\cdot b=b\cdot a\ \forall\ a,\ b\in F.$ (commutative)
(iii) $(a+b)+c=a+(b+c), a\cdot(b\cdot c)=(a\cdot b)\cdot c.$ (associative)
(iv) $a\cdot(b+c)=a\cdot b+a\cdot c\ \forall\ a,\ b,\ c\in F.$ (distributive)
(v) $\exists 0,1\in F$ such that $a+0=a, a\cdot 1=a\ \forall\ a\in F.$ (identity elements)
(vi) $\exists c\in F$ such that $a+c=0\ \forall\ a\in F.$ (additive inverse of a)
(vii) $\exists c\in F$ such that $a\cdot c=1\ \forall\ a\in F,\ a\neq 0.$ (multiplicative inverse of a)

We will denote $a\cdot b$ simply by ab, additive inverse of a by $-a$ and multiplicative inverse of a by a^{-1}. For any field F, we can deduce the following from the axioms of definition:

1. The identity elements are unique.
2. $a0=0$.
3. $ab=0 \implies a=0$ or $b=0$.
4. $-(-a)=a,\ (a^{-1})^{-1}=a.$
5. $(-1)a=-a$, also $(-a)(-a)=aa$ and we can continue.

3.1 Finite fields

Definition 3.2. A ***finite field*** is a field having a finite number of elements. The number of elements is called the ***order*** of the *field.*

Theorem 3.3. There exists a field of order q iff q is a *prime-power.* Also, if q is a prime, there is only one field, upto relabelling.

We will not go into proof as it requires some concepts of abstract algebra, which will be beyond the scope of this report. A field of order q is often called *Galois Field* of order q and is denoted by $GF(q)$.
Note: From now on in this report, mentioning $GF(q)$ will imply that q is a prime power.

Theorem 3.4. $\mathbb{Z}_m$ is a field (addition and multiplication defined as *modulo* m) iff m is a *prime.*

Proof. The first six properties can be easily verified even if m is not a prime, as the addition and multiplication are *modular.*
Now for the multiplicative inverse property,
$\implies$: Suppose m is not prime, then $m=ab$ for some non-zero $a,b<m$, but then

$$ab\equiv 0 \pmod{m} \implies a=0 \text{ or } b=0$$

which is contradiction. Hence, m is prime.
$\impliedby$: We have to prove that for all a in $\mathbb{Z}_m$, there exists a multiplicative inverse, a^{-1}. Consider the elements $a, 2a, 3a, \dots, (m-1)a$, each of these elements will have non-zero remainder with m. Further, these remainders will be distinct, for otherwise $(i-j)a\equiv 0 \pmod{m}$ for some $i,j\in\{1,2,\dots,m-1\}, i\neq j$, therefore $(i-j)a\equiv 0 \pmod{m}$, which is not possible as i,j are distinct and $|i-j|, a<m$ which is a prime. Therefore, there must exist an element with remainder 1 in the initial set . Hence, the multiplicative inverse exists. □

Theorem 3.5. Suppose F is a finite field, with $\alpha\in F$, then there exists a prime number p such that $p\alpha=\alpha+\alpha+\cdots+\alpha(\text{p terms})=0$. The prime number p is called ***characterstic*** of field F.

Proof. The term $n\alpha$ must have a same value for two different values of n as we iterate over n because F is a finite field. Let those n be $a,\ b$ such that $0<a<b$, then $(b-a)\alpha=0$. Let the minimum value of $b-a$ be p. So, $p\alpha=0$. If p was co-prime, then $p=lm$, with $0<l,m<p \implies (lm)\alpha=(l\alpha)(m\alpha)=0 \implies l\alpha=0$ or $m\alpha=0$, which is contradiction. Hence, p is a prime. □

3.2 Vector spaces over finite fields

Definition 3.6. A set is V is called a ***vector-space*** over a field F, if $+$ and $\cdot$ are defined as $+ : V \times V \to V$ binary-operation on V, and $\cdot : F \times V \to V$ a function, and the following axioms are satisfied.

(i) $u + v = v + u \;\forall\; u, v \in V$.
(ii) $u + (v + w) = (u + v) + w \;\forall\; u, v, w \in V$.
(iii) There exists $0 \in V$ such that $\forall\; u \in V : v + 0 = v$.
(iv) For every $u \in V$, $\exists w \in V$ such that $v + w = 0$.
(v) $a \cdot (v + w) = a \cdot u + a \cdot v \;\forall\; u, v \in V,\; a \in F$.
(vi) $(a + b) \cdot (u) = a \cdot u + b \cdot u \;\forall\; u \in V,\; a, b \in F$.
(vii) $(ab) \cdot (u) = a \cdot (b \cdot u) \;\forall\; u \in V,\; a, b \in F$.
(viii) $1 \cdot u = u \;\forall\; u \in V$ (1 is multiplicative identitiy of F).

Elements of V are called *vectors* and of F are called *scalars*.

The set $GF(q)^n$ of all the n-tuples over $GF(q)$ will be denoted as $V(n, q)$.
It can be seen that $V(n, q)$ is a vector-space over $GF(q)$ if we define addition and scalar multiplication as follows. For $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}, \mathbf{y} = \{y_1, y_2, \ldots, y_n\} \in V(n, q)$ and $a \in F$.

- $\mathbf{x} + \mathbf{y} = (x_1 + y_1, x_2 + y_2, \ldots, x_n + y_n)$
- $a\mathbf{x} = (ax_1, ax_2, \ldots, ax_n)$

Definition 3.7. A subset of $V(n, q)$ is called a ***subspace*** of $V(n, q)$ if itself is vector space under same addition and scalar multiplication.

Theorem 3.8. A subset C of $V(n, q)$ is a subspace if and only if
(i) If $\mathbf{x}, \mathbf{y} \in C$, then $\mathbf{x} + \mathbf{y} \in C$.
(ii) If $a \in GF(q)$ and $\mathbf{x} \in C$, then $a\mathbf{x} \in C$.

Proof. One can easily see that if these conditions are true, then all the axioms of vector space are satisfied. Therefore, C is a subspace. □

A ***linear combination*** of r vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r$ is a vector of the form $a_1\mathbf{v}_1 + a_2\mathbf{v}_2 + \cdots + a_r\mathbf{v}_r$, where a_i are scalars. **Note:** Set of all linear combinations of a set of given vectors is a subspace of$V(n, q)$.
A set of vectors $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r\}$ is called ***linearly independent*** if

$$a_1\mathbf{v}_1 + a_2\mathbf{v}_2 + \cdots + a_r\mathbf{v}_r = 0 \implies a_1 = a_2 = \cdots = a_r = 0.$$

If C is a subspace of $V(n, q)$. Then a subset $\{textbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r\}$ of C is called ***generating set*** if every vector of C can be expressed as the linear combination of these vectors.
A *generating set* of C which is also linearly independent is called ***basis*** of C.

Theorem 3.9. If C is a non-trivial subspace of $V(n, q)$. Then any generating set of C contians a basis of C.

Proof. We equate linear combination of generating matrix elements with $\mathbf{0}$, then the vectors with non-zero coefficients are removed from the generating matrix and we get a basis of C. □

Theorem 3.10. Suppose $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k\}$ be the basis of a subspace C of $V(n, q)$. Then
(i) every vector of C can be expressed *uniquely* as a linear combination of the basis vectors.
(ii) C contains exactly q^k vectors.

The order of basis of C is called the ***dimension*** of the subspace C, denoted by $\dim C$.

Proof. Let the basis of C be the set $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k\}$.

(i) If $\mathbf{x} = a_1\mathbf{v}_1 + a_2\mathbf{v}_2 + \cdots + a_k\mathbf{v}_k$, and $\mathbf{x} = b_1\mathbf{v}_1 + b_2\mathbf{v}_2 + \cdots + b_k\mathbf{v}_k$, then $(a_1 - b_1)\mathbf{v}_1 + (a_2 - b_2)\mathbf{v}_2 + \cdots + (a_k - b_k)\mathbf{v}_k = 0$, but as basis is linearly independent, $a_i - b_i = 0$ for all $0 < i \leq k$.

(ii) q choices for coefficient of each the basis element, therefore q^k elements in the subspace.

□

§4 Linear Codes

4.1 Introduction

Definition 4.1. A ***linear code*** C over $GF(q)$ is a subspace of $V(n, q)$.

Notation: If linear code C is a k-dimensional subspace of $V(n, q)$ with minimum distance d, C is denoted by $[n, k]$-code or $[n, k, d]$-code depending on the context.

The ***weight*** $\mathrm{w}(\mathbf{x})$ of a vector $\mathbf{x}$ in $V(n, q)$ is the number of non-zero entries of $\mathbf{x}$.

Lemma 4.2. If $\mathbf{x}$ and $\mathbf{y} \in V(n, q)$ then

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

Proof. $\mathbf{x} - \mathbf{y}$ will be non-zero exactly at those places where $\mathbf{x}$ and $\mathbf{y}$ differ. □

Theorem 4.3. Let C be a linear code and let $w(C)$ be the smallest of the non-zero weights. Then $d(C) = w(C)$.

Proof. Let $\mathbf{x}$ and $\mathbf{y}$ be codewords with distance $d(C)$. Then by Lemma 4.2,

$$d(C) = w(\mathbf{x} - \mathbf{y}) \geq w(C)$$

as $\mathbf{x} - \mathbf{y} \in C$. Let $\mathbf{z}$ be a codeword such that $w(\mathbf{z}) = w(C)$. Since $\mathbf{0} \in C$, by Lemma 4.2 again,

$$w(C) = w(\mathbf{z} - \mathbf{0}) = d(\mathbf{z}, \mathbf{0}) \geq d(C).$$

Therefore, $d(C) = w(C)$. □

This theorem allows us to calculate $d(C)$ in M iterations, which otherwise would have been $\binom{M}{2} = \frac{M(M-1)}{2}$, one of the reasons we focus on linear codes. One more reason being that every codeword can be expressed in terms of k codewords, in the case of $[n, k]$-code, we don't have to list all the codewords.

Definition 4.4. A $k \times n$ matrix whose rows form a basis of a linear $[n, k]$-code is called a ***generator matrix*** of the code.

Definition 4.5. Two linear codes are said to be ***equivalent*** if one can be obtained from another by combinations of operations of the types,

(i) permuations of the positions of the code.
(ii) multiplication of the symbols of fixed position by a non-zero scalar.

Theorem 4.6. Two $k \times n$ matrices generate equivalent linear $[n, k]$-codes over $GF(q)$ if one can be obtained from other by a sequence of these operations.

(i) Permutations of the rows.
(ii) Multiplication of a row by a non-zero scalar.
(iii) Addition of the scalar multiple of one row to another.
(iv) Permutations of the columns.
(v) Multiplication of any column by a non-zero scalar.

Proof. The first three operations are called ***row operations***, and they will preserve the linear independence of the row vectors and will replace one basis by another of the same code. Last two are called ***column operations***, they convert the basis into basis of a equivalent code. □

Theorem 4.7. Let G be a generator matrix of an $[n, k]$-code. Then by operations described in Theorem 4.6, G can be converted into ***standard form***

$$[I_k \mid A],$$

where I_k is the $k \times k$ identitiy matrix, and A is a $k \times (n - k)$ matrix.

Proof. Let $G = [g_{ij}]$ and $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_k$ and $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_n$ be its rows and columns respectively, then the following scheme will convert it to standard form.
Suppose G has already been transformed to

$$\begin{bmatrix} 1 & \cdots & 0 & g_{1j} & \cdots & g_{1n} \\ \vdots & \ddots & \vdots & \vdots & & \vdots \\ 0 & \cdots & 1 & g_{j-1,j} & \cdots & g_{j-1,n} \\ 0 & \cdots & 0 & g_{jj} & \cdots & g_{jn} \\ \vdots & & \vdots & \vdots & & \vdots \\ 0 & \cdots & 0 & g_{kj} & \cdots & g_{kn} \end{bmatrix}$$

where $j \in \{1, 2, \ldots, k\}$.

1. If $g_{jj} \neq 0$, proceed to next step. If $g_{ij} = 0$, if for some $i > j$, $g_{ij} \neq 0$, replace $\mathbf{r}_j$ with $\mathbf{r}_i$. If not, replace $\mathbf{c}_j$ with $\mathbf{c}_p$, where $g_{jp} \neq 0$. (We could have sone the column step directly without looking for the row first, but we are trying that the matrix remains basis of the original code C, if possible.)
2. Multiply $\mathbf{r}_j$ with g_{jj}^{-1}, so that g_{jj} becomes 1.
3. For each $i \in \{1, 2, \ldots, k\}, i \neq j$, convert $\mathbf{r}_i \to \mathbf{r}_i - g_{ij}\mathbf{r}_j$.

Now after these steps, we have converted $\mathbf{c}_j$ to the the required form. So repeating for all values of j will give the standard form. □

Note: *Standard form* of a generator matrix is not unique, for instance we can interchange the columns of A and still satisfy all the conditions. Also, the standard form will remain generator matrix of the original code C iff we don't use the column operations. It is possible if and only if the first k columns of generator matrix are *linearly independent.*

4.2 Encoding with a linear code

Let C be an $[n, k]$-code over $GF(q)$ with generator matrix G. It can be used to communicate q^k distinct messages. If the rows of G are $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_k$, then we *encode* a message vector $\mathbf{u} = u1u2 \cdots u_k \in V(k, q)$ by a function that maps $V(k, q) \to k$-dimensional subspace of $V(n, q)$ (the code C).

$$\mathbf{u}G = \sum_{i=1}^{k} u_i \mathbf{r}_i$$

In particular, if $G = [I_k \mid A]$ (standard form, $A = [a_{ij}]$), then encoding is

$$\mathbf{x} = \mathbf{u}G = x_1 x_2 \cdots x_k \cdots x_n,$$

where $x_i = u_i$, $1 \leq i \leq k$ are ***message digits***, and

$$x_{k+i} = \sum_{j=1}^{k} a_{ij} u_j \quad 1 \leq i \leq n - k,$$

are the ***check digits***. The check digits represent *redundancy* added to protect against noise.

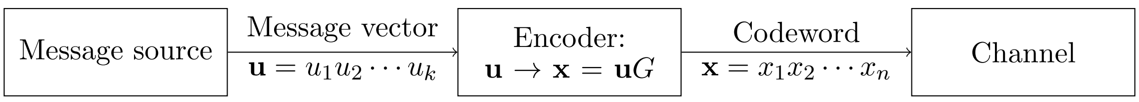

Message source
Message vector
$\mathbf{u} = u_1 u_2 \cdots u_k$
Encoder:
$\mathbf{u} \to \mathbf{x} = \mathbf{u}G$
Codeword
$\mathbf{x} = x_1 x_2 \cdots x_n$
Channel

Figure 1: Encoding a message vector.

4.3 Decoding with a linear code

Suppose the codeword sent is $\mathbf{x} = x_1x_2\cdots x_n$ and the recieved vector is $\mathbf{y} = y_1y_2\cdots y_n$. We define ***error vector*** $\mathbf{e}$ to be

$$\mathbf{e} = \mathbf{y} - \mathbf{x} = e_1e_2\cdots e_n.$$

Definition 4.8. Suppose that C is an $[n,k]$-code over and $\mathbf{a}$ is any vector in $V(n,q)$. Then set

$$\mathbf{a} + C = \{\mathbf{a} + \mathbf{x} \mid \mathbf{x} \in C\}$$

is called a ***coset*** of C.

Lemma 4.9. Suppose that $\mathbf{x} + C$ is a coset of C and $\mathbf{b} \in \mathbf{a} + C$. Then $\mathbf{b} + C = \mathbf{a} + C$.

Proof. Since $\mathbf{b} \in \mathbf{a} + C$, we have $\mathbf{b} = \mathbf{a} + \mathbf{x}$ for some $\mathbf{x} \in C$. Now if $\mathbf{b} + \mathbf{y} \in \mathbf{b} + C$, then

$$\mathbf{b} + \mathbf{y} = (\mathbf{a} + \mathbf{x}) + \mathbf{y} = \mathbf{a} + (\mathbf{x} + \mathbf{y}) \in \mathbf{a} + C.$$

Hence, $\mathbf{b} + C \subseteq \mathbf{a} + C$. Similarly, if $\mathbf{a} + \mathbf{z} \in \mathbf{a} + C$, then

$$\mathbf{a} + \mathbf{z} = (\mathbf{b} - \mathbf{x}) + \mathbf{z} = \mathbf{b} + (\mathbf{z} - \mathbf{x}) \in \mathbf{b} + C.$$

Hence, $\mathbf{a} + C \subseteq \mathbf{b} + C$. Therefore, $\mathbf{b} + C = \mathbf{a} + C$. □

Theorem 4.10. Suppose C is an $[n,k]$-code over $GF(q)$. Then

(i) every vector of $V(n,q)$ is in some coset of C,
(ii) every coset contains exactly q^k vectors,
(iii) two cosets are either disjoint or coincide.

Proof.

(i) If $\mathbf{a} \in V(n,q)$, then $\mathbf{a} = \mathbf{a} + 0 \in \mathbf{a} + C$.
(ii) It can be easily seen that the mapping $C \to \mathbf{a} + C$ is one-one. Therefore, $|C| = |\mathbf{a} + C|$.
(iii) Suppose cosets $\mathbf{a} + C$ and $\mathbf{b} + C$ overlap and $\mathbf{v} \in (\mathbf{a} + C) \cap (\mathbf{b} + C)$. Thus, for some $\mathbf{x}, \mathbf{y} \in C$,

$$\begin{aligned} &\mathbf{v} = \mathbf{a} + \mathbf{x} = \mathbf{b} + \mathbf{y} \\ \Longrightarrow\ &\mathbf{a} = \mathbf{b} + \mathbf{y} - \mathbf{x} \\ \Longrightarrow\ &\mathbf{a} \in \mathbf{b} + C \end{aligned}$$

Therefore, by Lemma 4.9, $\mathbf{a} + C = \mathbf{b} + C$.

□

Definition 4.11. The vector having minimum weight in a coset is called the ***coset leader***. (If there are more than one vectors with minimum weight then we choose anyone.)

Theorem 4.10 implies:

$$V(n,q) = (\mathbf{0} + C) \cup (\mathbf{a}_1 + C) \cup \cdots \cup (\mathbf{a}_s + C)$$

where $s = q^{n-k} - 1$, and by Lemma 4.9, we can take $\mathbf{a}_i$ to be the coset leaders.

Definition 4.12. A ***standard array*** for an $[n,k]$-code C is a $q^{n-k} \times q^k$ array of all the vectors of $V(n,q)$. First row will consist of code C in any order except $\mathbf{0}$ which will be in in the first column. Following rows will be cosets with coset leaders being the first element of each row, remaining elements in each row will be sum of its coset leader with corresponding codeword in the same column.

For example, let C be a binary $[4,2]$-code with generator matrix

$$G = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

then $C = \{0000, 1011, 0101, 1110\}$.
The standard array of C will be

$$\begin{array}{rcccc}
\text{codewords} \rightarrow & 0000 & 1011 & 0101 & 1110 \\
 & 1000 & 0011 & 1101 & 0110 \\
 & 0100 & 1111 & 0001 & 1010 \\
 & 0010 & 1001 & 0111 & 1100 \\
 & \uparrow & & & \\
 & \text{coset} & & & \\
 & \text{leaders} & & &
\end{array}$$

Decoding scheme using standard array: When $\mathbf{y}$ is recieved its position is found in array. Then the *decoder* assumes that the error vector $\mathbf{e}$ is the coset leader, and $\mathbf{y}$ is decoded as $\mathbf{x} = \mathbf{y} - \mathbf{e}$ at the top of the column containing $\mathbf{y}$. By choosing the minimum weight vector as the coset leader, we ensure that standard array decoding scheme is a nearest neighbour decoding scheme.

Theorem 4.13. Let C be a binary $[n, k]$-code, and for $i \in \{1, 2, \dots, n\}$, let α_i denote the number of cosets leaders of weight i. Then *probability* $P_{\text{corr}}(C)$ of correctly recognizing the codeword is given by

$$P_{\text{corr}}(C) = \sum_{i=0}^{n} \alpha_i p^i (1-p)^{n-i}.$$

Proof. The probability of error code to be vector of weight i is $p^i(1-p)^{n-i}$. Therefore, the result follows directly from the definition of decoding scheme. $\square$

Note: If $d(C) = 2t + 1$ or $2t + 2$, then $\alpha_i = \binom{n}{i}$ for $0 \geq i \geq t$. In particular, if C is a perfect code, in addition to the general result, $\alpha_i = 0$ for $i > t$ also holds.

Theorem 4.14. Let C be a binary $[n, k]$-code, and let A_i denote the number of codewords of weight i. Then *probability* $P_{\text{undetec}}(C)$ of an error going undetected is given by

$$P_{\text{undetec}}(C) = \sum_{i=1}^{n} A_i p^i (1-p)^{n-i}.$$

Proof. The error will only go undetected if and only if $\mathbf{y} - \mathbf{x}$ is a non-zero codeword, where $\mathbf{y}$ and $\mathbf{x}$ are received and sent vectors respectively. The probability of $\mathbf{y} - \mathbf{x} = \mathbf{z} \in C$ is $p^{w(\mathbf{z})}(1-p)^{n-w(\mathbf{z})}$, thus summing over non-zero $\mathbf{z}$ gives the result. $\square$

Definition 4.15. For a linear $[n, k]$-code C, ***rate*** is defined as the ratio of number of message symbols to the total number of symbols sent, i.e. $R(C) = \dfrac{k}{n}$.

Good code generally will have high rate.

Definition 4.16. The ***capacity*** $\mathscr{C}(p)$ of a binary symmetric channel with symbol error probability p is

$$\mathscr{C}(p) = 1 + p \log_2 p + (1-p)\log_2(1-p).$$

Theorem 4.17. Shannon's Theorem Suppose a channel is binary symmetric with symbol error probability p. Let $R \in \mathbb{R}$ satisfying $R < \mathscr{C}(p)$. Then for all $\epsilon > 0$, there exists, for some large n, an $[n, k]$-code C of rate $\dfrac{k}{n} \geq R$ such that $P_{\text{err}}(C) < \epsilon$. (where $P_{\text{err}}(C) = 1 - P_{\text{corr}}(C)$)

§5 Dual Code, parity-check matrix and syndrome decoding

Definition 5.1. The ***inner product*** $\mathbf{u}\cdot\mathbf{v}$ of vectors $\mathbf{u} = u_1u_2\cdots u_n$ and $\mathbf{v} = v_1v_2\cdots v_n$ in $V(n,q)$ is scalar defined by

$$\mathbf{u}\cdot\mathbf{v} = u_1v_1 + u_2v_2 + \cdots u_nv_n.$$

If $\mathbf{u}\cdot\mathbf{v} = 0$, then $\mathbf{u}$ and $\mathbf{v}$ are called ***orthogonal***.

Lemma 5.2. For any $\mathbf{u}$, $\mathbf{v}$ and $\mathbf{w}$ in $V(n,q)$ and $\lambda,\ \mu \in GF(q)$,

1. $\mathbf{u}\cdot\mathbf{v} = \mathbf{v}\cdot\mathbf{u}$
2. $(\lambda\mathbf{u} + \mu\mathbf{v})\cdot\mathbf{w} = \lambda(\mathbf{u}\cdot\mathbf{w}) + \mu(\mathbf{v}\cdot\mathbf{w})$

Definition 5.3. Given a linear $[n,k]$-code C, the ***dual code*** of C, denoted by $C^\perp$ is defined as

$$C^\perp = \{\mathbf{v} \in V(n,q) \mid \mathbf{v}\cdot\mathbf{u} = 0\ \forall\ \mathbf{u} \in C\}.$$

Lemma 5.4. Suppose C is an $[n,k]$-code having a generator matrix G, then $\mathbf{v} \in C^\perp \iff \mathbf{v}G^T = 0$, where G^T is transpose of G.

Proof. $\implies$: This part is obvious as rows of G are codewords.
$\impliedby$: Suppose the rows of G are $\mathbf{r}_1, \mathbf{r}_2, \dots \mathbf{r}_k$, and thus $\mathbf{v}\cdot\mathbf{r}_i = 0$ for all i. If $u \in C$, then $u = \sum_{i=1}^k \lambda_i\mathbf{r}_i$ for some scalars λ_i, and

$$\begin{aligned}\mathbf{v}\cdot\mathbf{u} &= \sum_{i=1}^k \lambda_i(\mathbf{v}\cdot\mathbf{r}_i) \qquad \text{(by Lemma 5.2)}\\ &= \sum_{i=1}^k \lambda_i 0 = 0.\end{aligned}$$

Hence, $\mathbf{v}$ is orthogonal to all codewords in C. □

Theorem 5.5. Suppose C is an $[n,k]$-code over $GF(q)$. Then the dual code $C^\perp$ is a linear $[n,n-k]$-code.

Proof. Suppose $\mathbf{v}_1, \mathbf{v}_2 \in C^\perp$ and $a \in GF(q)$. Then, for all $\mathbf{u} \in C$,

$$\begin{aligned}(\mathbf{v}_1 + \mathbf{v}_2)\cdot\mathbf{u} &= \mathbf{v}_1\cdot\mathbf{u} + \mathbf{v}_2\cdot\mathbf{u}\\ &= 0\\ (a\mathbf{v}_1)\cdot(u) &= a(\mathbf{v}_1\cdot\mathbf{u})\\ &= 0\end{aligned}$$

Hence, $C^\perp$ is a linear code.
For the dimension part, notice that if two codes C_1 and C_2 are equivalent, then so are $C_1^\perp$ and $C_2^\perp$. Hence it will be enough to show $\dim C^\perp = n-k$ in the case when C has a standard form of generator matrix

$$G = \begin{bmatrix} 1 & \cdots & 0 & a_{11} & \cdots & a_{1,n-k}\\ \vdots & \ddots & \vdots & \vdots & & \vdots\\ 0 & \cdots & 1 & a_{k1} & \cdots & a_{k,n-k}\end{bmatrix}$$

Then

$$C^\perp = \left\{(v_1, v_2, \dots, v_n) \in V(n,q) \mid v_i + \sum_{j=1}^{n-k} v_{j+k}a_{ij} = 0,\ \forall\ i \in \{1, 2, \dots, n-k\}\right\}$$

We have q^{n-k} choices for $(v_{k+1}, \dots, v_n)$, and for each combination we have a unique vector $v_1v_2\cdots v_n$ in $C^\perp$. Hence, $|C^\perp| = q^{n-k}$, and $\dim C^\perp = n-k$. □

Theorem 5.6. For any $[n,k]$-code C, $(C^\perp)^\perp = C$.

Proof. $C \subseteq (C^\perp)^\perp$ since every vector in C is orthogonal to every vector in $C^\perp$. But $\dim (C^\perp)^\perp = n-(n-k) = k = \dim C$. Therefore, $C = (C^\perp)^\perp$. □

Definition 5.7. A ***parity-check matrix*** H for an $[n,k]$-code C is a generator matrix of $C^\perp$.

Thus, H is $(n-k)\times n$ matrix satisfying $GH^T = \mathbf{0}$, where $\mathbf{0}$ is the all-zero matrix. It follows from Lemma 5.4 and Theorem 5.6, that C can be written as

$$C = \{\mathbf{x} \in V(n,q) \mid \mathbf{x}H^T = \mathbf{0}\}.$$

Theorem 5.8. If $G = [I_k \mid A]$ is the standard form of generator matrix of an $[n,k]$-code C, then a parity-check matrix for C is $H = [-A^T \mid I_{n-k}]$.

Proof. Suppose

$$G = \left[\begin{array}{ccc|ccc} 1 & \cdots & 0 & a_{11} & \cdots & a_{1,n-k} \\ \vdots & \ddots & \vdots & \vdots & & \vdots \\ 0 & \cdots & 1 & a_{k1} & \cdots & a_{k,n-k} \end{array}\right]$$

Let

$$H = \left[\begin{array}{ccc|ccc} -a_{11} & \cdots & -a_{k1} & 1 & \cdots & 0 \\ \vdots & & \vdots & \vdots & \ddots & \vdots \\ -a_{1,n-k} & \cdots & -a_{k,n-k} & 0 & \cdots & 1 \end{array}\right]$$

Then H has the required size of a parity-matrix and its rows are linearly independent (as coefficients of 1 in the identitiy part will remain). Also, $\mathbf{g}_i \cdot \mathbf{h}_j$ (where $\mathbf{g}_i$ and $\mathbf{h}_j$ are some rows of G and H respectively) is

$$0 + \cdots + 0 + (-a_{ij}) + 0 + \cdots + 0 + a_{ij} + 0 + \cdots + 0 = 0$$

□

Remark. Minus signs are unnecessary in the binary case.

Definition 5.9. A parity-check matrix is called to be in ***standard form*** if $H = [B \mid I_{n-k}]$.

So from Theorem 5.8, we can get standard form of generator matrix from standard form of parity-check matrix and vice-versa.

Now we will see some more efficient decoding schemes using *parity-check matrices*, but before that some definitions and lemmas.

Definition 5.10. Suppose H is a parity-check matrix of an $[n,k]$-code C. Then for any vector $y \in V(n,q)$, the row vector

$$S(\mathbf{y}) = \mathbf{y}H^T$$

is called the ***syndrome*** of $\mathbf{y}$.

It is clear that $S(\mathbf{y}) = 0$ if and only if $\mathbf{y} \in C$.

Lemma 5.11. Two vectors $\mathbf{u}$ and $\mathbf{v}$ are in the same coset of C if and only if they have the same syndrome.

Proof. $\mathbf{u}$ and $\mathbf{v}$ are in the same coset

$$\begin{aligned} &\iff \mathbf{u} + C = \mathbf{v} + C \\ &\iff \mathbf{u} - \mathbf{v} \in C \\ &\iff (\mathbf{u} - \mathbf{v})H^T = \mathbf{0} \\ &\iff \mathbf{u}H^T = \mathbf{v}H^T \\ &\iff S(\mathbf{u}) = S(\mathbf{v}) \end{aligned}$$

□

Corollary 5.11.1. There is a one-to-one correspondance between cosets and syndromes.

In standard array decoding, if n is large, finding codewords in the array becomes increasingly inefficient. The following *syndrome decoding scheme* is much more efficient.

Syndrome Decoding Scheme: Instead of storing all the cosets in the standard array, if we store the coset leaders along with their corresponding coset syndromes, will be sufficient for achieving the same probability of error detection. When a vector $\mathbf{y}$ is received, we calculate $S(\mathbf{y}) = \mathbf{y}H^T$ and locate $\mathbf{z} = S(\mathbf{y})$ in the syndromes column, find the corresponding coset leader $f(\mathbf{z})$, and decode $\mathbf{y}$ as $\mathbf{x} = \mathbf{y} - f(\mathbf{z})$. This works because of Corollary 5.11.1.

For example, let C be a binary $[4, 2]$-code with generator matrix

$$G = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

then $C = \{0000, 1011, 0101, 1110\}$ and,

$$H = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix}$$

The syndrome look-up table of C will be

syndrome $\mathbf{z}$	coset leaders f($\mathbf{z}$)
00	0000
11	1000
01	0100
10	0010

Incomplete Decoding Scheme: In this we mix error correction as well detection. If $d(C) = 2t + 1$ or $2t + 1$, we can precisely correct $\leq t$ errors, using syndrome lookup table as all the vectors with weight $\leq t$ will be coset leaders. Otherwise, we will simply seek *re-transmission.* So now, we now need to store even lesser data in the lookup table.

§6 The Hamming Codes

The Hamming codes, defined in this section, are an importtant family of single-error-correcting codes, which are easier to to encode and decode than other similar codes.
These codes are best defined using their parity-check matrix:

Definition 6.1. Let r be a positive integer and let H be an $r \times (2^r - 1)$ matrix whose columns are the distinct non-zero vectors of $V(r, 2)$. Then the code having H as its parity-check matrix is called a ***binary Hamming code*** and is denoted by $\mathrm{Ham}(r, 2)$.

Remark. (i) $\mathrm{Ham}(r, 2)$ has length $n = 2^r - 1$ and dimension $k = n - r$. r is also called the ***redundancy*** of the code.
(ii) $\mathrm{Ham}(r, 2)$ is the code generated by any of the matrix H, since its colums can be taken in any order.

Theorem 6.2. The binary Hamming code $\mathrm{Ham}(r, 2)$, for $r \geq 2$,

(i) is a $[2^r - 1, 2^r - 1 - r]$-code;
(ii) has a minimum distance 3 (therefore, single-error correcting);
(iii) is a perfect code.

Proof.

(i) By definition, $\mathrm{Ham}(r, 2)^{\perp}$ is a $[2^r - 1, r]$-code ans so $\mathrm{Ham}(r, 2)$ is a $[2^r - 1, 2^r - 1 - r]$-code.
(ii) Since, $\mathrm{Ham}(r, 2)$ is a linear code, it is enough, by Theoem 4.3, to show that every non-zero codeword has weight≥ 3.
Firstly, suppose that the code has codeword $\mathbf{x}$ of weight 1, with non-zero element at ith position, but as it is orthogonal to every row in H, this implies that the ith column of H is the all-zero vector, contradicting the definition of H.
Similarly, if codeword $\mathbf{x} \in \mathrm{Ham}(r, 2)$, where
$$\mathbf{x} = 0 \cdots 010 \cdots 010 \cdots 0$$
with 1s in the ith and jth places. Let $H = \{h_{pq}\}$ then since $\mathbf{x}$ is orthogonal to each row of H, this implies
$$h_{si} = h_{sj} \pmod{2} \quad \forall\, s \in \{1, 2, \dots, r\}$$
Hence the ith and jth columns of H are identical, again contradicting the definition. Thus $d(\mathrm{Ham}(r, 2)) \geq 3$, and it is easy to see that the equality exists, for example vector $11100 \cdots 0$ can exist in $\mathrm{Ham}(r, 2)$, by a suitable rearrangement of columns in H.
(iii) The left-hand side of shere packing bound is
$$2^{n-r}\left(1 + \binom{n}{1}\right) = 2^{n-r}(1 + n) = 2^n$$
Therefore, the bound is acheived and the code is a perfect code.

□

As a binary Hamming code is a perfect code, the coset leaders are precisely the 2^r vectors of $V(n, 2)$ of weight ≤ 1. Now if we arrange the columns of H in order of increasing binary numbers, we can have the following nice **decoding algorithm**. We calculate the syndrome $S(\mathbf{y})$ of the received vector $\mathbf{y}$ as usual, if it is $\mathbf{0}$, then assume it was the codeword sent, otherwise assume single error and the value of $S(\mathbf{y})$ gives the position of the error, as the syndromes are the columns of H itself.

Definition 6.3. The ***extended binary Hamming code*** $\mathrm{H\hat{a}m}(r, 2)$ is the code obtained from $\mathrm{Ham}(r, 2)$ by adding an overall parity-check.

It can be proven by above results that $\mathrm{H\hat{a}m}(r, 2)$ is a $[2^r, 2^r - 1 - r, 4]$-code. It is of no better than $\mathrm{Ham}(r, 2)$ when used for complete decoding, but can be used for incomplete decoding, for it can simultaneously correct any single error and detect any double error.

Now we shall define Hamming codes over arbitrary field $GF(q)$, before that we need to understand the following relationship between minimum distance of a linear code and linear inidependence property between columns of parity-check matrix.

Theorem 6.4. Suppose C is a linear $[n,k]$-code over $GF(q)$ with parity-check matrix H. Then the minimum distance of C is d if and only if any $d-1$ columns of H are linearly independent but some d columns are linearly dependent.

Proof. By Theoem 4.3, the minimum distance of C is equal to the smallest of weights of the non-zero codewords. Let $\mathbf{x} = x_1x_2\cdots x_n$ be a vector in $V(n,q)$. Then

$$\begin{aligned}\mathbf{x} \in C &\iff \mathbf{x}H^T = 0\\ &\iff x_1\mathbf{H}_1 + x_2\mathbf{H}_2 + \cdots + x_n\mathbf{H}_n = \mathbf{0}\end{aligned}$$

where $\mathbf{H}_i$ denote the columns of H.
Therefore, for each codeword $\mathbf{x}$ of weight d, there is a set of d linearly dependent columns of H. On the other hand, if there existed some $d-1$ dependent columns of H, then we would have a codeword, defined by coefficients (not all zero) of those $d-1$ columns, with coeffients being the value of codeword's ith position (where i belongs ti $d-1$ column positions in H), and 0 at other places. Thus, this codeword will have weight $< d$, hence contradiction. $\square$

Now, any non-zero vector $\mathbf{v}$ in $V(r,q)$ has exactly $q-1$ non-zero scalar multiples, forming set $\{\lambda\mathbf{v} | \lambda \in GF(q), \lambda \neq 0\}$. Hence, q^r-1 non-zero vectors of $V(r,q)$ may be partitioned into $(q^r-1)/(q-1)$ vectors, where each set consists of elements that are scalar multiples of other elements of that set. Now by choosing one vector from each set of $(q^r-1)/(q-1)$ vectors, no two of which are linearly dependent. Therfore, by theorem 6.4, taking these as the columns of H gives a parity-check matrix of a $\left[\frac{(q^r-1)}{q-1}, \frac{q^r-1}{q-1} - r, 3\right]$-code. This is called ***q-ary Hamming code*** and is denoted by $\mathrm{Ham}(r,q)$. Here also, $\mathrm{Ham}(r,q)$ is unique upto equivalence.

Example: A parity-check matrix for $\mathrm{Ham}(2,11)$ is

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1\\ 1 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{bmatrix}$$

Theorem 6.5. $\mathrm{Ham}(r,q)$ is a perfect single-error-correcting code.

Proof. $\mathrm{Ham}(r,q)$ was constructed to be an $(n,M,3)$-code with $n = (q^r-1)/(q-1)$ and $M = q^{n-r}$. The left-hand side of the sphere-packing bound becomes

$$\begin{aligned} q^{n-r}(1 + n(q-1)) &= q^{n-r}(1 + q^r - 1) && (1)\\ &= q^n && (2)\end{aligned}$$

which is the right-hand side, so $\mathrm{Ham}(r,q)$ is a perfect code. $\square$

Corollary 6.5.1. If q is a prime power and if $n = (q^r-1)/(q-1)$, for some $r \geq 2$, then

$$A_q(n,3) = q^{n-r}$$

Decoding with a q-ary Hamming code:
Since a Hamming code is a perfect single-error correcting code, the codet leaders, other than $\mathbf{0}$, are precisely the vectors of weight 1. The syndrome of such a coset leader $codex = 0\cdots 0b0\cdots 0$, with non-zero element at the jth place, is

$$S(\mathbf{x}) = b\mathbf{H}_j^T,$$

where $\mathbf{H}_j$ denotes the jth column of H.
So the decoding scheme is as follows. If the received vector is $\mathbf{y}$, then first calculate the syndrome $S(\mathbf{y})$, if $S(\mathbf{y}) = \mathbf{0}$, assume no errors, otherwise $S(\mathbf{y}) = b\mathbf{H}_j^T$ for some b and j and the assumed single error is corrected by subtracting b from the jth entry of $\mathbf{y}$.

§7 Codes and Latin Squares

The main aim of this section is to construct codes using some mathematical constructs called **Latin Squares**, and vice-versa. Further, we will solve the *main coding theory problem* for single-error-correcting codes of length 4 i.e. find the values of $A_q(4,3)$ for all values of q.

Definition 7.1. A **Latin square of order** q is a $q \times q$ array whose entries are from a set F_q of q distinct symbols suzh that each row and each column of the array contains each symbol exactly once.

Example: Let $F_3 = \{1,2,3\}$. Then an example of a Latin square of order 3 is

$$\begin{array}{ccc} 1 & 2 & 3 \\ 2 & 3 & 1 \\ 3 & 1 & 2 \end{array}$$

Theorem 7.2. There exists a Latin square of order q for any positive integer q.

Proof. We can take 1 2 $\cdots$ q as the first row and cycle this round once for each subsequent row to get

$$\begin{array}{cccccccc} 1 & 2 & 3 & \cdots & & & q \\ 2 & 3 & 4 & \cdots & & q & 1 \\ 3 & 4 & 5 & \cdots & q & 1 & 2 \\ \vdots & \vdots & \vdots & & & & \vdots \\ q & 1 & 2 & \cdots & & & q-1 \end{array}$$

Alternatively, the addition table of Z_q is a Latin square of order q. □

Definition 7.3. Let A and B be two Latin squares of order q. Let a_{ij} and b_{ij} denote the i,jth entries of A and B respectively. Then A and B are said to be ***mutually orthogonal*** Latin squares (abbreviated as MOLS) if the q^2 ordered pairs (a_{ij}, b_{ij}), $i,j = 1,2,\ldots,q$ are all distinct.

Example: The Latin squares

$$A = \begin{array}{ccc} 1 & 2 & 3 \\ 2 & 3 & 1 \\ 3 & 1 & 2 \end{array} \quad \text{and } B = \begin{array}{ccc} 1 & 2 & 3 \\ 3 & 1 & 2 \\ 2 & 3 & 1 \end{array}$$

7.1 Optimal single-error-correcting code of length 4

Theorem 7.4. $A_q(4,3) \le q^2$, for all q.

Proof. Suppose C is a q-ary $(4,M,3)$-code and let $\mathbf{x} = x_1x_2x_3x_4$ and $\mathbf{y} = y_1y_2y_3y_4$ be distinct codewords of C. Then $(x_1,x_2) \neq (y_1y_2)$, for otherwise $\mathbf{x}$ and $\mathbf{y}$ could differ only in the last two places, contradicting $d(C) = 3$. Therefore, $M \le q^2$. □

Theorem 7.5. There exists a q-ary $(4,q^2,3)$-code if and only if there exists a pair of MOLS of the order q.

Proof. Let

$$C = \{(i,j,a_{ij},b_{ij}) | (i,j) \in (F_q)^2\}$$

As in the proof of Theorem 7.4, the minimum distance of C is 3 if and only if, for each pair of coordinate positions, the ordered pairs appearing in those positions are distinct. Now the q^2 pairs of (i,a_{ij}) and q^2 pairs of (j,a_{ij}) are distinct if and only if A is a Latin square. Similarly, the q^2 pairs of (i,b_{ij}) and q^2 pairs of (j,b_{ij}) are distinct if and only if B is a Latin square. Lastly, the q^2 pairs (a_{ij},b_{ij}) are distinct if and only if there exists a MOLS of order q. □

Theorem 7.6. If q is a prime-power and $q \neq 2$, then there exists a pair of MOLS of order q.

Proof. Let F_q be the field $GF(q) = \{\lambda_0, \lambda_1, \ldots, \lambda_{q-1}\}$ where $\lambda_0 = 0$. Let μ and ν be two distinct non-zero elements of $GF(q)$. Let $A = [a_{ij}]$ and $B = [b_{ij}]$ be $q \times q$ arrays defined by

$$a_{ij} = \lambda_i + \mu\lambda_j \ \ and \ \ b_{ij} = \lambda_i + \nu\lambda_j$$

We now see that A id Latin square and similarly so is B. As if two elements in the same of A are identical, then we have

$$\lambda_i + \mu\lambda_j = \lambda_i + \mu\lambda_j^{'}$$

implying $j = j^{'}$ as μ is non-zero. Now for columns,

$$\lambda_i + \mu\lambda_j = \lambda_i^{'} + \mu\lambda_j$$

implying that $i = i^{'}$. Now, we prove that A and B are orthogonal, suppose on contrary that $(a_{ij}, b_{ij}) = (a_{i^{'}j^{'}}, b_{i^{'}j^{'}})$, then

$$\begin{aligned} \lambda_i + \mu\lambda_j &= \lambda_i^{'} + \mu\lambda_j^{'} \\ \text{and} \quad \lambda_i + \nu\lambda_j &= \lambda_i^{'} + \nu\lambda_j^{'} \end{aligned}$$

which on subtraction gives

$$(\mu - \nu)\lambda_j = (\mu - \nu)\lambda_j^{'}$$

Since $\mu \neq \nu$, we have $j = j^{'}$, and consequently, $i = i^{'}$. □

Theorem 7.7. If there exists a pair of MOLS of order n as well as order m, then there exists a pair of MOLS of order mn.

Proof. Suppose A_1, A_2 is a pair of MOLS of order m and B_1, B_2 is a pair of MOLS of order n. Let C_1 and C_2 be the $mn \times mn$ squares defined by

$$C_k = \begin{matrix} (a_{11}^{(k)}, B_k) & (a_{12}^{(k)}, B_k) & \cdots & (a_{1m}^{(k)}, B_k) \\ (a_{12}^{(k)}, B_k) & (a_{22}^{(k)}, B_k) & \cdots & (a_{2m}^{(k)}, B_k) \\ \vdots & & & \vdots \\ (a_{m1}^{(k)}, B_k) & (a_{m2}^{(k)}, B_k) & \cdots & (a_{mm}^{(k)}, B_k) \end{matrix}$$

where $k \in \{1, 2\}$, $A_k = [a_{ij}^{(k)}]$ and $(a_{ij}^{(k)}, B_k)$ denotes an $n \times n$ array (referred as block in this proof) whose r, sth entry is $(a_{ij}^{(k)}, b_{rs}^{(k)})$ for $r, s \in \{1, 2, \cdots, n\}$.
From this construction it is trivial to see that C_k are Latin squares. Further assuming them to be not a pair of MOLS implies that in both C_1, C_2 either two entries in a block are same or, two entries are same in blocks having different row and column. The first possibility contradicts B_k being a pair of MOLS, and the latter contradicts A_k being a pair of MOLS. □

Theorem 7.8. If $q \equiv 0, 1$ or $3 \pmod 4$. Then there exists a pair of MOLS of order q.

Proof. One can break down each of q satisfying $q \equiv 0, 1$ or $3 \pmod 4$ into their prime factorisation, then each of the prime-power in it will be ≥ 3. Thus, repeated application of Theorem 7.6 and Theorem 7.7 will give us the required pair of MOLS for each of these q. □

Note: Theorem 7.8 leaves cases when $q \equiv 2 \pmod 4$. It has been proved pair of MOLS also exist for these cases except for $q = 2$ and $q = 6$. The proof will not be covered in this report.

Corollary 7.8.1. $A_q(4, 3) = q^2$ for all $q \neq 2, 6$.

Proof. This is immediate from Theorems 7.4, 7.5 and 7.8. □

Remark. For $q = 2$, it is trivial to see that $A_2(4, 3) = 2$, while for $q = 6$, a construction similar to pair of orthogonal Latin squares gives $A_6(4, 3) = 34$.

Some generalization of the above results.

Theorem 7.9 (Singleton bound).

$$A_q(n,d) \leq q^{n-d+1}.$$

Proof. Suppose C is a q-ary (n,M,d)-code.Same as in proof of Theorem 7.4, if now we delete the last $d-1$ coordinates from each codeword, then the M vectors of length $n-d+1$ so obtained must be distinct and so $M \leq q^{n-d+1}$. □

Definition 7.10. A set $\{A_1, A_2, \ldots, A_t\}$ of Latin squares of order q is called a set of mutually orthogonal Latin squares (MOLS) if each pair $\{A_i, A_j\}$ is a pair of MOLS, for $1 \leq i < j \leq t$.

Theorem 7.11. There are at most $q-1$ Latin squares in any set of MOLS of order q.

Proof. Let $A_1, A_2, \ldots, A_t$ be the set of MOLS of order q. If we relabel elements of each Latin square such that the first row of A_i is $1\ 2\ \cdots q$ (as relabelling conserves orthogonality). Now considering t entries appearing in the $(2,1)$th positions cannot be 1 as well as no two of them can be same, as for all i, the pair (i,i) has already occured in the first row.
Therefore, we must have $t \leq q-1$. □

Definition 7.12. If a set of $q-1$ MOLS of order q exists, it is called a *complete* set of MOLS of order q.

Theorem 7.13. If q is prime-power, then there exists a complete set of $q-1$ MOLS of order q.

Proof. Similar to what we in Theorem 7.6, if we define $A_k = [a_{ij}^{(}k)], k \in \{1,2,\ldots,q-1\}$, with

$$a_{ij}^{(k)} = \lambda_i + \lambda_k \lambda_j.$$

It follows exactly as in the proof of Theorem 7.6, that the set formed by the Latin squares A_k is a set of MOLS of order q. □

Theorem 7.14. A q-ary $(n, q^2, n-1)$-code is equivalent to a set of $n-2$ MOLS of order q.

Proof. As in Theorem 7.5, code C of the form

$$\{(i,j,a_{ij}^{(1)},a_{ij}^{(2)},\ldots,a_{ij}^{(n-2)}) | (i,j) \in (F_q)^2\}$$

has $d(C) = n-1$ if and only if $A_k = [a_{ij}^{(k)}]$, form a set of MOLS of order q, same outline as in proof of Theorem 7.5. □

Corollary 7.14.1. If q is a prime power and $n \leq q+1$, then

$$A_q(n, n-1) = q^2$$

Proof. This is immediate from above theorems. □

§8 Basic BCH codes

Before we get into BCH codes, we will need some basic results from Linear Algebra. Therefore, we will first get introduced with some Linear Algebra results (proofs will be omitted), and then define generalised version of a particular class of BCH codes, along with their decoding procedure.

8.1 Preliminary results from linear algebra

Theorem 8.1. Suppose $a_1, a_2, \ldots, a_r$ are distinct non-zero elements of a field. Then the determinant of the following matrix A (called *Vandermonde matrix*) is non-zero and is given by $\prod_{i>j}(a_i - a_j)$.

$$A = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ a_1 & a_2 & \cdots & a_r \\ a_1^2 & a_2^2 & \cdots & a_r^2 \\ \vdots & \vdots & & \vdots \\ a_1^{r-1} & a_2^{r-1} & \cdots & a_r^{r-1} \end{bmatrix} \text{ and } det(A) = \prod_{i>j}(a_i - a_j)$$

Theorem 8.2. If A is an $r \times r$ matrix having a non-zero determinant, then the r columns of A are linearly independent (converse is also true).

8.2 A class of BCH codes

Now we will give construction for t-error-correcting code of length n over $GF(q)$, provided

$$2t + 1 \leq n \leq q - 1.$$

We will assume for simplicity that q is a prime, so that $GF(q) = \{0, 1, \ldots, q-1\}$, but this construction can be generalized for any prime-power by relabelling accordingly.
Let C be the code over $GF(q)$ defined to have the parity-check matrix

$$H = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & 2 & 3 & \cdots & n \\ 1 & 2^2 & 3^2 & \cdots & n^2 \\ \vdots & & & & \\ 1 & 2^{d-2} & 3^{d-2} & \cdots & n^{d-2} \end{bmatrix}$$

where $d \leq n \leq q-1$, i.e.

$$C = \left\{ x_1 x_2 \cdots x_n \in V(n, q) \mid \sum_{i=1}^{n} i^i x_i = 0 \text{ for } j = 0, 1, \ldots, d-2 \right\}.$$

Theorem 8.3. If q is a prime-power and if $d \leq n \leq q-1$, then

$$A_q(n, d) = q^{n-d+1}.$$

Proof. Any $d-1$ columns of H (defined above) form a Vandermonde matrix and so are linearly independent, by Theorems 8.1 and 8.2. Also, any d columns of H corresponds to solving $d-1$ linear equations in d variables, which implies the linear dependence of those d columns. Hence, by Theorem 6.4, C has a minimum distance of d and so is a q-ary (n, q^{n-d+1}, d)-code and it meets the Singleton bound (Theorem 7.9). □

Decoding algorithm:
Suppose the codeword $\mathbf{x} = x_1 x_2 \cdots x_n$ is transmitted and that the vector $\mathbf{y} = y_1 y_2 \cdots y_n$ is received in which we assume atmost t errors have occurred, at positions $X_1, X_2, \ldots, X_t$ with respective magnitudes $m_1, m_2, \ldots, m_t$ (if $e < t$ errors have occurred then we assume other magnitudes to be zero). From $\mathbf{y}$ we calculate the syndrome

$$(S_1, S_2, \ldots, S_{2t}) = \mathbf{y}H^T,$$

Thus, we must solve for X_i and m_i the following equations

$$\left.\begin{array}{llll} m_1 + m_2 & + \cdots + m_t & = S_1 \\ m_1X_1 + m_2X_2 & + \cdots + m_tX_t & = S_2 \\ m_1X_1^2 + m_2X_2^2 & + \cdots + m_tX_t^2 & = S_3 \\ \vdots & & \\ m_1X_1^{2t-1} + m_2X_2^{2t-1} & + \cdots + m_tX_t^{2t-1} & = S_2t. \end{array}\right\} \tag{3}$$

Now consider the expression

$$\phi(\theta) = \frac{m_1}{1 - X_1\theta} + \frac{m_2}{1 - X_2\theta} + \cdots + \frac{m_t}{1 - X_t\theta} \tag{4}$$

Now as $\dfrac{m_j}{1 - X_j\theta} = m_j(1 + X_j\theta + X_j^2\theta^2 + \cdots)$,
and therefore by combination of set of eqautions (3) and equation 4, we get

$$\phi(\theta) = S_1 + S_2\theta + S_3\theta^2 + \cdots + S_{2t}\theta^{2t-1} + \cdots \tag{5}$$

Reducing the fractions in (4), we have

$$\phi(\theta) = \frac{A_1 + A_2\theta + A_3\theta^2 + \cdots + A_t\theta^{t-1}}{1 + B_1\theta + B_2\theta^2 + \cdots + B_t\theta^t}. \tag{6}$$

Hence,

$$(S_1 + S_2\theta + S_3\theta^2 + \cdots)(1 + B_1\theta + B_2\theta^2 + \cdots + B_t\theta^t) = A_1 + A_2\theta + A_3\theta^2 + \cdots + A_t\theta^{t-1}.$$

Equationg coefficients of like powers of θ, we have

$$\left.\begin{array}{l} A_1 = S_1 \\ A_2 = S_2 + S_1B_1 \\ \quad\vdots \\ A_t = S_t + S_{t-1}B_1 + S_{t-2}B_2 + \cdots + S_tB_{t-1} \end{array}\right\} \tag{7}$$

$$\left.\begin{array}{l} 0 = S_{t+1} + S_tB_1 + S_{t-1}B_2 + \cdots + S_1B_t \\ 0 = S_{t+2} + S_{t+1}B_1 + S_tB_2 + \cdots + S_2B_t \\ \quad\vdots \\ 0 = S_{2t} + S_{2t-1}B_1 + S_{2t-2}B_2 + \cdots + S_tB_t \end{array}\right\} \tag{8}$$

Since S_is are known, the t equations (8) gives us B_is, and then A_is can be found from equations in (7).
After this, we can split equation (6) into partial fractions and compare with equation (4) to get corresponding m_i and X_i.

Note: By directly looking at the syndrome vector, we cannot predict the number errors which have actually occured, but the number of actual errors e will be same as the maximum number of linearly independent equations in system (8). After finding e, we can substitute $B_{e+1}, B_{e+2}, \ldots, B_t$ all equal to zero, and the denominator in equation (6), becomes

$$1 + B_1\theta + B_2\theta^2 + \cdots + B_e\theta^e$$

which can be factorised to give the e errors and their location.

§9 Cyclic codes

9.1 Definition

Cyclic codes form an important type of codes for several theoritical as well as practical reasons. From theoritical perspective, they can be expressed as a rich algebraic structure, while practically they can be efficiently implemented. Furthermore, various important codes such as binary Hamming codes and BCH codes, are equivalent to cyclic codes.

Definition 9.1. A code C is ***cyclic*** if(i) C is a linear code; and (ii) any cyclic shift of a codeword is also a codeword, i.e. whenever $a_0a_1\cdots a_{n-1}$ is in C, then so is $a_{n-1}a_0a_1\cdots a_{n-2}$.

Example:
The linear code $\{0000,\ 1001,\ 0110,\ 1111\}$ is not cyclic, but it is *equivalent* to a cyclic code; interchanging the third and fourth coordinates gives the cyclic code $\{0000,\ 1010,\ 0101,\ 1111\}$.

When considering cyclic codes we number the coordinate positions $0, 1, \ldots, n-1$. This is because then a vector $a_0a_1\cdots a_{n-1}$ in $V(n,q)$ correspond to the polynomial $a_0 + a_1x + \cdots + a_{n-1}x^{n-1}$.

9.2 Polynomials over finite fields

We denote by $F[x]$ the set of polynomials in x with coefficients in F_q (or simply F with q understood).

Degree of a polynomial in $F[x]$ is defined as usual. These polynomials can be added, subtracted and multiplied in the usual way, thus forming the algebraic structure, *ring*, not a field as they do not have multiplicative inverses.
Division algorithm states the same as in with the polynomials over $\mathbb{R}$, i.e. for every pair of polynomials $a(x)$ and $b(x) \neq 0$ in $F[x]$, there exists a unique polynomials $q(x)$, and $r(x)$, such that

$$a(x) = q(x)b(x) + r(x),$$

where deg $r(x) <$ deg $b(x)$.

Now we will establish similarites between the ring $F[x]$ of polynomials and the ring $\mathbb{Z}$ of integers.
Just as the ring $\mathbb{Z}_m$ is obtained after modulo m, we can consider polynomials in $F[x]$ modulo some $f(x)$. It is natural to define $g(x)$ and $h(x)$ as *congruent modulo* $f(x)$, symbolized by

$$g(x) \equiv h(x) \pmod{f(x)}$$

if $g(x) - h(x)$ is divisible by $f(x)$.

We denote by $F[x]/f(x)$ the set of polynomials in $F[x]$ of degree less than deg $f(x)$, with addition defined as normal addition as adding two polynomials in $F[x]/f(x)$ will not increase the degree; while multiplication is defined congruent modulo $f(x)$, i.e for any two polynomials in $F[x]/f(x)$, a unique polynomial of degree less than deg $f(x)$.

Finally, the set $F[x]/f(x)$ is called a *ring of polynomials(over F) modulo $f(x)$* and it is quite obvious that

$$|F_q[x]/f(x)| = q^n.$$

Now the next logical step is to find out when this ring forms a field, i.e. when each of the polynomials in $F[x]/f(x)$ have a multiplicative inverse.
For that we will first define *reducibility* of polynomials.

Definition 9.2. A polynomial $f(x)$ is called ***reducible*** if $f(x) = a(x)b(x)$, where $a(x), b(x) \in F[x]$ and deg $a(x)$, deg $b(x)$ are both smaller than deg $f(x)$. If $f(x)$ is not *reducible*, it is called ***irreducible***.

Theorem 9.3. The ring $F[x]/f(x)$ is a field if and only if $f(x)$ is irreducible in $F[x]$.

Proof. The proof follows the same line as followed by the proof of Theorem 3.4, with prime m being replaced by $f(x)$. □

9.3 Cyclic codes expressed as polynomials

From now on, we will fix $f(x) = x^n - 1$, as the ring $F[x]/(x^n - 1)$ of the polynomials modulo $x^n - 1$ is the natural one to consider in the context of cyclic codes. We will now denote $F[x]/(x^n - 1)$ as R_n, where the field $F = F_q$ will be understood.
Since $x^n \equiv 1 \pmod{x^n - 1}$, we can reduce any polynomial modulo $x^n - 1$ simply by replacing x^k by $x^{k \pmod n}$, without any long division.

We will now denote a vector $a_0 a_1 \cdots a_{n-1}$ in $V(n, q)$ with the polynomial

$$a(x) = a_0 + a_1 x + \cdots + a^{n-1} x^{n-1}$$

in R_n, that is, now a code C is subset of both $V(n, q)$ and R_n. Note that addition of vectors and multiplication of a vector by a scalar in R_n corresponds exactly to those operations in $V(n, q)$.
Now, it must be clear that *multiplying* by x^m to $a(x)$ corresponds to a cyclic shift through m positions. We can model cyclic codes in a way, given by the following theorem.

Theorem 9.4. A code C in R_n is a cyclic code if and only if C satisfies the following two conditions:

(i) $a(x), b(x) \in C \implies a(x) + b(x) \in C$,

(ii) $a(x) \in C$ and $r(x) \in R_n \implies r(x)a(x) \in C$.

Proof. Suppose C is a cyclic code in R_n. Then C is linear and so (i) holds. Now suppose $a(x) \in C$ and $r(x) = r_0 + r_1 x + \cdots + r_{n-1} x^{n-1} \in R_n$. Since, $x^m a(x) \in C \forall m$ (cyclic shifts). Hence,

$$r(x)a(x) = r_0 a(x) + r_1 x a(x) + \cdots + r_{n-1} x^{n-1} a(x)$$

is also in C since each summand is in C. Thus, (ii) also holds.
Now suppose (i) and (ii) hold. Taking $r(x)$ as a scalar, the conditions imply that C is linear. Taking $r(x) = x$ in (ii) shows that C is cyclic. □

Now we have a easy way of constructing cyclic codes. Let $f(x)$ be any polynomial in R_n, then we define

$$\langle f(x) \rangle = \{ r(x) f(x) | r(x) \in R_n \}$$

$\langle f(x) \rangle$ is a cyclic code for all $f(x)$ in R_n, as it satisfies the conditions of Theorem 9.4.

Theorem 9.5. Let C be a non-zero cyclic code in R_n. Then

(i) there exists a unique monic (coefficient of highest degree term 1) polynomial $g(x)$ of smallest degree in C,

(ii) $C = \langle g(x) \rangle$,

(iii) $g(x)$ is a factor of $x^n - 1$.

Proof. (i) Suppose $g(x)$ and $h(x)$ are both monic polynomials in C of the smallest degree. Then $g(x) - h(x) \in C$ and has smaller degree. This gives a contradiction if $g(x) \neq h(x)$, for then a suitable scalar multiple of $g(x) - h(x)$ is monic.
(ii) Suppose $a(x) \in C$. By the division algorithm for $F[x]$, $a(x) = q(x)g(x) + r(x)$, where $\deg r(x) < \deg g(x)$. But $r(x)$, belongs to C, as C is linear. By minimality of $\deg g(x)$, we must have $r(x) = 0$ and so $a(x) \in \langle g(x) \rangle$.
(iii) Again by division algorithm, $x^n - 1 = q(x)g(x) + r(x)$ where $\deg r(x) < g(x)$. But then $r(x) \equiv -q(x)g(x) \pmod{x^n - 1}$, and so $r(x) \in \langle g(x) \rangle$. By minimality again, $r(x) = 0$, which implies $g(x)$ is a factor of $x^n - 1$. □

Definition 9.6. In a non-zero cyclic code C the monic polynomial of least degree, given by theorem 9.5, is called the ***generator polynomial*** of C.

The third part of Theorem 9.5 gives us method of finding all the cyclic codes of length n. All we need are the factors of $x^n - 1$.

Example:
Finding all the binary cyclic codes of length 3. We have $x^3 - 1 = (x+1)(x^2+x+1)$, where $x+1$ and x^2+x+1 are irreducible over $GF(2)$. So the codes are

Generator ploynomial	Code in R_3	Corresponding code in $V(3,2)$
1	all of R_3	all of $V(3,2)$
$x+1$	$\{0, 1+x, x+x^2, 1+x^2\}$	$\{000, 110, 011, 101\}$
x^2+x+1	$\{0, 1+x+x^2\}$	$\{000, 111\}$
$x^3-1=0$	$\{0\}$	$\{000\}$

Lemma 9.7. Let $g(x) = g_0 + g_1x + \cdots + g_rx^r$ be the generator polynomial of a cyclic code, then g_0 is non-zero.

Proof. Suppose $g_0 = 0$. Then $x^{n-1}g(x) = x^{-1}g(x)$ is a codeword of C of degree $r-1$, contradicting the minimality of deg $g(x)$. □

Now, we are going to define generator matrix (thus, dimensions of C) directly from generator polynomial.

Theorem 9.8. Suppose C is a cyclic code with generator polynomial

$$g(x) = g_0 + g_1x + \cdots + g_rx^r$$

of degree r. Then dim $(C) = n - r$ and a generator matrix C is

$$G = \begin{bmatrix} g_0 & g_1 & g_2 & \cdots & & g_r & 0 & 0 & \cdots & 0 \\ 0 & g_0 & g_1 & g_2 & \cdots & & g_r & 0 & \cdots & 0 \\ 0 & 0 & g_0 & g_1 & g_2 & \cdots & & g_r & & \vdots \\ \vdots & \vdots & & \ddots & \ddots & \ddots & & & \ddots & 0 \\ 0 & 0 & \cdots & 0 & g_0 & g_1 & g_2 & \cdots & & g_r \end{bmatrix}$$

Proof. The $n-r$ rows of the above matrix G are certainly linearly independent due echelon of non-zero g_0s. The $n-r$ rows represent the cyclic permutations of codeword $g(x)$. The proof of Theorem 9.5(ii) shows that if $a(x)$ is a codeword of C, then

$$a(x) = q(x)g(x)$$

for some polynomial $q(x)$, and that this is an equality of polynomials within $F[x]$, i.e. without any modulo. Since deg $a(x) < n$, it follows that deg $q(x) < n - r$. Hence,

$$\begin{aligned} q(x)g(x) &= (q_0 + q_1x + \cdots + q_{n-r-1}x^{n-r-1})g(x) \\ &= q_0g(x) + q_1xg(x) + \cdots + q_{n-r-1}x^{n-r-1}g(x), \end{aligned}$$

which shows that every codeword can be written as a linear combination of those $n-r$ rows. □

Definition 9.9. Let C be a cyclic $[n,k]$-code with generator polynomial $g(x)$. By Theorem 9.5 $g(x)$ is a factor of $x^n - 1$ and so

$$x^n - 1 = g(x)h(x),$$

for some polynomial $h(x)$. $h(x)$ is of degree k, and is called the ***check-polynomial*** of C.

Theorem 9.10. Suppose C is a cyclic code in R_n with generator polynomial $g(x)$ and check polynomial $h(x)$. Then $c(x) \in R_n$ is a codeword in C if and only if $c(x)h(x) = 0$.

Proof. The forward implication is trivial as $g(x)h(x) = 0$. On the other hand, suppose $c(x)$ satisfies $c(x)h(x) = 0$. If $r(x)$ is the remainder of $c(x)$ with $g(x)$ then $r(x)h(x) = 0 \pmod{x^n - 1}$. But deg $(r(x)h(x)) < n - k + k = n$, so $r(x) = 0$, then $c(x) = q(x)g(x) \in C$. □

Theorem 9.11. Suppose C is a cyclic $[n,k]$-code with check polynomial

$$h(x) = h_0 + h_1x + \cdots + h_kx^k$$

Then a parity check matrix for C is

$$H = \begin{bmatrix} h_k & h_{k-1} & \cdots & h_0 & 0 & 0 & \cdots & 0 \\ 0 & h_k & h_{k-1} & \cdots & h_0 & 0 & \cdots & 0 \\ & & \ddots & \ddots & & & \ddots & 0 \\ 0 & \cdots & 0 & h_k & h_{k-1} & \cdots & & h_0 \end{bmatrix}$$

Proof. By Theorem 9.10, $c(x)= c_0 + c_1x + \cdots + c_{n-1}x^{n-1}$ is codeword if and only if $c(x)h(x) = 0$. Thus any codeword $c_0c_1\cdots c_{n-1}$ of C is orthogonal to the vector $h_kh_{k-1}\cdots h_0 \cdots 0$ and to its cyclic shifts (this results from equating coefficients of $x^k, x^{k+1}, \ldots, x^{n-1}$ must all be zero in $c(x)h(x)$). Thus all the $n-k$ rows are orthogonal to all the codewords and are linearly independent (becuase echelon of $h_k = 1$ in H), and the dimension of $C^{\perp}$ is also $n-k$. Hence, H is a generator matrix of $C^{\perp}$. □

§10 Weight enumerators

Definition 10.1. If C is a linear $[n,k]$-code, its ***weight enumerator*** is defined to be the polynomial

$$\begin{aligned} W_C(z) &= \sum_{i=0}^{n} A_i z^i \\ &= A_0 + A_1 z + \cdots + A_n z^n, \end{aligned}$$

or simply,

$$W_C(z) = \sum_{\mathbf{x}\in C} z^{w(\mathbf{x})}.$$

Example: Let $C = \{000, 011, 101, 110\}$. Its dual code $C^{\perp}$ is $\{000, 111\}$. The weight enumerators of C and $C_{\perp}$ are

$$W_C(z) = 1 + 3z^2, \qquad W_{C^{\perp}}(z) = 1 + z^3$$

The motto behind finding weight enumerator of a code is that it enables us find the probabilty of undetected errors when the code is purely used for error-detection.
Also, the objective of this section will be to find weight enumerator of any binary linear code C to be obtained from the weight enumerator of its dual code $C^{\perp}$, as the enumerator of the latter is much easier to find in cases when n, k are both large, while $n-k$ is relatively small.
In order to reach this result, we will need the following lemmas.

Lemma 10.2. Let C be a binary linear $[n,k]$-code and $\mathbf{y}$ is a fixed vector in $V(n,2)$ and $\mathbf{y} \notin C^{\perp}$. Then $\mathbf{x}\cdot\mathbf{y}$ is equal to 0 and 1 equally often as $\mathbf{x}$ runs over the codewords of C.

Proof. Let $A = \{\mathbf{x} \in C \mid \mathbf{xy} = 0\}$ and $B = \{\mathbf{x} \in C \mid \mathbf{xy} = 1\}$.
There exists $\mathbf{u} \in C$ such that $\mathbf{u}\cdot\mathbf{y} = 1$ as $\mathbf{y} \notin C^{\perp}$. Let $\mathbf{u} + A$ denote the set $\{\mathbf{u}+\mathbf{x} | \mathbf{x} \in A\}$. Then it follows that

$$\mathbf{u} + A \subseteq B,$$

Similarly,

$$\mathbf{u} + B \subseteq A.$$

Hence,

$$|A| = |\mathbf{u} + A| \leq |B| = |\mathbf{u} + B| \leq |A|.$$

□

Lemma 10.3. Let C be a binary $[n,k]$-code and let $\mathbf{y}$ be any element of $V(n,2)$. Then

$$\sum_{\mathbf{x}\in C} (-1)^{\mathbf{x}\cdot\mathbf{y}} = \begin{cases} 2^k & \text{if } \mathbf{y} \in C^{\perp} \\ 0 & \text{if } \mathbf{y} \notin C^{\perp} \end{cases}.$$

Proof. If $\mathbf{y} \in C^{\perp}$, then $\mathbf{x}\cdot\mathbf{y} = 0$ for all $\mathbf{x} \in C$, so the result follows.
On the other hand, if $\mathbf{y} \notin C^{\perp}$, then by Lemma 10.2, $(-1)^{\mathbf{x}\cdot\mathbf{y}}$ is equal to 1 and -1 equally often. □

Lemma 10.4. Let $\mathbf{x}$ be a fixed vector in $V(n,2)$ and z be an indeterminate. Then the following polynomial identity holds:

$$\sum_{\mathbf{y}\in V(n,2)} z^{w(\mathbf{y})}(-1)^{\mathbf{x}\cdot\mathbf{y}} = (1-z)^{w(\mathbf{x})}(1+z)^{n-w(\mathbf{x})}.$$

Proof.

$$\begin{aligned} \sum_{\mathbf{y}\in V(n,2)} z^{w(\mathbf{y})}(-1)^{\mathbf{x}\cdot\mathbf{y}} &= \sum_{y_1=0}^{1} \cdots \sum_{y_n=0}^{1} \left(\prod_{i=1}^{n} z^{y_i}(-1)^{x_i y_i} \right) \\ &= \prod_{i=1}^{n} \left(\sum_{j=0}^{1} z^j (-1)^{j x_i} \right) \\ &= (1-z)^{w(\mathbf{x})}(1+z)^{n-w(\mathbf{x})}, \end{aligned}$$

since, $$\sum_{j=0}^{1} z^j(-1)^{jx_i} = \begin{cases} 1+z & \text{if } x_i = 0 \\ 1-z & \text{if } x_i = 1 \end{cases}.$$ □

Theorem 10.5 (MacWilliams identity). If C is a binary $[n,k]$-code with dual code $C^\perp$, then

$$W_{C^\perp}(z) = \frac{1}{2^k}(1+z)^n W_C\left(\frac{1-z}{1+z}\right).$$

Proof. We will express

$$f(z) = \sum_{\mathbf{x}\in C}\left(\sum_{\mathbf{y}\in V(n,2)} z^{w(\mathbf{y})}(-1)^{\mathbf{x}\cdot\mathbf{y}}\right)$$

in two ways.
Firstly, by Lemma 10.4,

$$\begin{aligned} f(z) &= \sum_{\mathbf{x}\in C}(1-z)^{w(\mathbf{x})}(1+z)^{(n-w(\mathbf{x}))} \\ &= (1+z)^n \sum_{\mathbf{x}\in C}\left(\frac{1-z}{1+z}\right)^{w(\mathbf{x})} \\ &= (1+z)^n W_C\left(\frac{1-z}{1+z}\right) \end{aligned}$$

Secondly, reversing the order of summation gives

$$\begin{aligned} f(z) &= \sum_{\mathbf{y}\in V(n,2)} z^{w(\mathbf{y})}\left(\sum_{\mathbf{x}\in C}(-1)^{\mathbf{x}\cdot\mathbf{y}}\right) \\ &= \sum_{\mathbf{y}\in C^\perp} z^{w(\mathbf{y})}2^k \qquad \text{(by Lemma 10.3)} \\ &= 2^k W_{C^\perp}(z) \end{aligned}$$

Equating these two results gives the expression in theorem. □

More useful form of Theorem 10.5 is when the C and the $C^\perp$ are interchanged, (corresponding change in k also).

Example: If we have $C = \{000, 011, 101, 110\}$ and $C^\perp = \{000, 111\}$ as in earlier example, we have $W_{C^\perp}(z) = 1 + z^3$.
By theorem 10.5, we get

$$\begin{aligned} W_C = \frac{1}{2}(1+z)^3 W_{C^\perp}\left(\frac{1-z}{1+z}\right) &= \frac{1}{2}[(1+z)^3 + (1-z)^3] \\ &= 1 + 3z^2, \end{aligned}$$

The idea of finding W_C with the help of $W_{C^\perp}$ using Theorem 10.5 works really well in the case of binary Hamming code $\text{Ham}(r,2)$ because hamming codes have large number of codewords even for small values of r, so directly calculating W_C is not feasible. Now, we will see why finding $W_{C^\perp}$ first is much more easier in this case.

Theorem 10.6. Let C be the binary Hamming code $\text{Ham}(r,2)$. Then every non-zero codeword of $C^\perp$ has weight 2^{r-1}.

Proof. Let

$$H = \begin{bmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \vdots \\ \mathbf{h}_2 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1n} \\ h_{21} & h_{22} & \cdots & h_{2n} \\ \vdots & \vdots & & \vdots \\ h_{r1} & h_{r2} & \cdots & h_{rn} \end{bmatrix}$$

be the parity-check matrix of C where $\mathbf{h}_i$ are the rows. Let a non-zero codeword $\mathbf{c} = \sum_{i=1}^{r} \lambda_i \mathbf{h}_i$ of $C^{\perp}$. Since C is a Hamming code, columns of H are precisely the non-zero vectors of $V(r, 2)$, so number of zero coordinates ($n_0(\mathbf{c})$) are equal to non-zero elements of the set

$$X = \left\{ x_1 x_2 \cdots x_r \in V(r,2) \;\middle|\; \sum_{i=1} r\lambda_i x_i = 0 \right\}$$

i.e. $n_0(\mathbf{c}) = |X| - 1$. Now X is a $r-1$-dimensional subspace of $V(r, 2)$ (we can view it as a dual code of a code of 1-dimension).
So, $n_0(\mathbf{c}) = 2^{r-1} - 1$, which is independent of $\mathbf{c}$. Therfore,

$$\begin{aligned} w(\mathbf{c}) = n - n_0(\mathbf{c}) &= 2^r - 1 - (2^{r-1} - 1) \\ &= 2^{r-1} \end{aligned}$$

□

The combined result of theorems 10.5 and 10.6 gives the weight enumerator of binary Ham$(r, 2)$, of length $n = 2^r - 1$, as

$$\frac{1}{2^r}[(1+z)^n + n(1-z^2)^{(n-1)/2}(1-z)].$$

Also, the probability of undetected errors in a binary $[n, k]$-code C given by Theorem 4.14, becomes

$$P_{\text{undetec}}(C) = (1-p)^n \left[W_C\left(\frac{p}{1-p}\right) - 1 \right]$$

or by using Theorem 10.5,

$$P_{\text{undetec}}(C) = \frac{1}{2^{n-k}}[W_{C^{\perp}}(1-2p) - (1-p)^n].$$

References

[1] Raymond Hill. *A First Course in Coding Theory.* Oxford University Press, 1986.

[2] J.H. van Lint. *Introduction from Coding Theory.* Springer, 1991.