CHAPTER III : LINEAR BLOCK CODES ("SHORT")

Linearity is not a prerequisite to obtain good codes. All we need is to have a unique codewords for each message block and of course, should be able to correct errors. But for an arbitrary collection of codewords, encoding and decoding complexity is prohibitive. Linearity makes both operations tractable.

Agenda for linear block codes

- Representation & Encoding
- Decoding
- Performance Analysis
- Bounds on performance

Definition (Linear Block Codes): A block code of length n and 2^k codewords is called a *linear (n,k) code* if and only if its 2^k codewords form a k-dimensional subspace of the vector space $\mathbb{V} = \mathbb{F}^n$.

Basically, the sum of two codewords is also a codeword. In fact, linear combination of any number of codewords is also a codeword.

What does this buy us? A lot of structure!!

Since (n,k) linear code C is a k-dimensional subspace of the vector space $\mathbb{F}^n$, we know that it has a basis set. In fact that basis set can be k linearly independent codewords

$$\underline{g}_0, \underline{g}_1, \cdots, \underline{g}_{k-1}$$

Since they form a basis, every codeword is a linear combination of these k codewords

$$\underline{V} = U_0 \underline{g}_0 + U_1 \underline{g}_1 + \cdots + U_{k-1} \underline{g}_{k-1}$$

where $U_i \in \{0, 1\}$. Now we know how to arrange the above operation in vector-matrix notation.

$$G = \begin{bmatrix} \underline{g}_0 \\ \underline{g}_1 \\ \vdots \\ \underline{g}_{k-1} \end{bmatrix} \qquad \underline{U} = [U_0 \; U_1 \cdots \; U_{k-1}]$$

GENERATOR MATRIX

$$\underline{V} = \underline{U} \cdot G \quad \leftarrow \text{code}$$

$\underline{V}$: codeword; $\underline{U}$: message

Now this leads to an encoding rule.

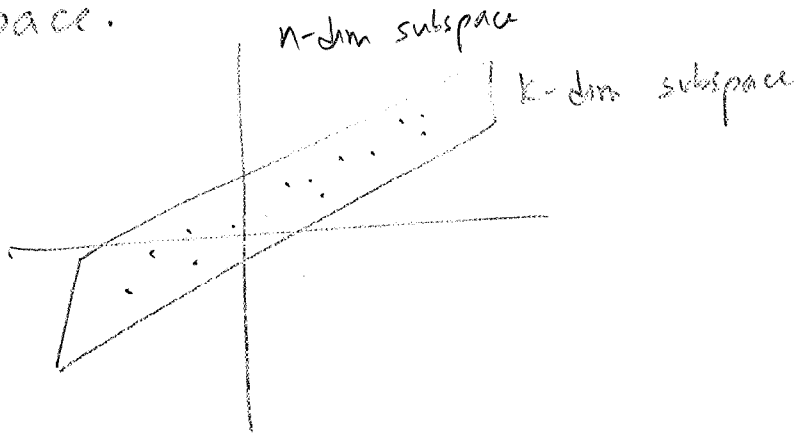
- Elements of $\underline{U}$ can be either 0 or 1. Since $\underline{U}$ has K elements, 2^k possible values for vector $\underline{U}$
  $= \#$ of possible messages

- For any $\underline{U}$, $\underline{U} \cdot G$ generates a valid codeword
- If $\underline{g}_i$'s are linearly independent, then all the codewords are unique. Why?
  Why can't $\underline{U}_1 \cdot G = \underline{U}_2 \cdot G$?

Cost savings in terms of encoding:

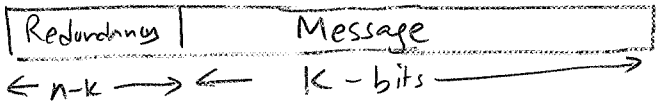
For a codebook with no structure, we need to store 2^k codewords. For a linear code, we need to store only K codewords and use some additions/multiplications.

Possible loss in performance? Yes since you are now constrained to be on a vector subspace instead of moving around freely on the whole vector space.



This kind of tradeoff will be very typical throughout all codes. To reduce complexity of encoding & decoding, we have to give up some performance.

Due to the linearity of the codes, the encoding can be made such that the codewords have two distinct parts: the message part and the redundant parts.



Such a block code is called

LINEAR SYSTEMATIC BLOCK CODE.

Using elementary row operations, we can express any full rank generator matrix in the following form

$$G = \begin{bmatrix} \underline{g}_0 \\ \underline{g}_1 \\ \vdots \\ \underline{g}_{k-1} \end{bmatrix} = \left[\begin{array}{ccc|ccccc} p_{00} & p_{01} & \cdots & p_{0,n-k-1} & 1 & 0 & 0 & \cdots & 0 \\ p_{10} & p_{11} & \cdots & p_{1,n-k-1} & 0 & 1 & 0 & \cdots & 0 \\ \vdots & & & \vdots & & & & & \\ p_{k-1,0} & \cdots & & p_{k-1,n-k-1} & 0 & 0 & 0 & \cdots & 1 \end{array}\right]$$

P (arity) matrix — $k \times k$ identity matrix

$$= \left[\begin{array}{c|c} P & I_k \end{array}\right] \leftarrow \text{Canonical Form}$$

The first $(n-k)$ bits are called parity check bits.

Example : A single error detecting code

$$G = \left[\begin{array}{c|cccc} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{array}\right]$$

systematic

parity check equation | message

$$[1\ 0\ 1\ 0]\, G = [0 \mid \underbrace{1\ 0\ 1\ 0}_{\text{original message}}]$$

Example (7,4) Hamming code

$$G = \left[\begin{array}{ccc|cccc} 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{array}\right]$$

$$[1\ 0\ 1\ 0]\, G = [1\ 0\ 0 \mid \underbrace{1\ 0\ 1\ 0}_{\text{original message}}]$$

The representation of G is not unique. From linear algebra, we know that elementary row operations preserve the row space of the matrix.

Elementary row operations

1. Interchange any two rows
2. Multiply any row by a non-zero scalar ← not interesting in GF(2)
3. Add a multiple of any row to another.

Example $G = \left[\begin{array}{c|ccc} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{array}\right]$ (4,3) parity check code

Add row 1 to row 2

$$G' = \left[\begin{array}{c|ccc} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{array}\right]$$

Claim : G' generates the same set of codewords as G

Check :

Message	Codewords generated by G	Codewords generated by G'
000	0000	0000
001	1001	1001
010	1010	0110
011	0011	1111
100	1100	1100
101	0101	0101
110	0110	1010
111	1111	0011

So the performance of code generated by G is same as that G', since it is the same set of codewords. We like the systematic representation

① simpler encoding (fewer multiply/add)

② conceptually clear (parity + message)

To understand decoding, we need the important concept of distance. The following concepts are useful in analyzing all block codes (not just linear block codes).

Definition (HAMMING WEIGHT OF CODEWORD): The Hamming weight of a vector is defined as the number of non-zero coordinates of the vector. Weight of v is denoted by $w(v)$.

Example $w([1\ 0\ 1\ 1]) = 3$

$w([1\ 0\ 0\ 0]) = 1$

Hamming weight is analogous to the concept of distance in Euclidean spaces. Similarly, we need a concept of distance between two points in the space.

Definition (Hamming Distance): The Hamming distance between two vectors $\underline{v}$ and $\underline{u}$ is the number of coordinates in which the two codewords differ.

$$d(v,w) = \left| \{ i : v_i \neq w_i \ , \ i = 0, .., n-1 \} \right|$$

Hamming distance will be used to quantify number of errors, distance between codewords etc.

Hamming distance is a true distance.

It satisfies triangle inequality

$$d(\underline{v},\underline{w}) + d(\underline{w},\underline{u}) \geq d(\underline{v},\underline{u}).$$

When $\underline{v}$ and $\underline{u}$ belong to $\mathbb{V} = \mathbb{F}^n$ $(\mathbb{F} = GF(2))$, then

$$d(\underline{v},\underline{u}) = w(\underline{v}+\underline{u})$$

Example : $v = [1 \quad 0 \quad 1 \quad 0]$

$u = [1 \quad 0 \quad 0 \quad 0]$

$d(v,u) = 1$

$v + u = [0 \quad 0 \quad 1 \quad 0] \Rightarrow w(v+u) = 1$

Given a block code C, we can compute the Hamming distance between any two codewords. The smallest of such distances is called the minimum distance of the code C.

Definition (Minimum Distance) : The minimum distance d_{min} of a code C is defined as

$$d_{min} = \min\{ d(v,u) : v,u \in C, v \neq u \}$$

Note that the concept of weight of a vector, distance between vectors and minimum distance holds for all codes, not just linear codes.

To find the minimum distance of an arbitrary code C requires us to go through all combinations of codewords. If there are 2^k codewords, you need to go through $\binom{2^k}{2}$ codeword combinations.

Now G has more columns than rows ($k \leq n$), which implies it has a non-trivial dual space. We also know that the dual space has the dimension $(n-k)$, so there exists a matrix H such that the row space of H is orthogonal to the row space of G.

Take any codevector v, we know that v belongs to the row space of G. Since every vector in row space of H is orthogonal to every vector in the row space of G,

$$v \cdot H^T = 0 \qquad H \text{ is } (n-k) \times n$$

$\Rightarrow$ Any n-vector v is a codeword in the code generated by G if and only if $v \cdot H^T = 0$

The matrix H is called <u>parity check Matrix</u>. In fact H generates a linear code, which is a $(n, n-k)$ linear block code. The code is called the dual code C_d of C.

Example (5,4) parity check code

$$G = \left[\begin{array}{c|cccc} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{array}\right]_{4\times 5}$$

$$H = \left[\begin{array}{c|cccc} 1 & 1 & 1 & 1 & 1 \end{array}\right]_{1\times 5}$$

The dual takes one input bit and hence generates two codewords.

Check. $G \cdot H^T = 0$

$$\left[\begin{array}{c|cccc} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{array}\right] \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

If G is systematic, then H can be written immediately as

$$H = \left[\begin{array}{c|c} I_{n-k} & P^T \end{array}\right]_{(n-k)\times n}$$

$$= \left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & p_{00} & p_{10} \cdots & p_{k-1,0} \\ 0 & 1 & 0 & 0 & p_{01} & \cdots & p_{k-1,1} \\ 0 & 0 & 1 \cdots & 0 & \vdots & & \vdots \\ 0 & 0 & 0 \cdots & 1 & p_{0,n-k-1} & & p_{k-1,n-k-1} \end{array}\right]$$

Check $g_1 \cdot h_2^T = 0$

$$[p_{00} \; p_{01} \cdots p_{0,n-k-1} \; | \; 1 \; 0 \; 0 \; 0] \cdot [0 \; 1 \; 0 \; 0 \; | \; p_{01} \cdots p_{k-1,1}]^T$$

($p_{01} \cdots p_{0,n-k-1}$: $n-k$; $1\;0\;0\;0$: k; $0\;1\;0\;0$: $n-k$; $p_{01} \cdots p_{k-1,1}$: k)

$$= p_{01} + p_{01} = 0$$

SUMMARY: For every (n, K) linear block code $(n \geq K)$, there exists a $K \times n$ matrix G whose row space contains C. Also, there exists an $(n-K) \times n$ matrix H such that for all $v \in C$, $v \cdot H^T = 0$. If $G = [P \mid I_K]$, then $H = [I_{n-K} \; P^T]$.

Theorem: The minimum distance of a linear block code is equal to the minimum weight of its non-zero codewords.

Proof:

$$d_{min} = \min \{ d(U, V) : U, V \in C, U \neq V \}$$

If C is linear then $U+V \in C$

and we know that $d(U,V) = w(U+V)$ for GF(2) codes

$$d_{min} = \min \{ w(\underbrace{U+V}_{\in C}) : U, V \in C, U \neq V \}$$

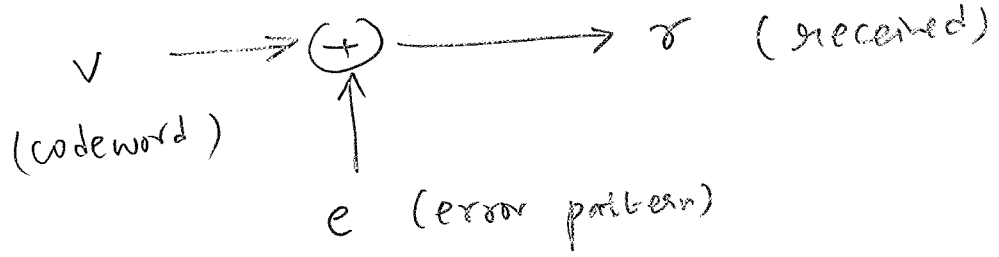
$$\Rightarrow d_{min} = \min \{ w(X) : X \in C, X \neq 0 \}$$

$$= \text{minimum weight codeword, } w_{min}$$

$\times$

Next we will see how d_{min} can be used to characterize the error-detecting and error-correcting performance of block codes (in general).

Error detection

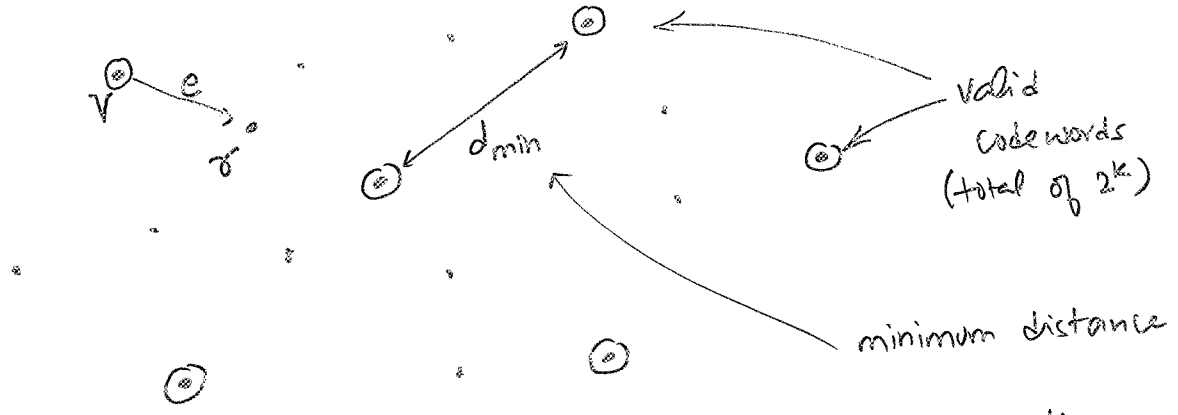


$$r = v + e$$

When l bits of v are flipped, then we know that the weight of the error vector is l, which implies

$$d(r, v) = l$$

Consider the following picture of the code C in n-dim space



When v was transmitted and r was received, the receiver can immediately tell if there was an error or not if the received vector is <u>not</u> a valid codeword. If the code C has a minimum distance of d_{min}, what is the maximum # of bit flips that can occur before you may not be able to tell if an error happened or not?

Answer is $d_{min}-1$.

If $l = w(e) < d_{min}$, no codeword can be converted into another codeword by e. So if the # of errors $\leq d_{min}-1$, then we can always detect errors.

Remember that errors cannot be detected when a codeword is converted into another codeword by e.

Q: Can we detect more than $d_{min}-1$ errors?

A. Yes, sometimes.

There may be codewords such that when you add e with $w(e) \geq d_{min}$, you may not get another valid codeword. In those cases, we can detect more than $d_{min}-1$ errors. But $d_{min}-1$ is the maximum number of errors which can be detected for any transmitted codeword.

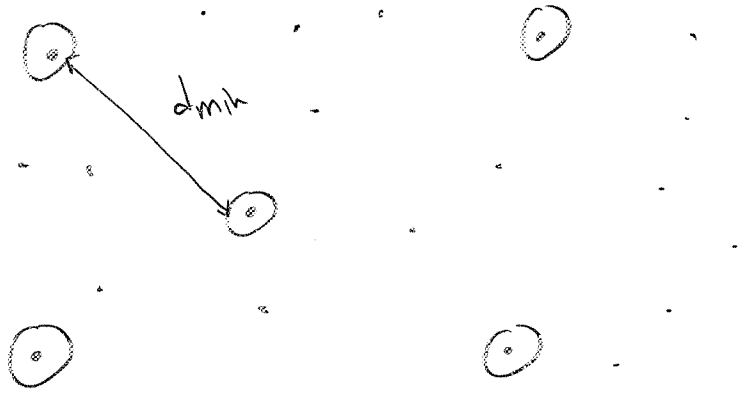
In fact, we can count the # of error patterns which can be detected.

- r can be any of the possible 2^n vectors
- all zero vector is not an error pattern, so total # of possible error vectors $= 2^n-1$
- an error pattern cannot be detected if it takes the current transmitted codeword and converts it into another codeword, which implies 2^k-1 error vectors.
- total # of detectable error vectors $= (2^n-1) - (2^k-1)$
  $= 2^n - 2^k$

If $R = \frac{1}{2} = \frac{k}{n} \Rightarrow n = 2k$

k	% undetectable $= \left(100 \times \frac{2^k-1}{2^n}\right)$
10	$\sim 0.1\%$
100	$\sim 8\times10^{-29}\%$

Next, we look at the **error correcting** performance of block codes. Again we use our geometrical picture to find out how many errors can be corrected for a block code with minimum distance d_{min}.



To correct errors, we need a notion of decoding.

Decoder is an operation which maps received vector r to a valid codeword.

$$\text{Decoder} : r \longmapsto \hat{c} \in C$$

No error is made in decoding if the decoded vector $\hat{c}$ is same as transmitted codeword c, i.e. $\hat{c} = c$.

Now d_{min} is either odd or even. Let t be a positive integer such that

$$2t+1 \leq d_{min} \leq 2t+2.$$

Claim: Next we will show that a block code (not necessarily linear) can correct <u>all</u> error patterns of weight t or less.

This implies that there exists a decoding rule which will map r to correct codewords if # of error bits $\leq t$.

Proof: Let v be the transmitted codeword.
r the received vector
w any other codeword ($w \neq v$)

We know that Hamming distance satisfies triangle inequality, so

$$d(v,r) + d(r,w) \geq d(v,w) \quad -(A)$$

Let $wt(e) = l \Rightarrow d(v,r) = l$

Also v, w are valid codewords, so

$$d(v,w) \geq d_{min} \geq 2t+1 \quad -(B)$$

From (A) & (B), we get

$$d(r,w) \geq 2t+1-l$$

If $l \leq t \Rightarrow d(r,w) > t$

$\Rightarrow$ r is closer to v than any other codeword if # of errors is less than t.

So use decoding rule: map r to closest codeword in Hamming distance sense.

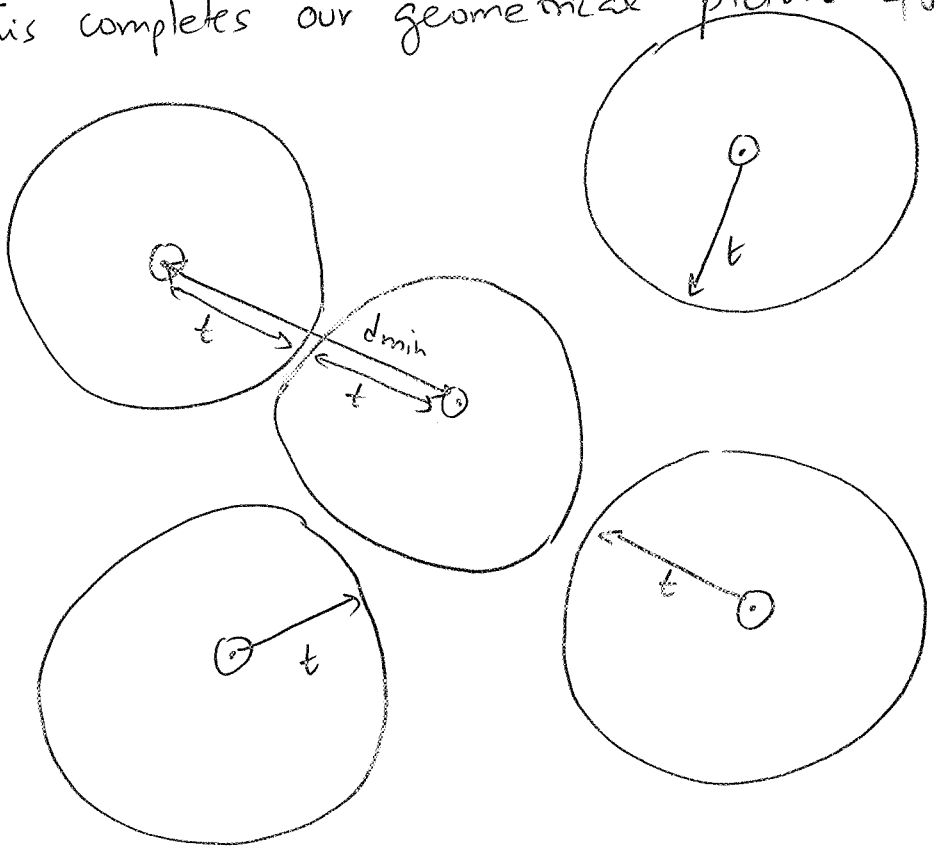
$\Rightarrow$ We can correct upto t errors

If d_{min} is odd, we can correct correct exactly t errors, $t = \frac{d_{min}-1}{2}$

If d_{min} is even, then we can correct all

$$t = \left\lfloor \frac{d_{min}-1}{2} \right\rfloor \text{ errors.}$$

This completes our geometrical picture for error correction



We can draw spheres of Hamming radius t around codewords, which implies that vectors in sphere around v are most likely to come from v. Decoding errors occur when $wt(e) > t$ and e puts you into another sphere.

There are "holes" in the above picture. why? How do we decode received vectors in the holes?

SUMMARY: A block code of minimum distance d_{min} allows

- detection of all error patterns of weight $\leq d_{min}-1$
- correction of all error patterns with weight $\leq \left\lfloor \frac{d_{min}-1}{2} \right\rfloor$

Decoding Linear Block Codes

Consider (n,k) linear code with generator G and parity check matrix H

$$r = v + e$$

To determine if there are any errors (at least those which can be detected), the decoder could use the fact that a valid codeword lies in the null space of the parity check matrix H.

$$\text{Let } s = r H^T$$
$$= (s_0\ s_1 \cdots s_{n-k-1})$$

s is called the syndrome of r.

NOTE $s = 0$ if and only if r is a codeword
$s \neq 0$ if and only if r is not a codeword

$$s = (v+e) \cdot H^T$$
$$= v \cdot H^T + e \cdot H^T$$
$$= e \cdot H^T$$

Hence the syndrome only depends on the error vector, not the transmitted codeword.

Now if the parity check matrix is expressed in the systematic form

$$H = \left[I_{n-k} \;\vdots\; P^T \right]$$

$$= \left[\begin{array}{cccc|cccc} 1 & 0 & \cdots & 0 & p_{00} & p_{10} & \cdots & p_{k-1,0} \\ 0 & 1 & \cdots & 0 & p_{01} & p_{11} & \cdots & p_{k-1,1} \\ \vdots & & & & \vdots & & & \\ 0 & 0 & \cdots & 1 & p_{0,n-k-1} & & & p_{k-1,n-k-1} \end{array}\right]_{(n-k)\times n}$$

$$s = e \cdot H^T \quad (1 \times (n-k) \text{ vector})$$

$$s_0 = e_0 + e_{n-k} p_{00} + e_{n-k+1} p_{10} + \cdots + e_{n-1} p_{k-1,0}$$

$$\Rightarrow \; s_1 = e_1 + e_{n-k} p_{01} + \cdots + e_{n-1} p_{k-1,1}$$

$$\vdots$$

$$s_{n-k-1} = e_{n-k-1} + e_{n-k} p_{0,n-k-1} + \cdots + e_{n-1} p_{k-1,n-k-1}$$

There are $(n-k)$ equations in n variables, so it has 2^k possible solutions. We will pick the solution with minimum weight, since it is the most likely error.

Then the most likely transmitted codeword is the one such that

$$r = c' + e'$$

such that $wt(e')$ has the smallest possible weight.

Standard Array Decoding : For an (n, k) linear code C, construct the standard array as follows:

(1) Collect all the codewords & write them in a row starting from all-zero codeword. Now consider the set $L = \mathbb{F}^n \setminus C$ (the set of all n-vectors minus the codewords)

(2) Select from L, the vector with the lowest weight. Write this vector below the all zero codeword. Add this pattern to each of the codeword, writing each sum under the respective codeword and removing the sum from the set of remaining n-tuples

(3) Repeat step (2) until all vectors in L are used up.

To decode : (a) Look up for r in the table

(b) The codeword at the top of the column in which r is located is the most likely codeword c associated with r.

Coset leader

$v_1 = 0$	v_2	v_3	. . .	v_{2^k} ← codewords
e_2	$e_2 + v_2$	$e_2 + v_3$	. . .	$e_2 + v_{2^k}$
e_3	$e_3 + v_2$	$e_3 + v_3$	. . .	$e_3 + v_{2^k}$] coset
⋮				
$e_{2^{n-k}}$	$e_{2^{n-k}} + v_2$	$e_{2^{n-k}} + v_3$	. .	$e_{2^{n-k}} + v_{2^k}$

Example : (6, 3) linear code

$$G = \left[\begin{array}{cccccc} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{array}\right] \leftarrow \text{systematic}$$

parity bits | message bits

$2^3 = 8$ codewords

If this was the received vector, then the ML codeword is 110001 & error is 000100

000000	011100	101010	110001	110110	101101	011011	000111
100000	111100	001010	010001	010110	001101	111011	100111
010000	001100	111010	100001	100110	111101	001011	010111
001000	010100	100010	111001	111110	100101	010011	001111
000100	011000	101110	110101	110010	101001	011111	000011
000010	011110	101000	110011	110100	101111	011001	000101
000001	011101	101011	110000	110111	101100	011010	000110
100100	111000	001110	010101	010010	001001	111111	100011
⋮							

Properties of Standard Array

(1) Sum of any two vectors in a row is a codeword in C

$$(e_2 + v_1) + (e_2 + v_2) = \underbrace{(e_2 + e_2)}_{=0} + \underbrace{v_1 + v_2}_{=\text{codeword}}$$

(2) No two entries in the same row are identical

Note all codewords of C are distinct — (*)

Now assume $e_i + v_i = e_i + v_j \quad i \neq j$

$\Rightarrow v_i = v_j$ impossible due to (*)

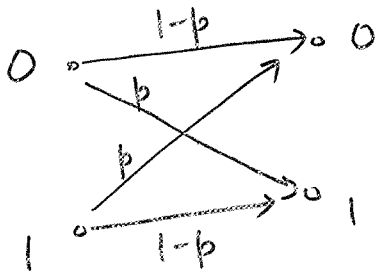
(3) There are $\frac{2^n}{2^k} = 2^{n-k}$ rows in the table

(including the row with zero codevector)

Each row has a unique leader

$\Rightarrow 2^{n-k}$ error patterns can be corrected by every (n,k) linear code.

(4) Standard array is a maximum likelihood decoder for a binary symmetric channel



On a BSC channel, 0 or 1 can be flipped with equal probability

Probability of one error = p and errors are independent from bit to bit.

For each r, we can compute the probability that v_i was the correct codeword as follows.

- $p(r|v_i) = p^{d(r,v_i)}(1-p)^{n-d(r,v_i)}$

In this case, more errors are less likely than less errors for $p < \frac{1}{2}$. So we select the most likely codeword (w/o prior, this minimizes probability of incorrect decoding)

$$v_i^* = \arg\max_{i=1,\dots,2^k} p(r|v_i)$$

Maximum Likelihood Estimate

$$= \arg\min_{i=1,\dots,2^k} d(r,v_i)$$

By construction of standard array, the coset leaders are minimal weight. Thus the decoded errors are minimal weight. Hence $d(r,v_i)$ is minimized.

(5) All 2^k vectors of a coset (row) have the same syndrome

Consider the coset leader is e_i

Any vector in the row is $e_i + v_j$

$$(e_i + v_j)\cdot H^T = e_i H^T + v_j H^T = e_i H^T$$

$\Rightarrow$ All vectors in a row have the same syndrome

(6) The syndrome of different cosets are different

Proof: e_i, e_j are the coset leaders of i & j rows, $j<i$

Assume $e_i H^T = e_j H^T$

$\Rightarrow (e_i + e_j) H^T = 0$

$\Rightarrow (e_i + e_j)$ is a codevector

Let $(e_i + e_j) = v_k$

$\Rightarrow e_i = e_j + v_k$

$\Rightarrow e_i$ is in the j^{th} row

Contradiction, since e_i is the coset leader and hence should not have appeared before in the table.

This leads to a simple decoding compared to syndrome table decoding, which requires us to store all possible 2^n vectors. [next page]

NOTE: There are 2^{n-k} coset leaders = # of possible error vectors which can be corrected

---

Since syndrome for each row is unique (prop (5)) we can use syndromes and their association with coset leaders as a decoding method.

Syndrome Decoding

(1) Take the first column of the standard array and compute syndromes for each vector. [offline step]

(2) Compute $r \cdot H^T$ $(n-k)$ dim vector $\Rightarrow$ 1 of the possible syndromes

(3) Find the syndrome it matches and use the corresponding error vector e to obtain $\boxed{v^* = r+e}$

Error Analysis of Linear Codes over BSC

We will look at standard array as our guide. Look at error detection first.

$P_W(E)$ = Prob(codeword error)

= Prob that channel causes the transmitted codeword to be received in error

$= 1 - (1-p)^n$

$= \sum_{i=1}^{n-1} \binom{n}{i} p^i (1-p)^{n-i}$

$P_d(E)$ = Prob(detected word error)

$P_U(E)$ = Prob(undetected word error)

Then $P_W(E) = P_d(E) + P_U(E)$

Look at $P_U(E)$. Undetected word errors occur iff the error pattern induced by the channel is a non-zero codeword.

First we derive a simple upper bound using d_{min}.

Now "undetected word error" $\subseteq$ "error pattern has wt $\geq d_{min}$"

$\Rightarrow P_U(E) \leq P(\text{error pattern of wt} \geq d_{min})$

$= \sum_{i=d_{min}}^{n} P(\text{error has wt} = i)$

$= \sum_{i=d_{min}}^{n} \binom{n}{i} p^i (1-p)^{n-i}$

The upper bound only depends only on d_{min}. The bound can be a little bit conservative, but it shows the importance of d_{min} as a design parameter. If one maximizes d_{min}, the upper bound is smaller and usually leads to smaller $P_b(E)$.

Next we compute exact $P_b(E)$.

$$P_b(E) = P(\text{error pattern is a non-zero codeword})$$

$$= \sum_{\substack{v_i \in C \\ v_i \neq 0}} \text{Prob}(\text{error pattern is } v_i)$$

$$= \sum_{\substack{v_i \in C \\ v_i \neq 0}} p^{wt(v_i)} (1-p)^{n-wt(v_i)}$$

collect terms $\rightarrow$

$$= \boxed{\sum_{i=1}^{n} A_i \, p^i (1-p)^{n-i}} \leftarrow \text{Exact}$$

Now for a linear code C with minimum distance d_{min}

$$A_0 = 1$$

$$A_1 = A_2 = \cdots = A_{d_{min}-1} = 0$$

The collection $\{A_0, A_1, A_2, \ldots, A_n\}$ is called the weight distribution of the code. It is common practice to track these weights via generating polynomial approach:

$$A(x) = A_0 + A_1 x + A_2 x^2 + \dots + A_n x^n$$

weight distribution polynomial
or weight enumerator

For most codes, weight distribution is not known. But it is known for several block codes like BCH. (Sometimes the weight distribution of the dual code is easier to find).

Denote the weight distribution polynomial of the dual code by $B(x)$ of (n,k) code, then

Theorem (MacWilliams Identity)

$$B(x) = \frac{1}{2^k}(1+x)^n A\left(\frac{1-x}{1+x}\right)$$

Error Correction of Linear Codes over BSC

$P(E)$ = Prob that decoder outputs the wrong codewords (we will assume that the decoder will always output a codeword)

Event "Output is a wrong codeword" $\subseteq$ Event "error pattern has wt $> t$"

$$2t+1 \le d_{min} \le 2t+2$$

$$t = \left\lfloor \frac{d_{min}-1}{2} \right\rfloor$$

This allows us to get an upper bound

$$P(E) \leq P(\text{error pattern has wt} > t)$$

$$= \sum_{i=t+1}^{n} \binom{n}{i} p^i (1-p)^{n-i}$$

Thus the bound depends only on d_{min}.
If we need the exact answer for $P(E)$, we need more info about the codeword (typically more than what is known).

$$P(E) = \text{Prob}(\text{error pattern is not correctable})$$

$$= \text{Prob}(\text{error pattern is not a coset leader in the standard array})$$

(assumes that we are using the optimal decoder)

$$= 1 - P(\text{error pattern is a coset leader})$$

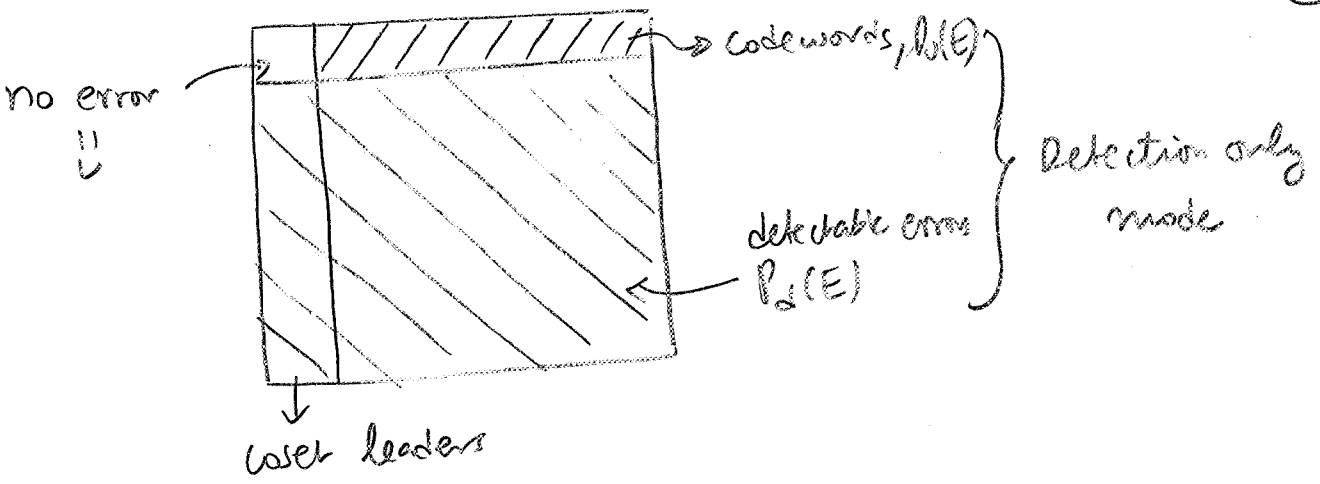
$$= 1 - \sum_{i=1}^{2^{n-k}} P(\text{error pattern is } e_i)$$

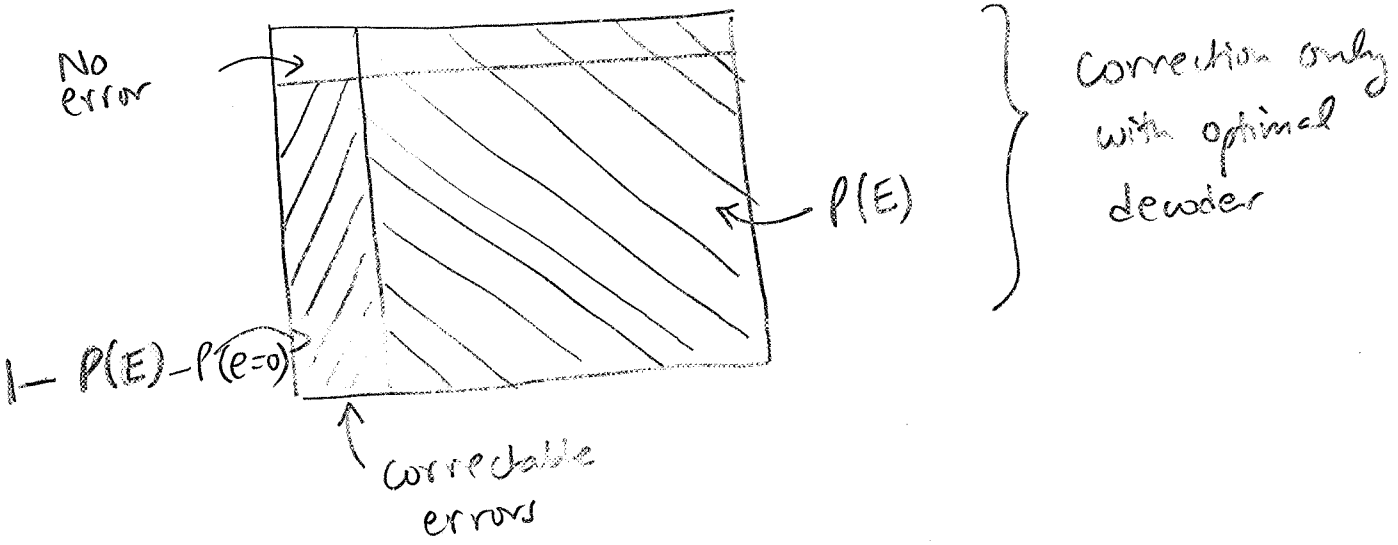
$$= 1 - \sum_{i=1}^{2^{n-k}} p^{wt(e_i)} (1-p)^{n-wt(e_i)}$$

$$= 1 - \sum_{i=0}^{n-1} L_i \, p^i (1-p)^{n-i}$$

where L_i = # coset leaders of weight i

We just used standard array for error analysis. Instead of looking at it as a set of received vectors, we can look at it as a set of error vectors.


No error
codewords, $P_u(E)$
Detection only mode
detectable errors
$P_d(E)$
coset leaders


No error
$P(E)$
Correction only with optimal decoder
$1 - P(E) - P(e=0)$
correctable errors

Bounds on Code Parameters

Now we will look at bounds on how many extra bits you need to achieve a certain level of error correction capability.

We need a notion of sphere, so we will define using Hamming distance.

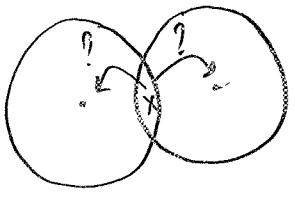
Definition (Volume of Hamming sphere): The volume of Hamming sphere of radius t is the number of vectors in $\mathbb{F}^n$ that are of distance $\leq t$ from center (origin).

Denote it by $V(n,t) = \binom{n}{0} + \binom{n}{1} + \binom{n}{2} + \cdots + \binom{n}{t}$

Theorem (Sphere packing bound or Hamming bound):
Let C be any (n, M) code with error correcting capability t, where M is the number of codewords.
Then

$$\boxed{M \leq \frac{2^n}{V(n,t)}}$$

Proof: For C to be t-error correcting, the Hamming spheres of radius t about each codeword must be non-overlapping



← cannot happen

These vectors account for $M V(n,t)$ vectors in $\mathbb{F}^n$. That may or may not be all of them

$$\Rightarrow \quad M V(n,t) \leq |\mathbb{F}^n| = 2^n$$

$$M \leq \frac{2^n}{V(n,t)}$$

Similarly using geometrical arguments, one can also bound M from below.

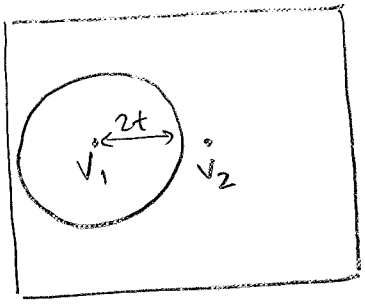
Theorem (Gilbert bound):

There is an (n,M) code such that it is t-error correcting and satisfies

$$M \geq \frac{2^n}{V(n,2t)}$$

Proof ("cookie-cutter" proof): What we will do is cut out spheres from $\mathbb{F}^n$ of a certain radius. And what we want to do is waste as little dough as possible. We want as many cookies out of this sheet of dough as possible. So let's start cutting out cookies.

Let $S_0 = \mathbb{F}^n$. From S_0, pick v_1 arbitrarily. We want t-error correcting so $d_{min} \geq 2t+1$.



To guarantee this, we need to pick v_2 outside the sphere of radius $2t$ about v_1. Hence, we set

$S_1 = S_0$ – sphere of radius $2t$ about v_1.

Pick $v_2 \in S_1$ arbitrarily. In order to ensure t-error correction, we should exclude the sphere of radius $2t$ about v_2. Let $S_2 = S_1$ – sphere of radius $2t$ abt v_2

Now pick $v_3 \in S_2$ arbitrarily.

Keep doing this as long as we can. How long can we do this depends on the choice of $v_1, v_2, \ldots$ If we pick well, then the sets S_i stay large for a long time.

Look at the general case

$$S_k = \mathbb{F}^n - \bigcup_{i=1}^{k} \text{sphere of radius } 2t \text{ about } v_i$$

We can pick v_{k+1} provided $S_k \neq \emptyset$

Note that $|S_k| = |\mathbb{F}^n - \bigcup_{i=1}^{k} \text{sphere about } v_i|$

$$|S_k| \geq 2^n - k\,V(n,2t)$$

Let M = smallest integer k for which

$$2^n - kV(n,2t) \leq 0$$

In other words

$$2^n - (M-1)V(n,2t) > 0$$

$$2^n - MV(n,2t) \leq 0$$

Hence we are guaranteed to be able to pick $v_M \in S_{M-1} \neq \phi$. But we are not guaranteed to be able to find any v_{M+1} because it is possible that $S_{M+1} = \phi$

Note $2^n - MV(n,2t) \leq 0$

$$\Leftrightarrow \boxed{M \geq \frac{2^n}{V(n,2t)}}$$

For linear codes

$$\frac{2^n}{V(n,2t)} \leq M \leq \frac{2^n}{V(n,t)}$$

Gilbert bound may or may not be true

Hamming bound always holds

But $M = 2^k$ for linear codes. So

$$2^k \leq \underbrace{\frac{2^n}{V(n,t)}} \iff V(n,t) \leq 2^{n-k} = 2^r$$

$$\Rightarrow \boxed{r \geq \log_2 V(n,t)}$$

Likewise Gilbert bound gives $\boxed{r \leq \log_2 V(n,2t)}$

So $\log_2 V(n,t) \leq r \leq \log_2 V(n,2t)$

$\log_2 V(n,t)$: always

$\log_2 V(n,2t)$: sometimes if the code is good

[Gilbert bound is more popularly known as Gilbert-Varshamov bound or sometimes Gilbert-Varshamov-Shannon bound]

Theorem (Singleton Bound): For a linear (n,k) code,

$$d_{min} \leq 1 + n - k$$

Proof: Look at systematic generator matrix for C

$$G = \begin{bmatrix} g_1 \\ \vdots \\ g_k \end{bmatrix}$$

$d_{min} \leq wt(g_i)$ for any i

Note that $wt(g_i) \leq \underbrace{1}_{\text{systematic part}} + \underbrace{n-k}_{\text{\# parity bits is at most } n-k}$

DEFINITION: (1) If the Hamming (sphere) packing bound is satisfied with equality, the code is said to be *perfect*.

(2) If the singleton bound is satisfied with equality, then the code is said to be *Maximum distance separable (MDS)*.

Examples
(1) Hamming codes are always perfect (pg. 101)
(2) Maximum distance separable binary codes (pg 238)

If $d_{min} = r+1$, then linear code has generator matrix in which parity checks are all 1's:

$$G = \left[\begin{array}{cccccc|ccccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & \cdots & 0 \end{array}\right] \Rightarrow d_{min} = 2$$

$\Rightarrow r = 1$ and $G = \left[\begin{array}{c|cccc} 1 & 1 & 0 & 0 \cdots & 0 \\ 1 & 0 & 1 & 0 \cdots & 0 \\ \vdots & & & & \\ 1 & 0 & \cdots & & 1 \end{array}\right]$

This is the $(n, n-1)$ single parity check code.

On the other hand, if there is only one row then

$$G = [1\ 1\ 1\ 1\ 1\ 1\ \cdots\ 1]$$

$d_{min} = n$, $r+1 = n-1+1 = n$

These are the only <u>binary</u> linear MDS codes