Week 2 : Session 1

LINEAR BLOCK CODES

Agenda:
- Describe Linear Block codes
  - What is Generator Matrix
  - What is Parity check Matrix
- What is Syndrome? And how this can be used for Error detection and Error Correction?

Any linear block code can be described by the parameters n and k where n is the codeword length, k is the information sequence length. So, an (n, k) linear block code can be defined by a $k \times n$ generator matrix, denoted by G.

$$G = \begin{bmatrix} g_0 \\ g_1 \\ \vdots \\ g_{k-1} \end{bmatrix} = \begin{bmatrix} g_{0,0} & g_{0,1} & g_{0,2} & \cdots & g_{0,n-1} \\ g_{1,0} & g_{1,1} & g_{1,2} & \cdots & g_{1,n-1} \\ \vdots & & & & \\ g_{k-1,0} & g_{k-1,1} & g_{k-1,2} & \cdots & g_{k-1,n-1} \end{bmatrix}_{k \times n}$$

Rank of this Generator matrix is k. This Generator Matrix is used to generate codewords, which is why it is called Generator Matrix.

The set of 2^k binary codewords is formed by taking the linear combinations of the rows of G.

For the binary information sequence $u = (u_0, u_1, \ldots, u_{k-1})$, the corresponding binary codeword sequence is given by

$$v = uG$$

$$= u_0 g_0 + u_1 g_1 + \cdots + u_{k-1} g_{k-1}$$

Modulo-2 addition

PROPERTIES

- The sum of any two codewords in a linear code is also a codeword. (ie) if v_1 and v_2 are codewords, then $v_1 + v_2$ is also a codeword.
- The all zero vector $0 = (0, 0, \ldots, 0)$ is a codeword in every linear code
- An (n, k) linear block code is a k-dimensional subspace of the vector space V_n of all binary n-tuples.

Example 1: Let $k = 3$ and $n = 6$.

Below table gives a (n, k) linear block code.

(ii) $(6, 3)$ linear block code.

$n = 6$

$k = 3$

Message (u_0, u_1, u_2)	Codewords $(v_0, v_1, v_2, v_3, v_4, v_5)$
(0 0 0)	(0 0 0 0 0 0)
(1 0 0)	(0 1 1 1 0 0)
(0 1 0)	(1 0 1 0 1 0)
(1 1 0)	(1 1 0 1 1 0)
(0 0 1)	(1 1 0 0 0 1)
(1 0 1)	(1 0 1 1 0 1)
(0 1 1)	(0 1 1 0 1 1)
(1 1 1)	(0 0 0 1 1 1)

$k = 3$

$2^k = 2^3 = 8$

? using Encoding equations.

$v_0 = u_1 + u_2$

$v_1 = u_0 + u_2$

$v_2 = u_0 + u_1$

$v_3 = u_0$

$v_4 = u_1$

$v_5 = u_2$

WKT, $v = uG$

$$\Rightarrow [v_0\ v_1\ v_2\ v_3\ v_4\ v_5] = [u_0\ u_1\ u_2]\ [G]_{k \times n}$$

$$= [u_0\ u_1\ u_2] \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}_{3 \times 6}$$

Thus, the Generator Matrix for this code is

Identity Matrix

$$G = \begin{bmatrix} g_0 \\ g_1 \\ g_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}_{3 \times 6}$$

Now, let's see, how we can generate codewords using this Generator matrix.

Consider the information sequence $u = (1\ 0\ 1)$ (say for example)

the codeword for the message $u = (1\ 0\ 1)$ is

$$v = uG$$

$$= u_0 g_0 + u_1 g_1 + u_2 g_2$$

$$= 1\,(0\ 1\ 1\ 1\ 0\ 0) + 0\,(1\ 0\ 1\ 0\ 1\ 0) + 1\,(1\ 1\ 0\ 0\ 0\ 1)$$

$$= (0\,1\,1\,1\,0\,0) + (0\,0\,0\,0\,0\,0\,0) + (1\,1\,0\,0\,0\,1)$$

$$= (0\,1\,1\,1\,0\,0) + (1\,1\,0\,0\,0\,1)$$

$$v = (1\,0\,1\,1\,0\,1)$$

Modulo 2 addition (XOR)

A	B	$Y = A'B + AB' = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

This is the codeword corresponding to the information sequence $u = (1\,0\,1)$.

So, this is how we generate codewords from the given Generator Matrix.

Now, we can also write the Generator matrix in Systematic form.

An (n, k) linear block code is in systematic form, if its generator matrix is in the following form:

$$G = [P : I_k] = [I_k : P]$$

$I_k \rightarrow$ Identity Matrix

$$= \left[\begin{array}{cccc|cccc} p_{0,0} & p_{0,1} & \cdots & p_{0,n-k-1} & 1 & 0 & 0 & \cdots\ 0 \\ p_{1,0} & p_{1,1} & \cdots & p_{1,n-k-1} & 0 & 1 & 0 & \cdots\ 0 \\ p_{2,0} & p_{2,1} & \cdots & p_{2,n-k-1} & 0 & 0 & 1 & \cdots\ 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots & \vdots \\ p_{k-1,0} & p_{k-1,1} & \cdots & p_{k-1,n-k-1} & 0 & 0 & 0 & \cdots\ 1 \end{array}\right]$$

k rows

$n-k$ columns | k columns

$k \times n$

$= n-k+k$

$= n$ columns

Every codeword consists of two parts:

- A message part
- A parity check part.

For Systematic Linear Block code, the message part consists of the k unaltered message bits, and the parity check part consists of $n-k$ parity check bits.

The encoding equations for a systematic code are given by

$$v_j = u_0 p_{0,j} + u_1 p_{1,j} + \cdots + u_{k-1} p_{k-1,j}, \quad 0 \le j \le n-k-1$$

$\longrightarrow$ Parity check equations

$$v_{n-k+i} = u_i, \quad 0 \le i \le k-1$$

$\hookrightarrow$ Message bits.

Each parity bit v_j, $0 \le j \le n-k-1$ is a sum (modulo 2) of certain message bits.

Example 2 : Consider the (6,3) code given in Example 1.

Its generator matrix in systematic form is

$n = 6$
$k = 3$

$$G = \left[\begin{array}{ccc|ccc} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{array}\right] = \left[\begin{array}{ccc|ccc} p_{00} & p_{01} & p_{02} & 1 & 0 & 0 \\ p_{10} & p_{11} & p_{12} & 0 & 1 & 0 \\ p_{20} & p_{21} & p_{22} & 0 & 0 & 1 \end{array}\right]$$

Let $u = (u_0, u_1, u_2)$ be the message to be encoded.

Then the codeword is

$$v = (v_0, v_1, v_2, v_3, v_4, v_5)$$
$$= uG$$

$0 \le j \le n-k-1$
$\Rightarrow 0 \le j \le 2$
$\Rightarrow j = 0, 1, 2.$

$0 \le i \le k-1$
$\Rightarrow 0 \le i \le 2$
$\Rightarrow i = 0, 1, 2$

$$v_5 = u_2$$
$$v_4 = u_1$$
$$v_3 = u_0$$

$v_{n-k+i} = u_i, \quad 0 \le i \le 2$

$v_{3+i} = u_i, \quad 0 \le i \le 2$

$$v_2 = u_0 + u_1$$
$$v_1 = u_0 + u_2$$
$$v_0 = u_1 + u_2$$

$v_j = u_0 p_{0,j} + u_1 p_{1,j} + \dots + u_{k-1} p_{k-1,j}, \quad 0 \le j \le 2$

$v_j = u_0 p_{0,j} + u_1 p_{1,j} + u_2 p_{2,j}, \quad 0 \le j \le 2$

Similar to Generator matrix, we can also describe linear block codes by Parity check matrix.

Linear (n, k) block can also be specified by an $(n-k) \times n$ Parity check matrix H. $\leftarrow$ [Rank $(n-k)$ Matrix].

If $v = (v_0, v_1, \dots, v_{n-1})$ is a binary n-tuple, then v is a codeword if and only if

$$vH^T = (0, 0, \dots, 0)$$

where $H \rightarrow$ Parity check Matrix.

WKT, $v = uG$

and $vH^T = 0$.

$\Rightarrow uGH^T = 0$

u may not be zero, So, we get $GH^T = 0$.

This means, Parity check Matrix lies in the NULL space of the Generator Matrix and Vice-versa.

So, Given a Generator Matrix, we can find out the Parity check Matrix. (or) Given a Parity check Matrix, we can find out the Generator Matrix.

For a systematic code with Generator Matrix $G = [P : I_k]$, the parity check matrix can be written as

$$H = [I_{n-k} : P^T]$$

$$= \left[\begin{array}{cccc|cccc} 1 & 0 & 0 & \cdots & 0 & P_{0,0} & P_{1,0} & \cdots & P_{k-1,0} \\ 0 & 1 & 0 & \cdots & 0 & P_{0,1} & P_{1,1} & \cdots & P_{k-1,1} \\ 0 & 0 & 1 & \cdots & 0 & P_{0,2} & P_{1,2} & \cdots & P_{k-1,2} \\ \vdots & & & & & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & 1 & P_{0,n-k-1} & P_{1,n-k-1} & \cdots & P_{k-1,n-k-1} \end{array}\right]_{(n-k)\times n}$$

($n-k$ rows; $n-k$ columns; k columns; $= n-k+k = n$ columns)

Example 3 : Consider a (7,4) linear systematic code with Generator Matrix

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

$n = 7$, $k = 4$

$\Rightarrow n - k = 3$

$G = [P : I_k]$

$\downarrow$

$H = [I_{n-k} : P^T]$

(Recall $G = [P : I_k] = [I_k : P]$)

Then the corresponding Parity check matrix in Systematic form is

$$H = [I_{n-k} : P^T]$$

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

From the Parity check Matrix, we can get Parity check equations.

$$v_2 = u_1 + u_2 + u_3$$
$$v_1 = u_0 + u_1 + u_2$$
$$v_0 = u_0 + u_2 + u_3$$
$$v = (v_0, v_1, v_2, u_0, u_1, u_2, u_3)$$

Syndrome and Error detection

Now that we have described the Generator Matrix and Parity check Matrix, let us now see how we can use Linear Block code to detect or correct error.

So, we'll introduce the notion of Syndrome, and how a Syndrome can be used for error detection or error correction.

Let $v = (v_0, v_1, \ldots, v_{n-1})$ be a codeword from a binary (n, k) linear block code, with Generator Matrix G and Parity check Matrix H.

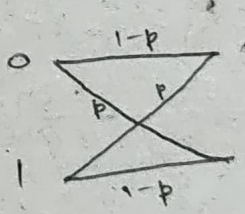
Assume v is transmitted over a Binary Symmetric channel (BSC) (for simplicity), then the binary received sequence is given by,

Transmitted codeword — Error

$$r = (r_0, r_1, \ldots, r_{n-1}) = v + e \pmod{2}$$
$$= (v_0, v_1, \ldots, v_{n-1}) + (e_0, e_1, \ldots, e_{n-1})$$
$$= (v_0 + e_0, v_1 + e_1, \ldots, v_{n-1} + e_{n-1})$$

where the binary vector, $e = (e_0, e_1, \ldots, e_{n-1})$ is the Error pattern.

BSC



- With probability 1-p, the bits are received correctly.
- With cross over probability of p, there is an error.

The "1's" in e represent transmission errors, (ie)

$$e_i = \begin{cases} 1 & \text{if } r_i \neq v_i \quad (\text{Error}) \\ 0 & \text{if } r_i = v_i \quad (\text{No Error}) \end{cases}$$

and $e_i = 1$ indicates that the i^{th} position in r has an error.

After receiving r, the decoder must determine if r contains errors (error detection), and locate the errors in r (error correction).

Error detection is achieved by computing the $(n-k)$ tuple.

$$s = (s_0, s_1, \ldots, s_{n-k-1}) = r H^T \quad (\text{Syndrome})$$

where, r → Received sequence
H → Parity check Matrix

r is a codeword if and only if Syndrome is zero.

(ie) $s = r H^T = 0$.

So, looking at the Syndrome, we can find out whether our received sequence has error or not. If there are errors, the received sequence would give a non-zero Syndrome. If the received sequence is received correctly, the Syndrome will be zero.

Another situation which happen, when the Error pattern is a valid codeword. When the error pattern gets added to the codeword, resulting received sequence is a valid codeword, and Syndrome will be zero in that case. So, those errors are known as undetected error. So, only when the Error pattern is also a valid codeword, the decoder would not be able to detect those errors. Why? Because when the error pattern is also valid codeword, and the received sequence is a valid codeword, so Sum of two codewords will be another codeword, and WKT $v H^T = 0$. So in those situation, the Syndrome will be zero.

When Syndrome is not zero, we know that r is not a valid codeword and the errors have occured. And when Syndrome is zero, there are two situations. Either (1) There is no error. (ii) the receive sequence is received correctly. That received sequence is a codeword. (2) The Error is a another valid codeword. That would result in undetected error.

If $s = 0$, r is a codeword and no errors are detected. If r is a codeword other than the actual transmitted codeword, then an undetected error occurs. This happens whenever the error pattern e is a non-zero codeword.

If $s \neq 0$, r is not a codeword and transmission errors have been detected.

Properties of Syndrome

(1) The Syndrome s computed from the received vector r actually depends only on the error pattern e, and not on the transmitted codeword V.

(ii) $$s = r \cdot H^T$$
$$= (V + e) H^T$$
$$= \cancel{V H^T}^{\,0} + e H^T$$

If V is a valid codeword, $V H^T = 0$.

$$\boxed{s = e \cdot H^T}$$

(2) The Syndrome equations for a Systematic code are given by

$$s_j = r_j + r_{n-k} p_{0,j} + r_{n-k+1} p_{1,j} + \dots + r_{n-1} p_{k-1,j} \quad \Big| \; s = r H^T$$

$$0 \leq j \leq n-k-1$$

And, we have the Parity check Equations,

$$V_j = u_0 p_{0,j} + u_1 p_{1,j} + \dots + u_{k-1} p_{k-1,j}$$

$$0 \leq j \leq n-k-1$$

Comparing the Syndrome equations with the Parity check

equations, we see that s_j is formed by adding the received parity bit r_j to the "parity bit" obtained by "encoding" the received information bits $r_{n-k}, r_{n-k+1}, \cdots, r_{n-1}$.

Example 4: Consider a (7,4) linear code with parity-check matrix, $H = \left[\begin{array}{ccc|cccc} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{array}\right]$.

Let $r = (0\ 1\ 0\ 0\ 0\ 0\ 1)$. The Syndrome of r is

$$s = (s_0, s_1, s_2) = r \cdot H^T$$

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

$$= \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \neq 0.$$

Note that, Syndrome is non-zero, which means there is an error in the received sequence. In the next section, we'll talk about how we can use the Syndrome for Error correction.

Week 2 : Session 2

DECODING OF LINEAR BLOCK CODES

In this section, we are going to talk about how we can use syndrome for Error Correction. So, Lets talk about decoding, which is essentially figuring out where the Errors have occurred and how we can find out the locations where error has happened and then correct them.

Recall, the Syndrome s computed from the received vector r actually depends only on the Error pattern e, and not on the transmitted codeword v.

$$s = r \cdot H^T = (v+e)H^T = e \cdot H^T \quad (\because v H^T = 0).$$

For Error pattern $e = (e_0, e_1, \ldots, e_{n-1})$, and H given by

$$H = [I_{n-k} : P^T] = \left[\begin{array}{cccc|ccc} 1 & 0 & 0 \cdots & 0 & p_{0,0} & p_{1,0} & \cdots p_{k-1,0} \\ 0 & 1 & 0 \cdots & 0 & p_{0,1} & p_{1,1} & p_{k-1,1} \\ 0 & 0 & 1 \cdots & 0 & p_{0,2} & p_{1,2} & p_{k-1,2} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 \cdots & 1 & p_{0,n-k-1} & p_{1,n-k-1} & p_{k-1,n-k-1} \end{array}\right]$$

the Syndrome equations can be rewritten as

$$\boxed{s_j = e_j + e_{n-k} p_{0,j} + e_{n-k+1} p_{1,j} + \cdots + e_{n-1} p_{k-1,j}, \quad 0 \le j < n-k.}$$

In this equations, we have $n-k$ equations, and we have n unknowns $(e_0, e_1, \ldots, e_{n-1})$. (ie) There are more unknowns than Equations. This will lead to a situation where there is more than one solution.

The decoder must solve these equations for the estimated error pattern, $\hat{e}$. The estimated codeword is

$$\hat{v} = r + \hat{e}.$$

(ii) Decoder has received the sequence r and it is trying to figure out the Error pattern. If it is able to correctly

figure out the error pattern, then it is going to estimate the codeword like $\hat{v} = r + \hat{e}$. Now if it is successful in estimating this error pattern, the decoded codeword $\hat{v}$ would be same as v, otherwise an error will happen.

As we have seen, there are n unknowns and n-k equations. And remember, these n unknowns that we have (e_i's) can take two values. They can be either 0, which corresponds to the case when there is no error, Or they can be 1 which corresponds to the case when there is an error. So, each of these e_i's can be either 0 or 1.

Since we have n unknowns and n-k equations, total solution that we have is 2^k. Now remember, among those 2^k solutions, there is only one solution which is correct, because there is a one error pattern. Now among those 2^k solutions, how do we figure out the most likely error pattern?

Since our objective is to minimize the probability of error, the most probable error pattern should be chosen. Out of those 2^k error pattern which are solution of the syndrome equations, we want to pick up the most likely pattern.

Recall, for BSC, the ML decoder chose $\hat{v}$ as the codeword $\hat{v}$, which would minimize the hamming distance between the receive sequence and the codeword. (ie) $d(r, \hat{v})$ is minimized.

Example 1: Consider a (n, k) linear binary code with Parity check Matrix, $H = \left[\begin{array}{ccc|cccc} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{array}\right]_{(n-k) \times n}$ in the Systematic form. Suppose $v = (1001011)$ is transmitted and $r = (1001001)$ is received. Then the Syndrome of r is $s = (s_0, s_1, s_2) = r \cdot H^T = (1, 1, 1)$.

The decoder needs to figure out from the received sequence r, where the error has occured and correct it. Now, the first thing the decoder will do is, it will compute the syndrome. So, once decoder computes the syndrome, and the syndrome comes out to be non-zero, the decoder knows that there is an error.

In this case, $s = r \cdot H^T = (1\ 1\ 1) \neq 0$, which means the decoder knows that there is an error.

In this example, $n = 7$ (7 unknowns)

$n - k = 3$ (3 equations)

$\Rightarrow k = 4$.

No. of solutions, $2^k = 2^4 = 16$.

So, in this particular example, there are 16 solutions. Out of those 16 solutions, we need to figure out which is the most probable error pattern.

Now, Syndrome, $s = r \cdot H^T = e \cdot H^T$

Let $e = (e_0, e_1, e_2, e_3, e_4, e_5, e_6)$ be the error pattern we get the following 3 equations.

$$1 = e_0 + e_3 + e_5 + e_6$$
$$1 = e_1 + e_3 + e_4 + e_5$$
$$1 = e_2 + e_4 + e_5 + e_6$$

3 equations
7 unknowns
$2^4 = 16$ Solutions

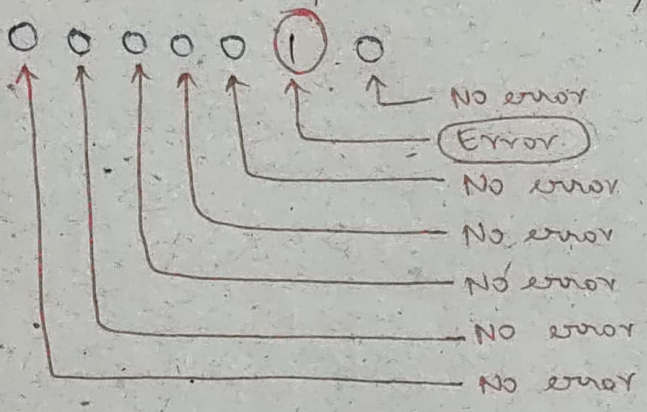
The 16 solutions are

(0 0 0 0 0 1 0)	(1 0 1 0 0 1 1)
(1 1 0 1 0 1 0)	(0 1 1 1 0 1 1)
(0 1 1 0 1 1 0)	(1 1 0 0 1 1 1)
(1 0 1 1 1 1 0)	(0 0 0 1 1 1 1)
(1 1 1 0 0 0 0)	(0 1 0 0 0 0 1)
(0 0 1 1 0 0 0)	(1 0 0 1 0 0 1)
(1 0 0 0 1 0 0)	(0 0 1 0 1 0 1)
(0 1 0 1 1 0 0)	(1 1 1 1 1 0 1)

There are all possible error patterns, which are solution to the set of 3 syndrome equations.

Remember, for a BSC, we should choose the codeword such that the Hamming distance between the codeword and the receive sequence is minimized. (ie) the Hamming weight or No. of 1's in the error pattern should be minimum.

So, by looking at the 16 solutions, the error pattern 0000010 has the least number of 1's. Lets pay attention to this error pattern. 0 0 0 0 0 1 0



It is indeed, error has occured in the same location in the Example 1.

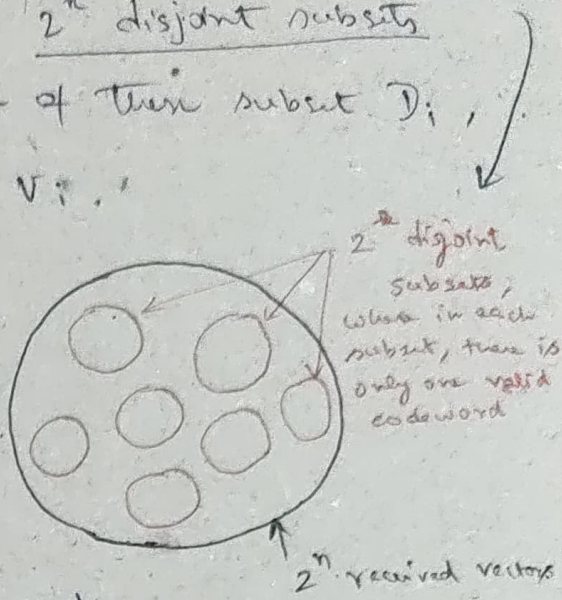
Note that, the true error pattern,

$$e = r + v$$
$$= (1001001) + (1001011)$$
$$= (0000010).$$

So, we have picked the correct pattern. Out of these 16 possible solutions, we pick the one which has least No. of 1's in error pattern, because that's the most likely error pattern according to the ML decoding rule that we derived for a BSC.

Now, lets talk about general methodology of decoding this Linear block codes. We are considering a BSC. So the received sequence is also binary. So, the received sequence can have total of 2^n possibilities.

Any decoding scheme used at the receiver, has a rule to partition the 2^n possible received vectors into 2^k disjoint subsets $D_1, D_2, \ldots, D_{2^k}$, such that each of these subset D_i, $1 \le i \le 2^k$ contains only one codeword V_i.



So, how do we perform decoding?

(i) Once we figured out, to which disjoint subset the received sequence belong to, with there is only one valid codeword associated with that disjoint subset. So, we decode that codeword as the estimated codeword.

This partition is based on linear structure of the code.

Let $V_1 = 0, V_2, \ldots, V_{2^k}$ be the 2^k codewords in an (n, k) linear block code.

Form an array of vectors from vector space, V_n as follows.

(i). Arrange the 2^k codewords as the top row of the array with $V_1 = 0$ (all zero codeword) as the first element.

So, first row first element is all zero codeword. And first row other elements are the other codewords.

(ii) Suppose, $j-1$ rows of the array have been formed, then choose a vector e_j from V_n which is not in the previous $j-1$ rows

(iii) Form the j^{th} row by adding e_j to each codeword V_i in the top row and placing $e_j + V_i$ under V_i.

(iv) Continue until all the vectors (2^n tuples) from V_n appear in the array.

This array is known as Standard array.

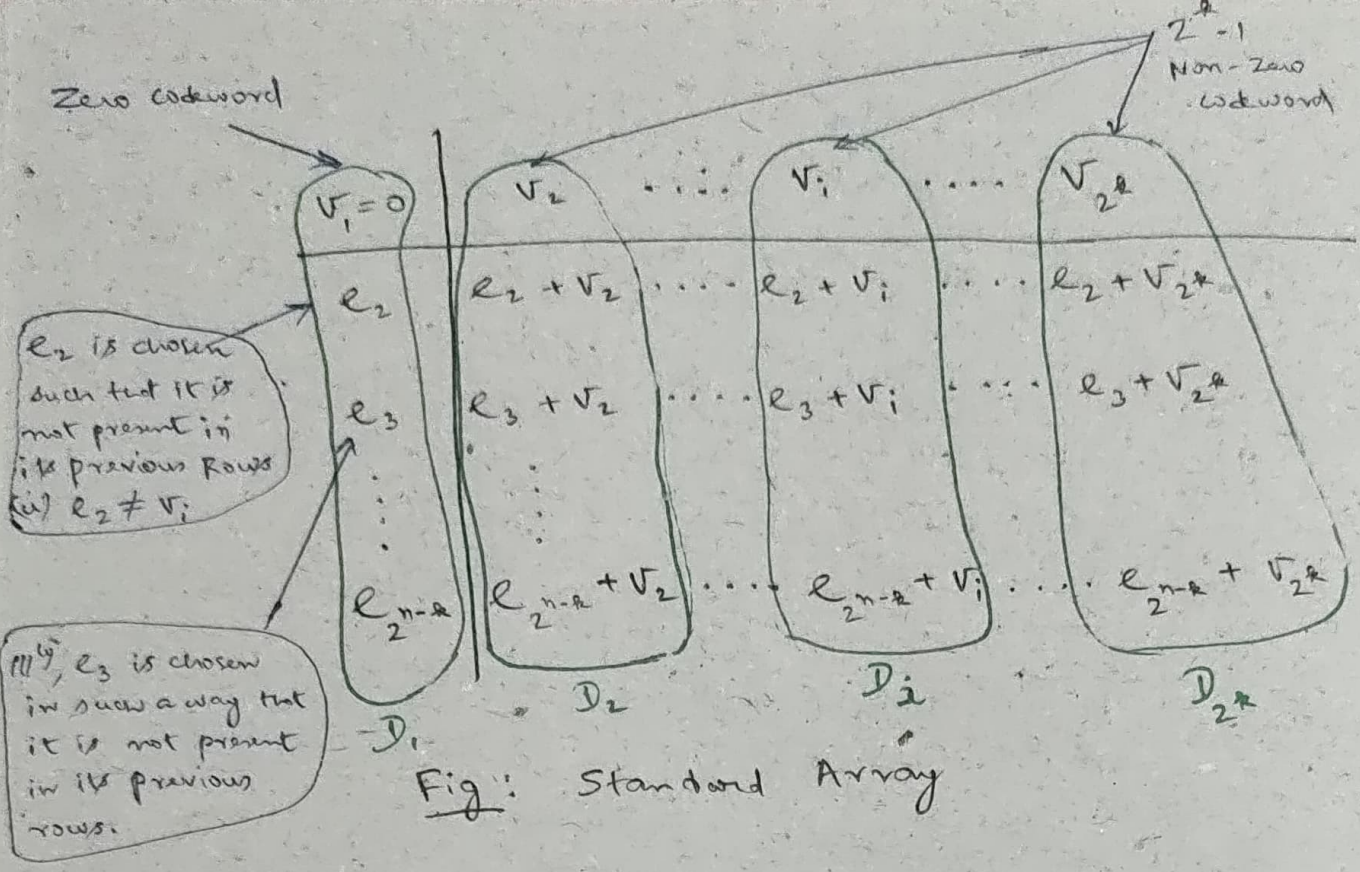


Fig: Standard Array

Notice: There are 2^{n-k} rows and 2^k columns. Thus, we have $2^{n-k} \times 2^k = 2^n \times 2^{-k} \times 2^k = 2^n$ elements.

Now, there 2^n possibilities have been partitioned into 2^k disjoint sets.

$D_1 \rightarrow$ Subset where V_1 is present

$D_2 \rightarrow$ Subset where V_2 is present

$D_i \rightarrow$ Subset where V_i is present

$D_{2^k} \rightarrow$ Subset where V_{2^k} is present

So, by exploiting the linear structure of the Block code, we have partitioned 2^n possible received sequence into 2^k disjoint subsets, where in each subset, there is one codeword.

Example: For a (6,3) linear code generated by the following Matrix,

$$G = \begin{bmatrix} g_0 \\ g_1 \\ g_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}_{3\times 6}$$

. The Standard array is $\longrightarrow$

Coset Leader v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8
(0 0 0 0 0 0)	(0 1 1 1 0 0)	(1 0 1 0 1 0)	(1 1 0 0 0 1)	(1 1 0 1 1 0)	(1 0 1 1 0 1)	(0 1 1 0 1 1)	(0 0 0 1 1 1)
(1 0 0 0 0 0)	(1 1 1 1 0 0)	(0 0 1 0 1 0)	(0 1 0 0 0 1)	(0 1 0 1 1 0)	(0 0 1 1 0 1)	(1 0 1 1 0 1)	(1 0 0 1 1 1)
(0 1 0 0 0 0)	(0 0 1 1 0 0)	(1 1 1 0 1 0)	(1 0 0 0 0 1)	(1 0 0 1 1 0)	(1 1 1 1 0 1)	(0 0 1 1 0 1)	(0 1 0 1 1 1)
(0 0 1 0 0 0)	(0 1 0 1 0 0)	(1 0 0 0 1 0)	(1 1 1 0 0 1)	(1 1 1 1 1 0)	(1 0 0 1 0 1)	(0 1 0 1 1 1)	(0 0 1 1 1 1)
(0 0 0 1 0 0)	(0 1 1 0 0 0)	(1 0 1 1 1 0)	(1 1 0 1 0 1)	(1 1 0 0 1 0)	(1 0 1 0 0 1)	(0 1 1 0 0 1)	(0 0 0 1 1 1)
(0 0 0 0 1 0)	(1 1 1 1 1 0)	(1 0 1 0 0 0)	(1 1 0 0 1 1)	(1 1 0 1 0 0)	(1 0 1 1 1 1)	(0 1 1 1 1 1)	(0 0 0 1 1 0)
(0 0 0 0 0 1)	(0 1 1 1 0 1)	(1 0 1 0 1 1)	(1 1 0 0 0 0)	(1 1 0 1 1 1)	(1 0 1 1 0 0)	(0 1 1 0 1 0)	(0 0 0 1 1 0)
(1 0 0 1 0 0)	(1 1 1 1 0 0)	(0 0 1 1 1 0)	(0 1 0 1 0 1)	(0 1 0 0 1 0)	(0 0 1 0 0 1)	(1 1 1 1 1 1)	(1 0 0 0 1 1)

This is the standard array for this particular (6, 3) linear block code. How to generate this standard Array ?

⊙ The first row first column is all zero code word. The other columns $(v_2, v_3, \ldots, v_8)$ of first row are computed

$$V = U G_{3\times 6}$$

$$= \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}_{8\times 3} \cdot \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}_{3\times 6}$$

$$= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}_{8\times 6} \begin{matrix} \leftarrow v_1 \\ \leftarrow v_2 \\ \leftarrow v_3 \\ \\ \\ \\ \\ \leftarrow v_8 \end{matrix}$$

⊙ Now, choose a vector e_j from V_n which has not happened before.

⊙ Adding e_j to each codeword v_i in the top row, and place $e_j + v_i$ under v_i

$$\begin{array}{rl} 0\ 1\ 1\ 1\ 0\ 0 & \leftarrow v_2 \\ +\ 1\ 0\ 0\ 0\ 0\ 0 & \leftarrow e_j \\ \hline 1\ 1\ 1\ 1\ 0\ 0 & \end{array}$$

⊙ continue until all possible 2^n tuples from V_n appear in this array.

Properties of Standard array.

⊙ Every vector in V_n appears exactly once in the standard array. This follows from the construction rule of the standard array. We'll use the method of contradiction to prove that. (i.e.) We'll assume that there are two elements in the standard array which are same, and then we'll show that it is not possible.

So, consider two rows i and j in the standard array.

Consider

$$\underbrace{e_i + V_1}_{\text{An element in Row } i} = \underbrace{e_j + V_2}_{\text{An element in Row } j}$$

$$\Rightarrow e_j = e_i + V_1 + V_2$$

$$\Rightarrow e_j = e_i + V_3$$

But, the construction rule says, we should use a pattern which has not appeared before. Whereas, in here, in Row j, we are using a pattern which has already appeared in Row i under codeword V_3.

Hence, by contradiction, there is no possibility of two patterns (same patterns) in the standard array.

⊙ No two vectors in the same row of a standard array are identical. because each row will have elements of the form e_i + codeword. and since since codeword is distinct, this is not possible.

⊙ Each row is called a COSET. There are exactly 2^{n-k} coset. The first element of each coset is called the coset Leader. (Any element in a coset can be used as it's coset leader. This does not change the elements of the coset, it changes the order of them).

⊙ All 2^k elements of a coset, have the same syndrome as their coset leader, since

$$s = (e_j + V_i)H^T = e_j H^T + V_i H^T = e_j H^T.$$

- The 2^k elements of a coset are the 2^k solutions to the syndrome equations.
- Each of the 2^{n-k} coset leaders has a different syndrome. Hence, there is one-to-one correspondence between a coset leader and a syndrome.
- The j^{th} column of a standard array

$$D_j = \{v_j, e_2 + v_j, e_3 + v_j, \ldots, e_{2^{n-k}} + v_j\}$$

contains exactly one codeword.
- If the received sequence r belongs to column D_j, then r is decoded into codeword v_j.
- If v_j is the transmitted codeword, and the error pattern is a coset leader e_i, then the received sequence $r = v_j + e_i$ is in column of D_j, which contains v_j (correct decoding).
- If the error pattern is not a coset leader, then r is not in column D_j (Incorrect decoding).
- Lets say the error pattern x caused by the channel is in l^{th} coset and under the code vector $v_i \neq 0$. Then $x = e_l + v_i$ and the received vector is

$$r = v_j + x = e_l + v_i + v_j = e_l + v_s$$

The received vector is in D_s, and decoded as v_s, which is not the transmitted codevector v_j.
- Therefore, decoding is correct if and only if the error pattern is a coset leader, and the 2^{n-k} coset leaders are all the correctable error patterns.

- To minimize the probability of error, the error patterns most likely to happen should be chosen as coset Leaders.
- For BSC, an error pattern of smaller weight is more probable than an error pattern of higher weight.
- Each coset leader should be chosen to be a vector of least weight from the available vectors.
- This way coset leader has minimum weight in its coset
- The decoding based on Standard Array, is the minimum distance decoding. (i.e.) ML decoding.

- Assume that the received vector r is found in the i^{th} column, and l^{th} coset of the standard array. Then r is decoded as code vector v_i.
- Since $r = e_l + v_i$, distance between r and v_i is

$$d(r, v_i) = w(r + v_i) = w(e_l + v_i + v_i) = w(e_l)$$

- Now consider the distance between r and any other code vector, say v_j,

$$d(r, v_j) = w(r + v_j) = w(e_l + v_i + v_j) = w(e_l + v_s)$$

where $v_s = v_i + v_j$

- Since e_l, and $e_l + v_s$ are in the same coset; and since $w(e_l) \leq w(e_l + v_s)$, it follows that

$$d(r, v_i) \leq d(r, v_j)$$

Hence if coset leader is chosen to have minimum weight in its coset, then decoding based on the standard array is the ML decoder.

Summary :

step 1 : Compute the Syndrome, $s = r \cdot H^T$

step 2 : Find the coset leader $\hat{e}$ whose syndrome is equal to s

step 3 : Decode r into the estimated codeword.

$$\hat{v} = r + \hat{e} .$$

⊙ Syndrome decoding can be implemented using a look-up table that consists of 2^{n-k} correctable error patterns (coset leaders) and their corresponding syndromes.

$s_1 = 0$	$\rightarrow$	$e_1 = 0$
s_2	$\rightarrow$	e_2
⋮		
$s_{2^{n-k}}$	$\rightarrow$	$e_{2^{n-k}}$

⊙ Syndrome decoding can also be used to perform a combination of error correction and error detection.

⊙ Coset leaders corresponding to the lowest weight error patterns are used for error correction. These are the most likely error patterns.

⊙ Syndromes corresponding to higher weight (less likely) error patterns are used to declare a detected error rather than for correction.

Example 2 : Consider a (6,3) linear systematic code generated by $G = \left[\begin{array}{ccc|ccc} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{array}\right] = [P \;\; I]$.

Its parity check Matrix is

$$H = \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{array}\right] = [I_3 \;\; P^T]$$

Encoding :

$$(u_0, u_1, u_2) \longleftrightarrow (v_0, v_1, v_2, u_0, u_1, u_2)$$

where

$$v_0 = u_1 + u_2$$

$$v_1 = u_0 + u_2$$

$$v_2 = u_0 + u_1$$

Syndrome look-up table.

Syndromes (s_0, s_1, s_2)	Correctable Error Patterns $(e_0, e_1, e_2, e_3, e_4, e_5)$
(0 0 0)	(0 0 0 0 0 0)
(1 0 0)	(1 0 0 0 0 0)
(0 1 0)	(0 1 0 0 0 0)
(0 0 1)	(0 0 1 0 0 0)
(0 1 1)	(0 0 0 1 0 0)
(1 0 1)	(0 0 0 0 1 0)
(1 1 0)	(0 0 0 0 0 1)
(1 1 1)	(1 0 0 1 0 0)

Coset Leader, v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8
(0 0 0 0 0 0 0)	(0 1 1 1 0 0 0)	(1 0 1 0 1 0 0)	(1 1 0 0 0 0 1)	(1 1 0 1 1 1 0)	(1 0 1 1 1 0 1)	(0 1 1 0 1 1 1)	(0 0 0 1 1 1 1)
(1 0 0 0 0 0 0)	(1 1 1 1 0 0 0)	(0 0 1 0 1 1 0)	(0 1 0 0 0 0 1)	(0 1 0 1 1 1 0)	(0 0 1 1 1 0 1)	(0 0 1 0 1 1 1)	(1 0 0 1 1 1 1)
(0 1 0 0 0 0 0)	(0 0 1 1 0 0 0)	(1 1 1 0 1 1 0)	(1 0 0 0 0 0 1)	(1 0 0 1 1 1 0)	(1 1 1 1 0 1 1)	(1 1 0 1 0 1 1)	(0 1 0 1 1 1 1)
(0 0 1 0 0 0 0)	(0 1 0 1 0 0 0)	(1 0 0 0 1 1 0)	(1 1 1 0 0 1 1)	(1 1 1 1 1 1 0)	(1 0 0 1 0 1 1)	(1 0 1 0 1 1 1)	(0 0 1 1 1 1 1)
(0 0 0 1 0 0 0)	(0 1 1 0 0 0 0)	(1 0 1 1 1 1 0)	(1 1 0 1 0 1 1)	(1 1 0 0 1 0 1)	(1 0 1 0 1 0 1)	(1 1 1 0 1 0 1)	(0 0 0 0 1 1 1)
(0 0 0 0 1 0 0)	(0 1 1 1 1 0 0)	(1 0 1 0 0 0 1)	(1 1 0 0 1 1 1)	(1 1 0 1 0 1 0)	(1 0 1 1 0 0 1)	(1 1 0 1 0 1 0)	(0 0 1 1 0 1 0)
(0 0 0 0 0 1 0)	(0 1 1 1 1 0 1)	(1 0 1 0 1 1 1)	(1 1 0 0 0 0 0)	(1 1 0 1 1 1 1)	(1 0 1 1 1 0 0)	(1 1 1 1 1 1 1)	(1 0 0 1 1 0 1)
(0 0 0 0 0 0 1)	(0 1 1 1 0 0 0)	(1 0 1 0 1 0 0)	(1 1 0 0 0 0 1)	(1 1 0 1 1 1 0)	(1 0 1 1 1 0 1)	(0 1 1 0 1 1 1)	(0 0 0 1 1 1 1)
(1 0 0 1 0 0 0)	(1 1 1 0 0 0 0)	(0 0 1 1 1 1 0)	(1 0 1 0 1 0 1)	(1 0 1 1 0 1 0)	(0 0 1 0 1 0 0)	(1 1 0 0 1 0 0)	(1 0 0 0 1 1 0)

Weight-2

⊙ Correctable Error Patterns : 7

Whenever, the Syndrome is pointing to these error patterns, we do Error correction.

⊙ Detectable Error Patterns : 8

In the last row, we see there are three patterns of weight-2. We don't know which one is the most likely coz all three are equally likely. So, when Syndrome is pointing to that row, we cannot correct the Error, but only detect. "Hey. I detected an Error, but I don't know which one of these errors have occured"

⊙ Undetected decoding Errors : 49

- Why? whenever the Syndrome is pointing to these cosets, we choose the coset Leader to be the Most likely Error pattern.

The Problem with this approach is, it is a quite complex procedure coz as 'n' increases, the size of the standard array also increases a lot.

In subsequent sections, we'll use the Structure of the Linear Block codes, to come up with simplified decoding algorithm.