Summer of Science, 2020

MnP Club, IIT Bombay

Coding Theory

- Mid-term Report -

Abhinav Gupta [1]

April 30, 2020

[1] Mentored by Shourya Pandey

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 $\implies A_q(n, d) \leq q$. But there exists q-ary repetition code of length n, $\begin{bmatrix} 00\dots0 \\ 11\dots1 \\ \vdots \\ qq\dots q \end{bmatrix}$, 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 $\mathbf{0} = 00\dots0$.

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 $\mathbf{x} = x_1 x_2 \cdots x_n$, $\mathbf{y} = y_1 y_2 \cdots y_n \in (F_2)^n$, we define the following operations, namely, the ***sum*** $\mathbf{x} + \mathbf{y}$ is the vector in $(F_2)^n$ defined as

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

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

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

The terms $x_i + y_i$ and $x_i y_i$ 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 $\mathbf{x} = x_1 x_2 \cdots x_n$ 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

$$\hat{C} = \{\hat{\mathbf{x}} \mid \hat{\mathbf{x}} = x_1 x_2 \cdots x_n x_{n+1} \forall\ \mathbf{x} \in C\}$$

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 $A_2(n, d) \leq 2\left\lfloor \dfrac{d}{2d-n} \right\rfloor$.
(ii) if d is odd and $n < 2d+1$, then $A_2(n, d) \leq 2\left\lfloor \dfrac{d+1}{2d+1-n} \right\rfloor$.
(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, $A_2(n, d) = A_2(n+1, d+1) = 2\left\lfloor \dfrac{d+1}{2d+1-n} \right\rfloor$(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 $A_2(2d-1, d) \leq 2\left\lfloor \dfrac{d}{1} \right\rfloor \implies A_2(2d, d) \leq 4d$

(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 $a, 2a, 3a, \ldots, (m-1)a$, 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 $i, j \in \{1, 2, \ldots, m-1\}, i \neq j$, 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 $\{textbffv_1, \mathbf{v}_2, \ldots, \mathbf{v}_r\}$ 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.

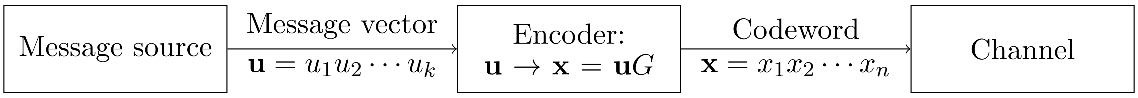
Message source
Message vector
$\mathbf{u} = u_1 u_2 \cdots u_k$
Encoder:
$\mathbf{u} \to \mathbf{x} = \mathbf{u}G$
Codeword
$\mathbf{x} = x_1 x_2 \cdots x_n$
Channel

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 $i \in \{1, 2, \ldots, n\}$, 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. □

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 $P_{\text{corr}}(C)$ 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. □

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 Vn, 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. $\Longrightarrow$: This part is obvious as rows of G are codewords.
$\Longleftarrow$: 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 be have 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

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

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 quite 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.

Updated PoA

I have mostly covered what I had planned till midterm (adding worked out problems is left), so I'll continue with the original plan i.e. trying to complete the text, *A first Course in Coding Theory*[**?**], with some help from *Introoduction to Coding Theory* [**?**].

- Week 4 : *May 2, 2020 - May 8, 2020*
  - Chapter 8
  - Chapter 9
- Week 5 : *May 9, 2020 - May 15, 2020*
  - Chapter 9
  - Chapter 10
- Week 6 : *May 16, 2020 - May 22, 2020*
  - Chapter 11
  - Chapter 12
- Week 7 : *May 23, 2020 - May 29, 2020*
  - Chapter 13
  - Chapter 14

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.