

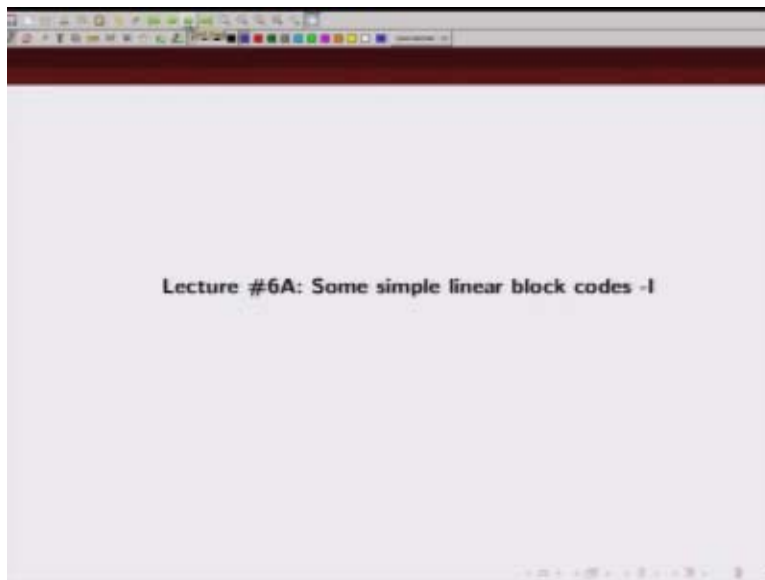
Indian Institute of Technology Kanpur
National Programme on Technology Enhanced Learning (NPTEL)
Course Title
Error Control Coding: An Introduction to Linear Block Codes

Lecture-6A
Some Simple Linear Block Codes-I

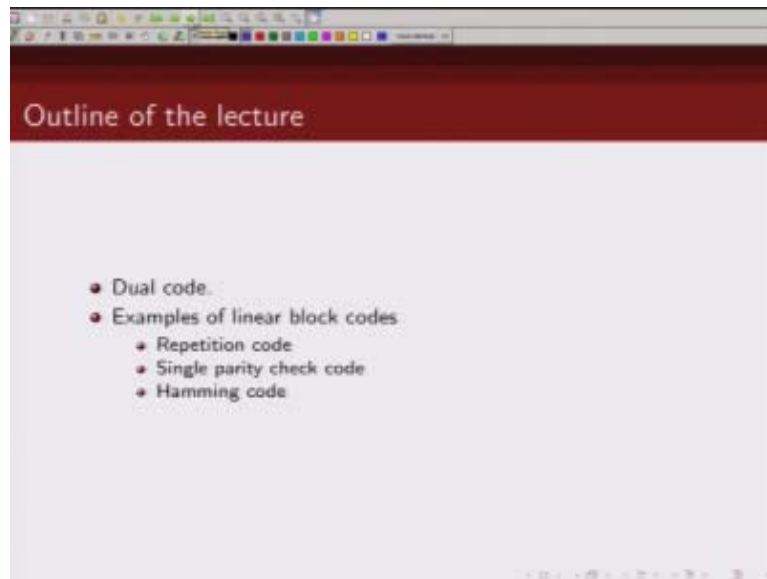
by
Prof. Adrish Banerjee
Department of Electrical Engineering, IIT Kanpur

Welcome to the course on error control coding, an introduction to linear block codes. So today we will discuss about some very simple block codes.

(Refer Slide Time: 00:25)

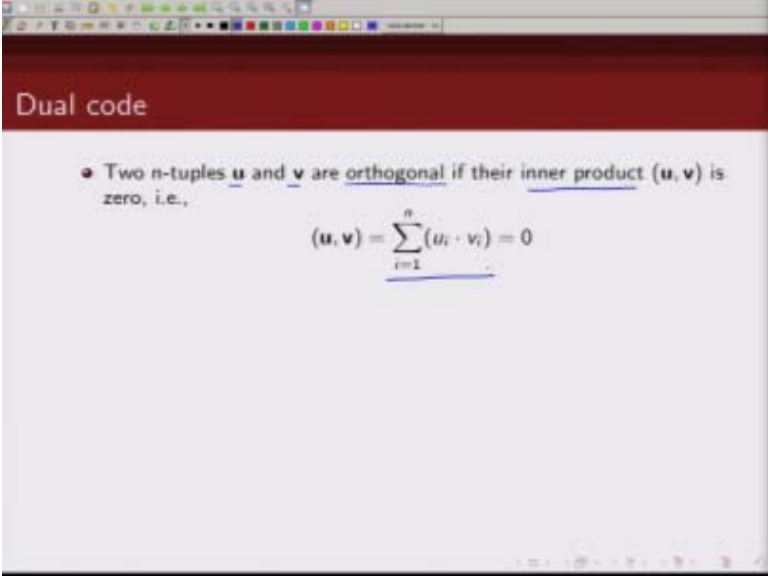


(Refer Slide Time: 00:27)



So this is the outline of today's talk. Before I discuss some examples of linear block code, I will first describe what do I mean by dual of a code. And then I will move on and describe some very simple linear block codes such as repetition code, single parity check code, hamming code.

(Refer Slide Time: 00:50)

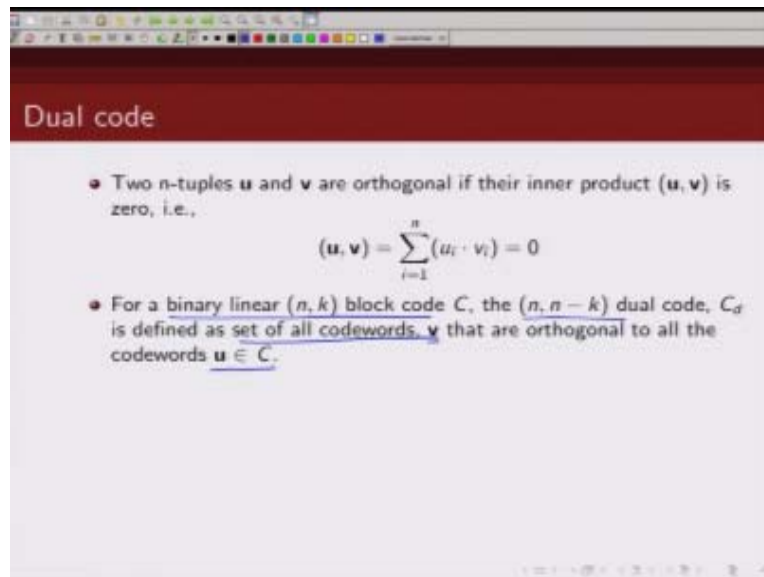


Dual code

- Two n-tuples u and v are orthogonal if their inner product (\mathbf{u}, \mathbf{v}) is zero, i.e.,
$$(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^n (u_i \cdot v_i) = 0$$

So before I discuss what is dual code, I would like to define what do I mean by two vectors \mathbf{u} and \mathbf{v} being orthogonal. So two vectors \mathbf{u} and \mathbf{v} are orthogonal if their inner product which is defined like this, so component wise dot product if that inner product is zero we call these n-tuples \mathbf{u} and \mathbf{v} as orthogonal.

(Refer Slide Time: 01:19)

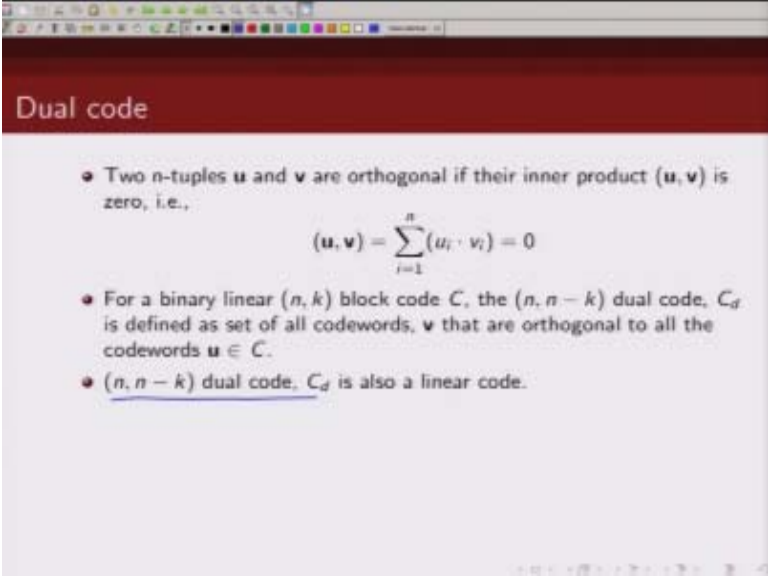


Dual code

- Two n -tuples \mathbf{u} and \mathbf{v} are orthogonal if their inner product (\mathbf{u}, \mathbf{v}) is zero, i.e.,
$$(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^n (u_i \cdot v_i) = 0$$
- For a binary linear (n, k) block code C , the $(n, n - k)$ dual code, C_d is defined as set of all codewords, \mathbf{v} that are orthogonal to all the codewords $\mathbf{u} \in C$.

So for a binary linear (n, k) code its dual has the parameter n and $n-k$ and it has the following properties. So a dual code has – is defined such that its set of code words \mathbf{v} are orthogonal to the set of code words of the original code C . So if \mathbf{v} is the code word which belongs to the dual of a code C then \mathbf{v} would be orthogonal to code words \mathbf{u} which belongs to the original code C .

(Refer Slide Time: 02:05)



Dual code

- Two n -tuples \mathbf{u} and \mathbf{v} are orthogonal if their inner product (\mathbf{u}, \mathbf{v}) is zero, i.e.,
$$(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^n (u_i \cdot v_i) = 0$$
- For a binary linear (n, k) block code C , the $(n, n - k)$ dual code, C_d is defined as set of all codewords, \mathbf{v} that are orthogonal to all the codewords $\mathbf{u} \in C$.
- $(n, n - k)$ dual code, C_d is also a linear code.

We can show that for a linear block code C the dual code is also a linear code.

(Refer Slide Time: 02:15)

Dual code

- Two n -tuples \mathbf{u} and \mathbf{v} are orthogonal if their inner product (\mathbf{u}, \mathbf{v}) is zero, i.e.,

$$(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^n (u_i \cdot v_i) = 0$$
- For a binary linear (n, k) block code C , the $(n, n - k)$ dual code, C_d is defined as set of all codewords, \mathbf{v} that are orthogonal to all the codewords $\mathbf{u} \in C$.
- $(n, n - k)$ dual code, C_d is also a linear code.
- Proof: Let $\mathbf{x}, \mathbf{y} \in C_d$, then $\mathbf{x} \cdot \mathbf{u} = \mathbf{y} \cdot \mathbf{u} = 0$ for every $\mathbf{u} \in C$. C_d

Handwritten annotations in blue: Arrows point from \mathbf{x} and \mathbf{y} to C_d , and from \mathbf{u} to C .

So let us take x and y , two code words which belong to this dual code of C which we are denoting by C_d somewhere – sometimes people use this notation also for the dual. Now if x and y belongs to the dual code then we know that any code words belonging to the dual code they are orthogonal to the code words in the original code C . So if u belongs to C and x belongs to dual code C_d then $x \cdot u$ will be 0, because the code word x is orthogonal to code word u .

Similarly code word y which belongs to the dual code and code word u which belongs to the original code u , since they are orthogonal their dot product will be 0. So we can write $x \cdot u = y \cdot u = 0$ this follows from the property that a code which belongs to the dual code is orthogonal to the code words in the original code C .

(Refer Slide Time: 03:40)

Dual code

- Two n -tuples \mathbf{u} and \mathbf{v} are orthogonal if their inner product (\mathbf{u}, \mathbf{v}) is zero, i.e.,
$$(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^n (u_i \cdot v_i) = 0$$
- For a binary linear (n, k) block code C , the $(n, n - k)$ dual code, C_d is defined as set of all codewords, \mathbf{v} that are orthogonal to all the codewords $\mathbf{u} \in C$.
- $(n, n - k)$ dual code, C_d is also a linear code.
- Proof: Let $\mathbf{x}, \mathbf{y} \in C_d$, then $\mathbf{x} \cdot \mathbf{u} = \mathbf{y} \cdot \mathbf{u} = 0$ for every $\mathbf{u} \in C$
- Thus,
$$(\lambda \mathbf{x} + \mu \mathbf{y}) \cdot \mathbf{u} = \lambda(\mathbf{x} \cdot \mathbf{u}) + \mu(\mathbf{y} \cdot \mathbf{u}) = 0$$
for every $\mathbf{u} \in C$

So then for some binary λ and μ we can write this $\lambda \mathbf{x} + \mu(\mathbf{y} \cdot \mathbf{u})$ as $\lambda(\mathbf{x} \cdot \mu) + \mu(\lambda \cdot \mathbf{u})$. Now what is $\mathbf{x} \cdot \mathbf{u}$? $\mathbf{x} \cdot \mathbf{u}$ is 0, because \mathbf{x} belongs to the dual code and \mathbf{u} belongs to the original code C . So they are orthogonal that is why $\mathbf{x} \cdot \mathbf{u}$ is 0. Similarly $\mathbf{y} \cdot \mathbf{u}$ is also 0, hence we can write $\lambda \mathbf{x} + \mu \mathbf{y} \cdot \mathbf{u} = 0$. So we have shown basically this is – this belongs to the dual code. So we have shown that if our original code linear block code is original block (n, k) code is linear the dual code is also linear.

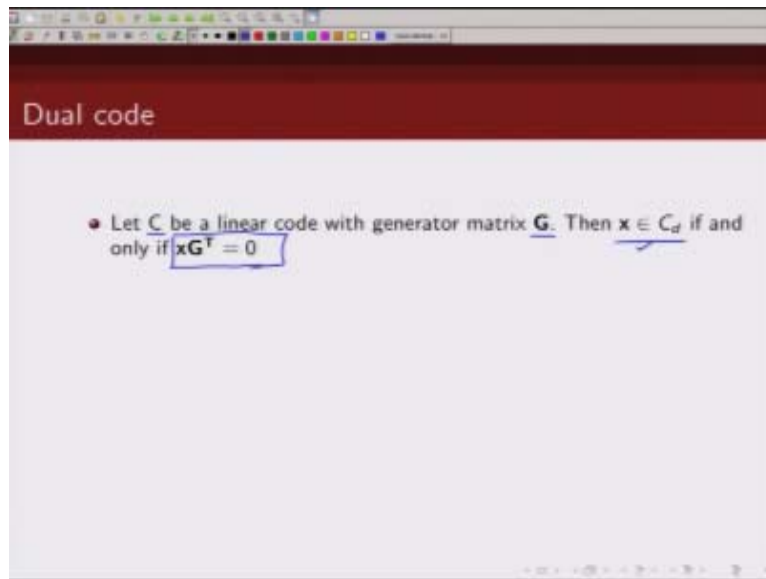
(Refer Slide Time: 04:46)

Dual code

- Two n -tuples \mathbf{u} and \mathbf{v} are orthogonal if their inner product (\mathbf{u}, \mathbf{v}) is zero, i.e.,
$$(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^n (u_i \cdot v_i) = 0$$
- For a binary linear (n, k) block code C , the $(n, n - k)$ dual code, C_d is defined as set of all codewords, \mathbf{v} that are orthogonal to all the codewords $\mathbf{u} \in C$.
- $(n, n - k)$ dual code, C_d is also a linear code.
- Proof: Let $\mathbf{x}, \mathbf{y} \in C_d$, then $\mathbf{x} \cdot \mathbf{u} = \mathbf{y} \cdot \mathbf{u} = 0$ for every $\mathbf{u} \in C$
- Thus,
$$(\lambda \mathbf{x} + \mu \mathbf{y}) \cdot \mathbf{u} = \lambda(\mathbf{x} \cdot \mathbf{u}) + \mu(\mathbf{y} \cdot \mathbf{u}) = 0$$
for every $\mathbf{u} \in C$ $\in C_d$ $\in C$
- This implies $\lambda \mathbf{x} + \mu \mathbf{y} \in C_d$

And why does this belong to the dual code that is because this is orthogonal to the code word which belongs to the – \mathbf{u} belongs to C and since this is orthogonal to a code word which belongs to C , so this must belong to dual code.

(Refer Slide Time: 05:08)



The next property that we are going to show is, if we have a linear block code we denote it by C , whose generator matrix is given by this capital G and if x belongs to the dual code of this original code C . So the claim that we are making is, if x belongs to the dual code then this relation holds and if this relation holds x belongs to the dual code. That is what we mean by if and only if.

So if x belongs to the dual code then xG^T should be 0 and if xG^T is 0 x should belong to the dual code. So we are going to prove that if x belongs to dual code, then this relation holds and further we will show if this condition holds then x will belong to the dual code.

(Refer Slide Time: 06:17)

Dual code

- Let C be a linear code with generator matrix G . Then $x \in C^\perp$ if and only if $xG^T = 0$
- Let G be given by
$$G = \begin{bmatrix} g_0 \\ g_1 \\ \vdots \\ g_{k-1} \end{bmatrix}$$
where $\{g_i\}$ is some basis of G .
- Also, $xG^T = (x \cdot g_0, \dots, x \cdot g_{k-1})$.

So let us write the generator matrix of a linear block code C as you know, this is a $k \times n$ matrix and this g_0, g_1, g_2, g_{k-1} are these k generators each of length up to n , these are length n . So this G is $k \times n$ matrix and any code word can be generated using these generators g_0, g_1, g_2, g_{k-1} . Now xG^T is basically given by inner product of x with g_0 , x with g_1 , x with g_2 up to x with g_{k-1} . Now what happens if x belongs to the dual code.

If x belongs to the dual code, then the set of code words which belongs to the dual code, they are orthogonal to the set of code words which belongs to the original code C .

(Refer Slide Time: 07:24)

Dual code

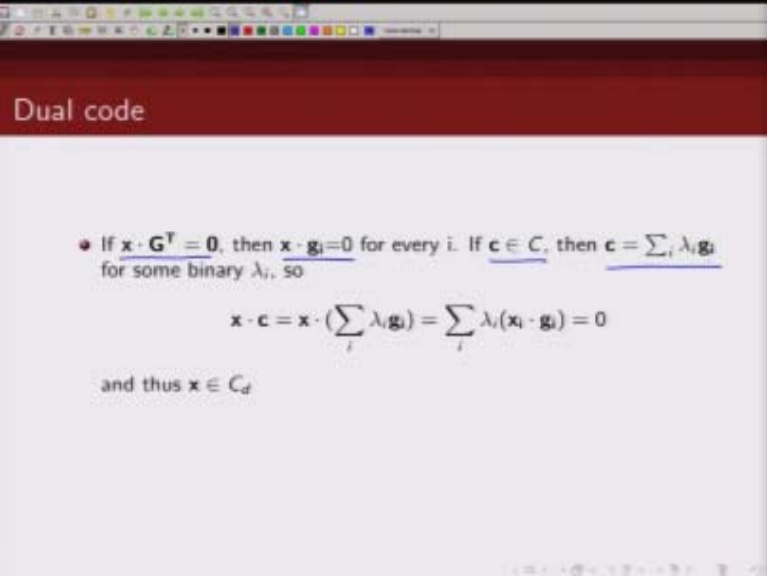
- Let C be a linear code with generator matrix G . Then $x \in C_d$ if and only if $xG^T = 0$
- Let G be given by

$$G = \begin{bmatrix} g_0 \\ g_1 \\ \vdots \\ g_{k-1} \end{bmatrix}$$
 where $\{g_i\}$ is some basis of G .
 $v = u_0g_0 + \dots + u_{k-1}g_{k-1}$
- Also, $xG^T = (x \cdot g_0, \dots, x \cdot g_{k-1})$.
- If $x \in C_d$, then $x \cdot g_i = 0$ for every i , so $xG^T = 0$.

Now we know that if x belongs to the dual code then x inner product of x with g_i should be 0, why? because the original code is generated using these generator sequence is g_1, g_2, g_{k-1} any linear combination of this g_0, g_1, g_2, g_{k-1} will give me my coded sequence, v is – you can write $g_0, g_0 + \dots + u_{k-1}, g_{k-1}$ right. So if x belongs to the dual code then inner product of x with g_i 's would be 0.

And hence xG^T will be 0. So what we have shown is, if x belongs to the dual code from the property that the code words in the dual code and code words in the original codes they are orthogonal to each other, from that property we get this condition that, that inner product of x with g_i 's will be 0 or in matrix form we can then write, because x , the inner product of x with g_i 's is nothing but xG^T . So then xG^T would be 0. Next we are going to show if xG^T is 0, then x must belong to the dual code.

(Refer Slide Time: 09:02)



Dual code

- If $\mathbf{x} \cdot \mathbf{G}^T = \mathbf{0}$, then $\mathbf{x} \cdot \mathbf{g}_i = 0$ for every i . If $\mathbf{c} \in C$, then $\mathbf{c} = \sum_i \lambda_i \mathbf{g}_i$ for some binary λ_i , so

$$\mathbf{x} \cdot \mathbf{c} = \mathbf{x} \cdot \left(\sum_i \lambda_i \mathbf{g}_i \right) = \sum_i \lambda_i (\mathbf{x} \cdot \mathbf{g}_i) = 0$$

and thus $\mathbf{x} \in C_d$

Now if \mathbf{xG}^T is 0, then this condition holds, that inner product of \mathbf{x} with \mathbf{g}_i is 0 for $i=0, 1, \dots, k-1$. Now what is a code word? A code word is obtained by linear combinations of these generators $\mathbf{g}_0, \mathbf{g}_1, \mathbf{g}_2, \dots, \mathbf{g}_{k-1}$. So if \mathbf{c} belongs to the linear block code C , then \mathbf{c} is essentially generated from my linear combinations of these generator sequences, where these λ_i 's are zeros and ones, because we are talking about binary codes.

(Refer Slide Time: 09:49)

Dual code

- Let C be a linear code with generator matrix G . Then $x \in C_d$ if and only if $xG^T = 0$
- Let G be given by

$$G = \begin{bmatrix} g_0 \\ g_1 \\ \vdots \\ g_{k-1} \end{bmatrix} \quad \text{where } \{g_i\} \text{ is some basis of } G$$

where $\{g_i\}$ is some basis of G .

- Also, $xG^T = (x \cdot g_0, \dots, x \cdot g_{k-1})$.
- If $x \in C_d$, then $x \cdot g_i = 0$ for every i , so $xG^T = 0$.

Now note what do we want to show, we want to show if xG^T is 0, then x must belong to the dual code. And when will x belongs to dual code, we have to show that a code word which belongs to original linear block code C and set of code words which belong to the dual code they are orthogonal to each other or in other words their dot product is zero.

(Refer Slide Time: 10:15)

Dual code

• If $\underline{x} \cdot \underline{G}^T = \underline{0}$ then $\underline{x} \cdot \underline{g}_i = 0$ for every i . If $\underline{c} \in C$, then $\underline{c} = \sum_i \lambda_i \underline{g}_i$ for some binary λ_i , so

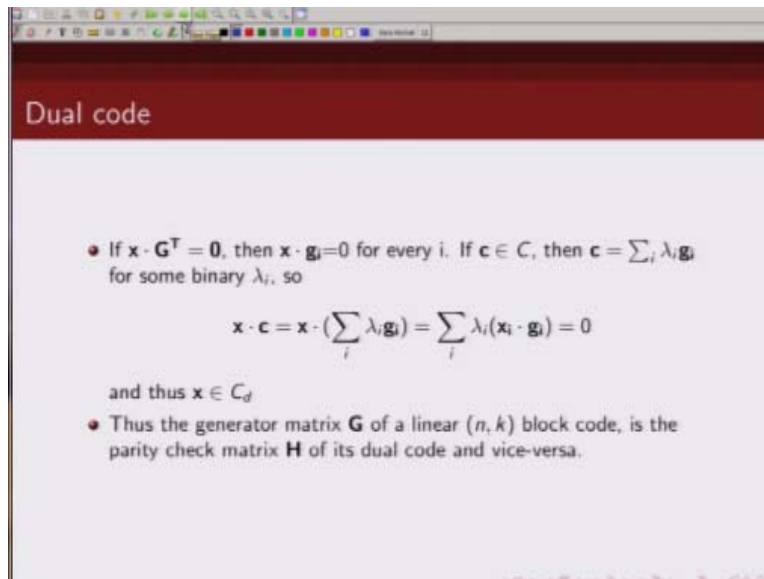
$$\underline{x} \cdot \underline{c} = \underline{x} \cdot \left(\sum_i \lambda_i \underline{g}_i \right) = \sum_i \lambda_i (\underline{x} \cdot \underline{g}_i) = 0$$

and thus $\underline{x} \in C_d$

So let us take a dot product this is x which you want to show that it belongs to the dual code and C is a code word in original code C . So what is $x \cdot c$, $x \cdot c$ can be written $x \cdot \sum \lambda_i g_i$ this, I can write as $\sum \lambda_i x \cdot g_i$. And what do I know from this condition that xG^T is 0 I know from this condition that inner product of x with g_i is 0. So that means this term is equal to 0, then what I have shown that $x \cdot c$ is equal to zero that means x and c are orthogonal to each other.

So if c belongs to the original code this C then x must belong to the dual code of this code C so hence we have show that x belongs to the dual code of C

(Refer Slide Time: 11:33)



Dual code

- If $\mathbf{x} \cdot \mathbf{G}^T = \mathbf{0}$, then $\mathbf{x} \cdot \mathbf{g}_i = 0$ for every i . If $\mathbf{c} \in C$, then $\mathbf{c} = \sum_i \lambda_i \mathbf{g}_i$ for some binary λ_i , so

$$\mathbf{x} \cdot \mathbf{c} = \mathbf{x} \cdot \left(\sum_i \lambda_i \mathbf{g}_i \right) = \sum_i \lambda_i (\mathbf{x} \cdot \mathbf{g}_i) = 0$$

and thus $\mathbf{x} \in C_d$

- Thus the generator matrix \mathbf{G} of a linear (n, k) block code, is the parity check matrix \mathbf{H} of its dual code and vice-versa.

Now we also know that if \mathbf{x} belongs to particular code let us say

(Refer Slide Time: 11:33)

Dual code

- If $\mathbf{x} \cdot \mathbf{G}^T = \mathbf{0}$, then $\mathbf{x} \cdot \mathbf{g}_i = 0$ for every i . If $\mathbf{c} \in C$, then $\mathbf{c} = \sum_i \lambda_i \mathbf{g}_i$ for some binary λ_i , so

$$\mathbf{x} \cdot \mathbf{c} = \mathbf{x} \cdot \left(\sum_i \lambda_i \mathbf{g}_i \right) = \sum_i \lambda_i (\mathbf{x} \cdot \mathbf{g}_i) = 0$$

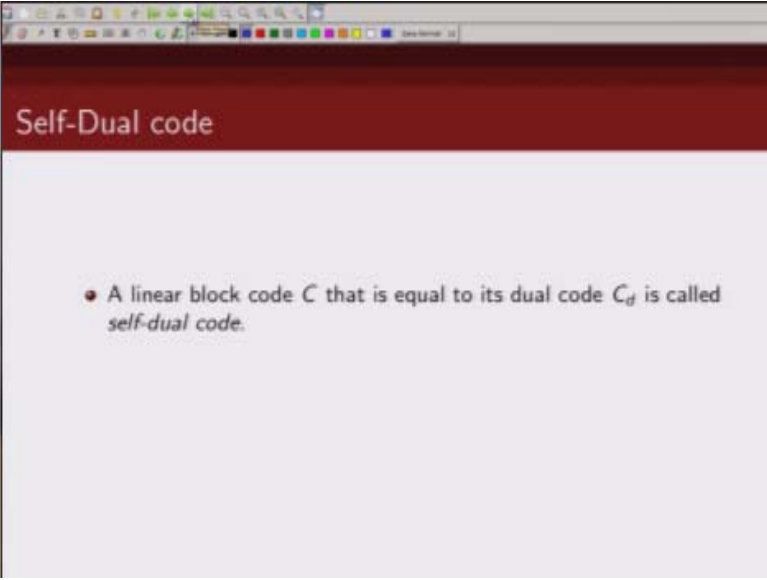
and thus $\mathbf{x} \in C_d$

- Thus the generator matrix \mathbf{G} of a linear (n, k) block code, is the parity check matrix \mathbf{H} of its dual code and vice-versa.

$\mathbf{xH}^T = 0$

A dual code then we know the condition that if for any valid code word we know that let us say \mathbf{x} is a valid code word then we know this property holds, that a code word parity check matrix transform is basically equal to 0 so if we compare this form with this you can immediately guess that the generator matrix for the dual code is given by the parity check matrix of the original linear block code C and the parity check matrix of the dual code is given by this generator matrix of the original code C , so this is what I am saying here the generator matrix \mathbf{G} of a linear block code is the parity check matrix for the dual code and vice-versa.

(Refer Slide Time: 12:50)

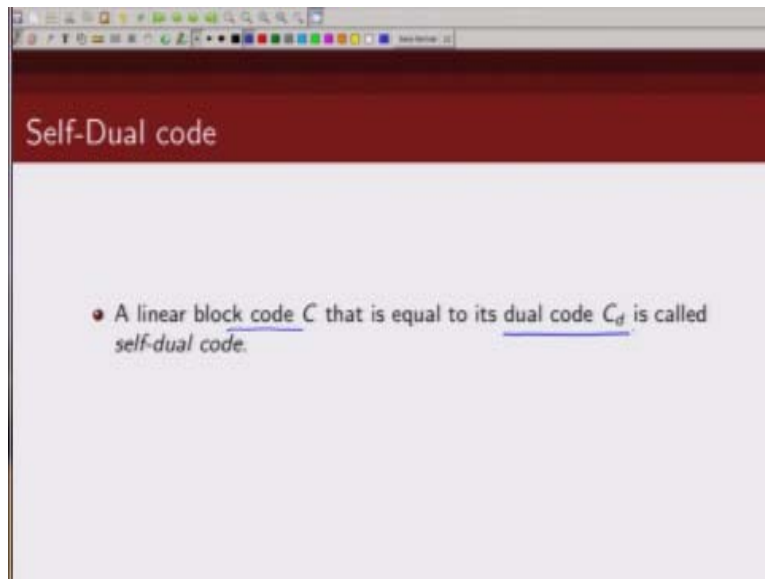


Self-Dual code

- A linear block code C that is equal to its dual code C_d is called *self-dual code*.

Okay now what is a self dual code, if a linear block code

(Refer Slide Time: 12:50)

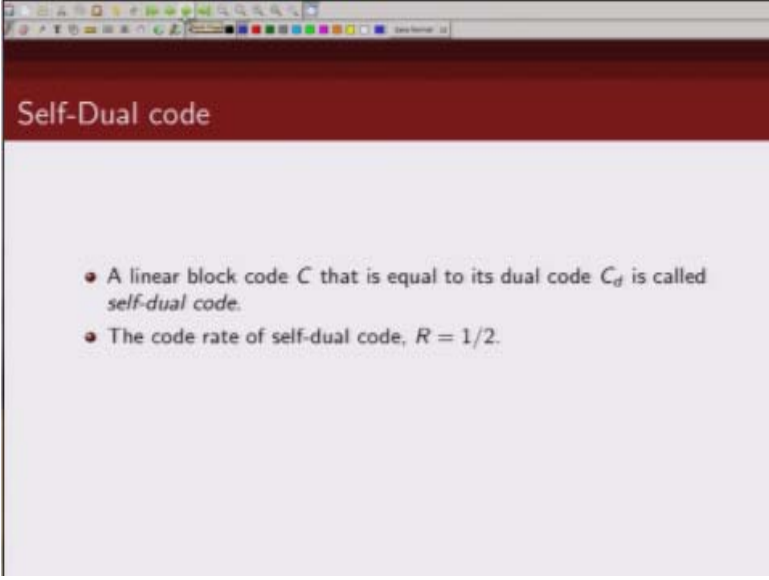


Self-Dual code

- A linear block code C that is equal to its dual code C_d is called *self-dual code*.

Is same is equal to its dual code then it is called a self dual code

(Refer Slide Time: 12:50)

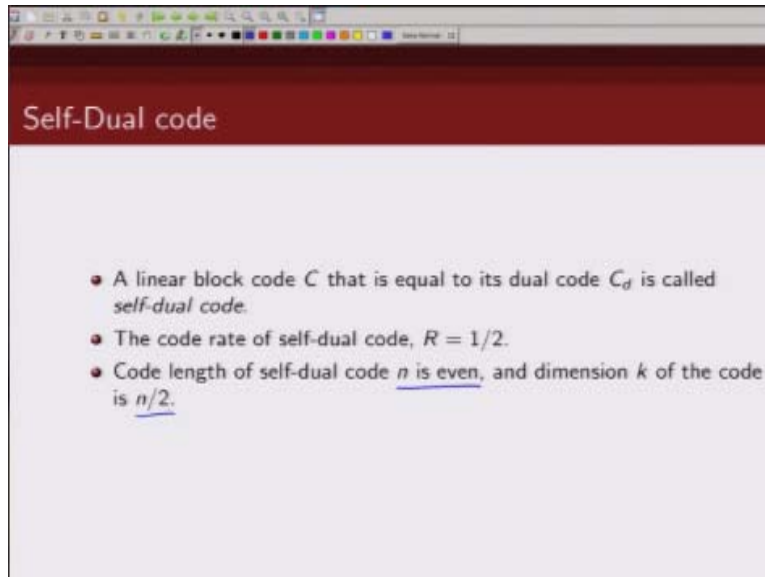


Self-Dual code

- A linear block code C that is equal to its dual code C_d is called *self-dual code*.
- The code rate of self-dual code, $R = 1/2$.

So as you can make out the rate for a self dual code should be half because k should be $n/2$

(Refer Slide Time: 13:15)

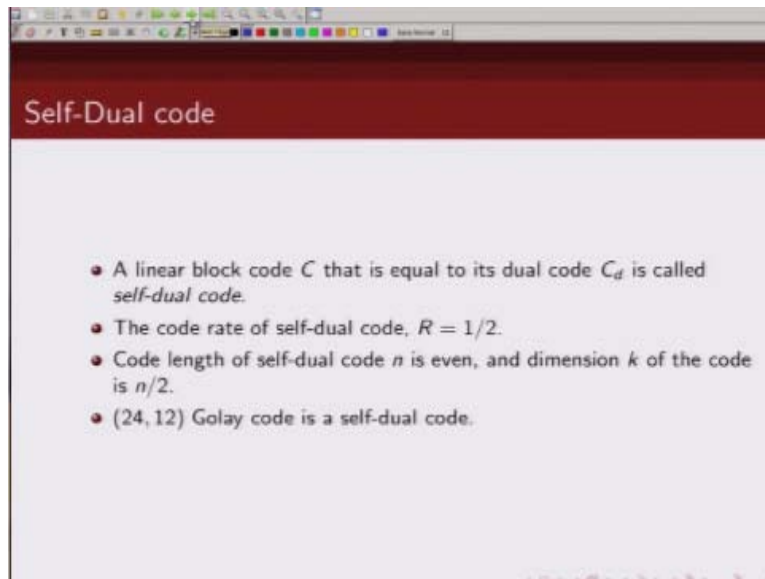


Self-Dual code

- A linear block code C that is equal to its dual code C_d is called *self-dual code*.
- The code rate of self-dual code, $R = 1/2$.
- Code length of self-dual code n is even, and dimension k of the code is $n/2$.

So n is always even for a self dual code and the dimension of the code is always $n/2$

(Refer Slide Time: 13:22)

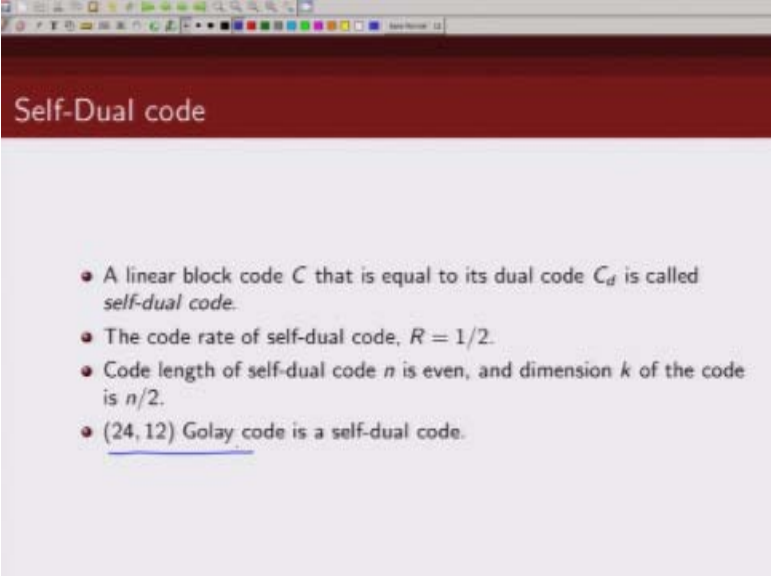


Self-Dual code

- A linear block code C that is equal to its dual code C_d is called *self-dual code*.
- The code rate of self-dual code, $R = 1/2$.
- Code length of self-dual code n is even, and dimension k of the code is $n/2$.
- $(24, 12)$ Golay code is a self-dual code.

An example of self dual code is

(Refer Slide Time: 13:25)



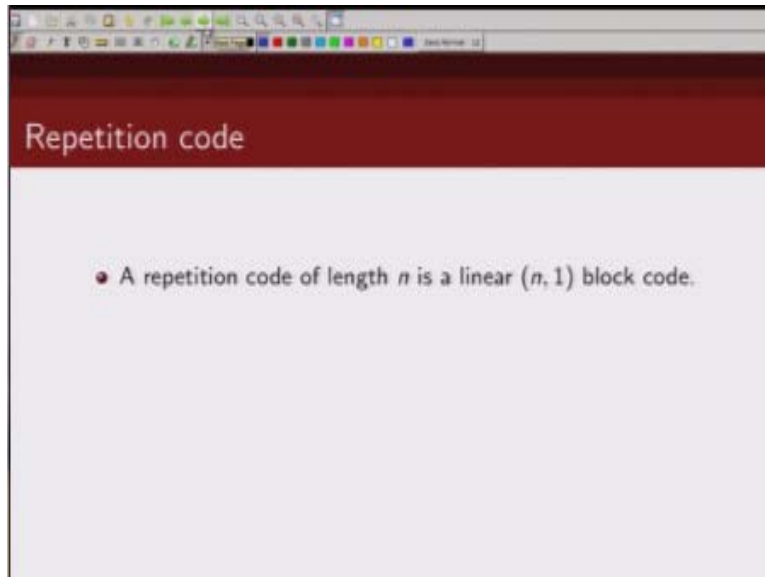
The image is a screenshot of a presentation slide. At the top, there is a dark red header bar with the text "Self-Dual code" in white. Below the header, the slide has a light gray background. It contains a bulleted list of four points. The first point defines a self-dual code. The second point gives the code rate. The third point states conditions on code length and dimension. The fourth point mentions the (24, 12) Golay code, which is underlined.

Self-Dual code

- A linear block code C that is equal to its dual code C_d is called *self-dual code*.
- The code rate of self-dual code, $R = 1/2$.
- Code length of self-dual code n is even, and dimension k of the code is $n/2$.
- (24, 12) Golay code is a self-dual code.

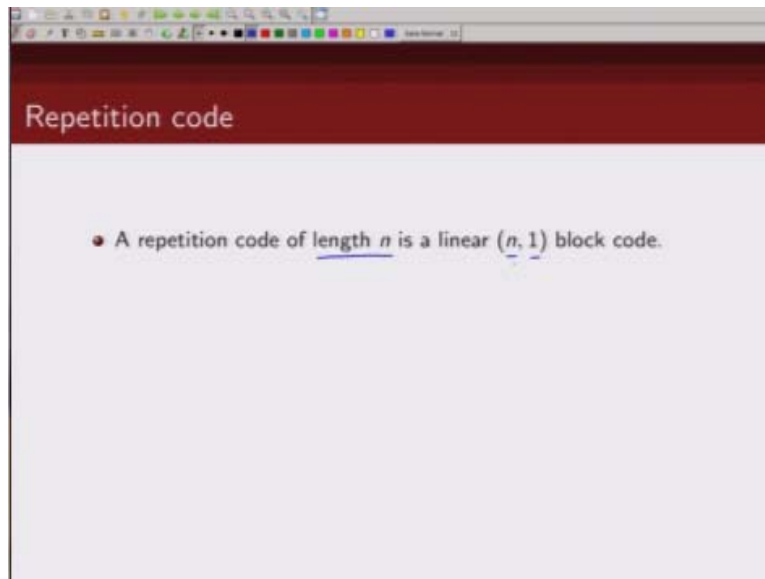
24, 12 Golay code

(Refer Slide Time: 13:27)



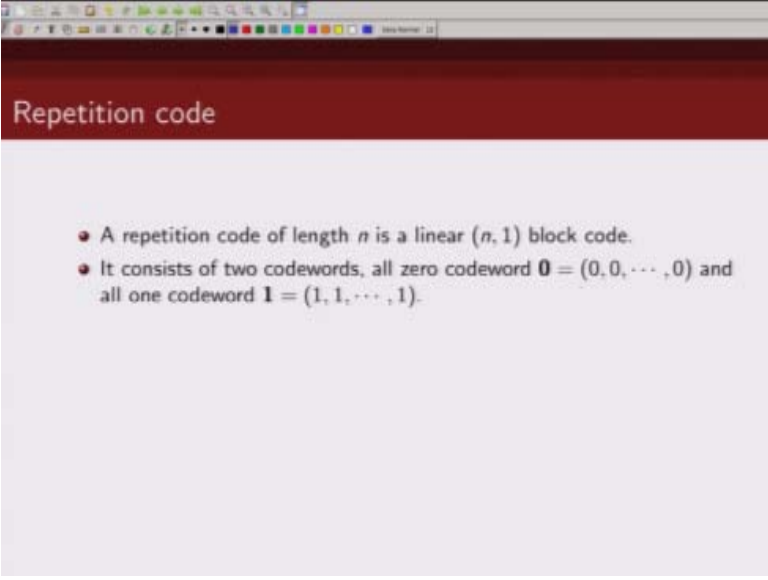
Okay now that we have defined dual code let us now come to the other topic which is some examples of linear block codes. So a very simple example is a repetition code so if

(Refer Slide Time: 13:42)



For repetition code k is 1 and let us say we have a rate $1/n$ repetition code, so the code word length is n this one input bit and n outputs.

(Refer Slide Time: 13:57)

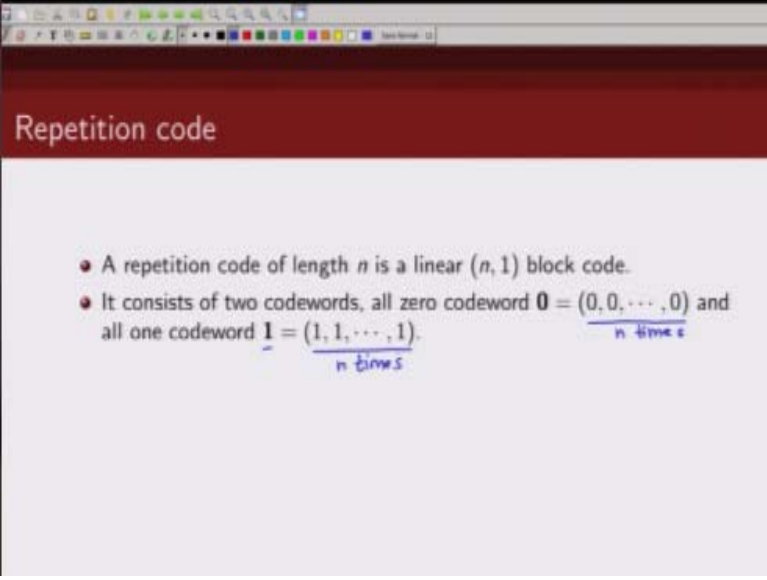


Repetition code

- A repetition code of length n is a linear $(n, 1)$ block code.
- It consists of two codewords, all zero codeword $\mathbf{0} = (0, 0, \dots, 0)$ and all one codeword $\mathbf{1} = (1, 1, \dots, 1)$.

And how is repetition codes generated so we repeat the same information n times, so for a binary repetition code it will have two code words 0 and 1 so if the information sequence information bit is 0 the output will.

(Refer Slide Time: 14:17)

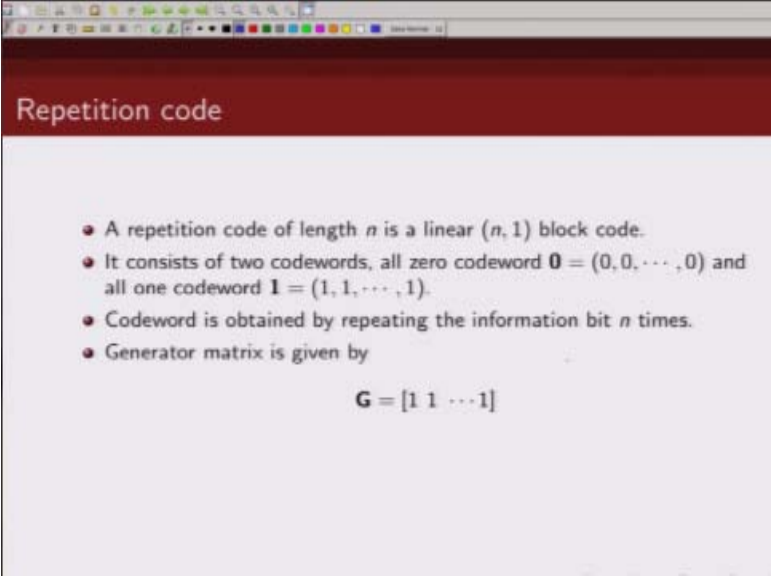


Repetition code

- A repetition code of length n is a linear $(n, 1)$ block code.
- It consists of two codewords, all zero codeword $\mathbf{0} = (0, 0, \dots, 0)$ and all one codeword $\mathbf{1} = (1, 1, \dots, 1)$.
n times

Be all 0's repeated n times and if the input is 1 this will be output will be all 1 repeated n times so there are only two code words in our repetition code.

(Refer Slide Time: 14:36)



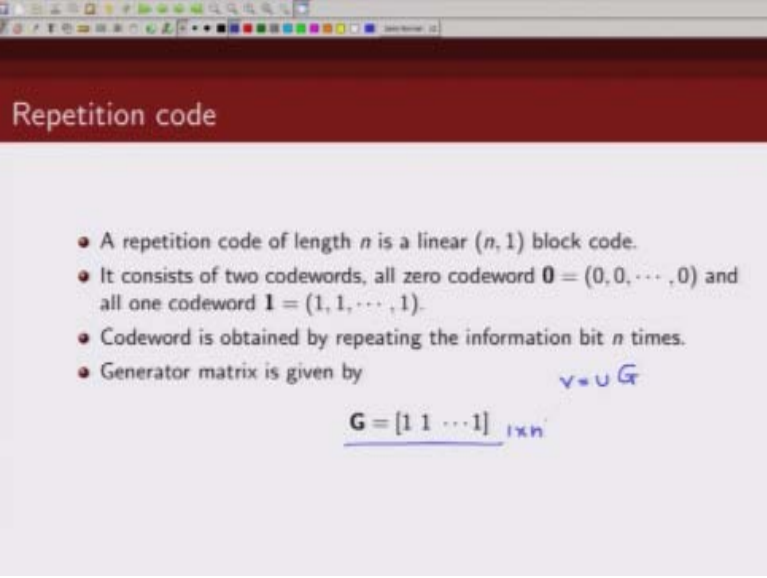
Repetition code

- A repetition code of length n is a linear $(n, 1)$ block code.
- It consists of two codewords, all zero codeword $\mathbf{0} = (0, 0, \dots, 0)$ and all one codeword $\mathbf{1} = (1, 1, \dots, 1)$.
- Codeword is obtained by repeating the information bit n times.
- Generator matrix is given by

$$\mathbf{G} = [1 \ 1 \ \dots \ 1]$$

So we can then write down our generator matrix.

(Refer Slide Time: 14:41)



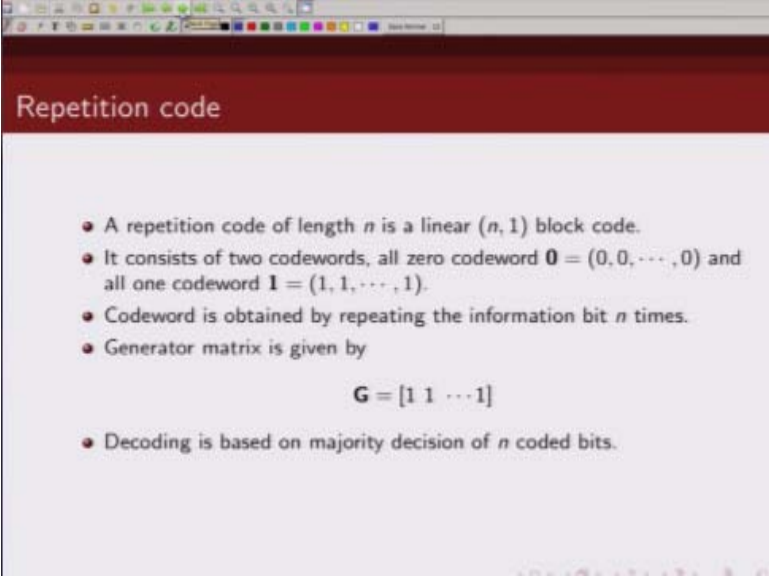
Repetition code

- A repetition code of length n is a linear $(n, 1)$ block code.
- It consists of two codewords, all zero codeword $\mathbf{0} = (0, 0, \dots, 0)$ and all one codeword $\mathbf{1} = (1, 1, \dots, 1)$.
- Codeword is obtained by repeating the information bit n times.
- Generator matrix is given by $\mathbf{v} = \mathbf{u} \mathbf{G}$

$$\mathbf{G} = [1 \ 1 \ \dots \ 1]_{1 \times n}$$

How is generator matrix if \mathbf{v} is the coded sequence \mathbf{u} is information sequence then generator matrix they are related generator matrix in this particular way, so since our output is the bit repeated n time the generator matrix for repetition code will be all ones so this is $1 \times n$.

(Refer Slide Time: 15:07)



Repetition code

- A repetition code of length n is a linear $(n, 1)$ block code.
- It consists of two codewords, all zero codeword $\mathbf{0} = (0, 0, \dots, 0)$ and all one codeword $\mathbf{1} = (1, 1, \dots, 1)$.
- Codeword is obtained by repeating the information bit n times.
- Generator matrix is given by
$$\mathbf{G} = [1 \ 1 \ \dots \ 1]$$
- Decoding is based on majority decision of n coded bits.

Now how do we decode a code which is n coded using repetition code, so what we do is we take a majority decision so let us say for 0 we are sending n 0's and for 1 we are sending n 1's so at the receiver when some of the bits get flipped or they are changed what do we notice as we look at block of n bits and we see what is the majority, is it 0 or 1, if it is 0 majority of the bits are 0 we decide in favor of 0 if it is 1 we decide in favor of 1.

(Refer Slide Time: 15:48)

Repetition code

- A repetition code of length n is a linear $(n, 1)$ block code.
- It consists of two codewords, all zero codeword $\mathbf{0} = (0, 0, \dots, 0)$ and all one codeword $\mathbf{1} = (1, 1, \dots, 1)$.
- Codeword is obtained by repeating the information bit n times.
- Generator matrix is given by

$$\mathbf{G} = [1 \ 1 \ \dots \ 1]$$

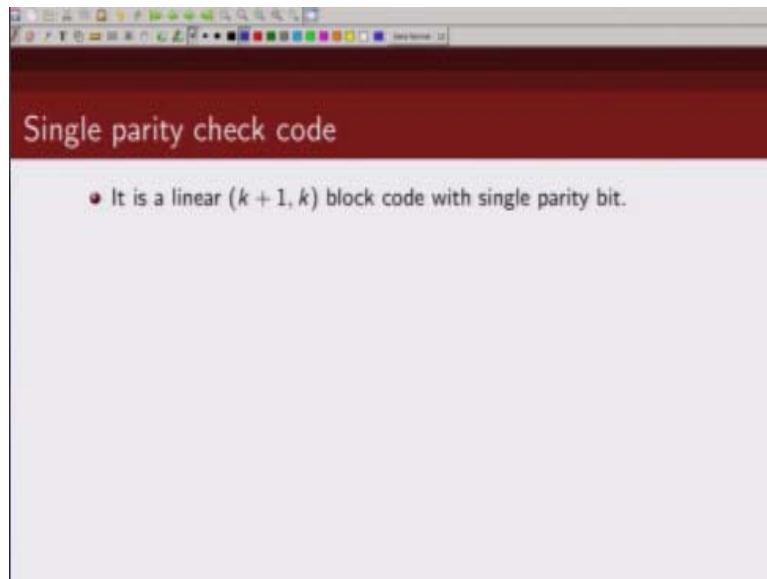
- Decoding is based on majority decision of n coded bits.
- Minimum distance of the code is n .

Handwritten notes in blue ink:

- Next to "Minimum distance of the code is n ": $n-1$ detect
- Next to "Decoding is based on majority decision of n coded bits.": $\left\lfloor \frac{n-1}{2} \right\rfloor$ correct

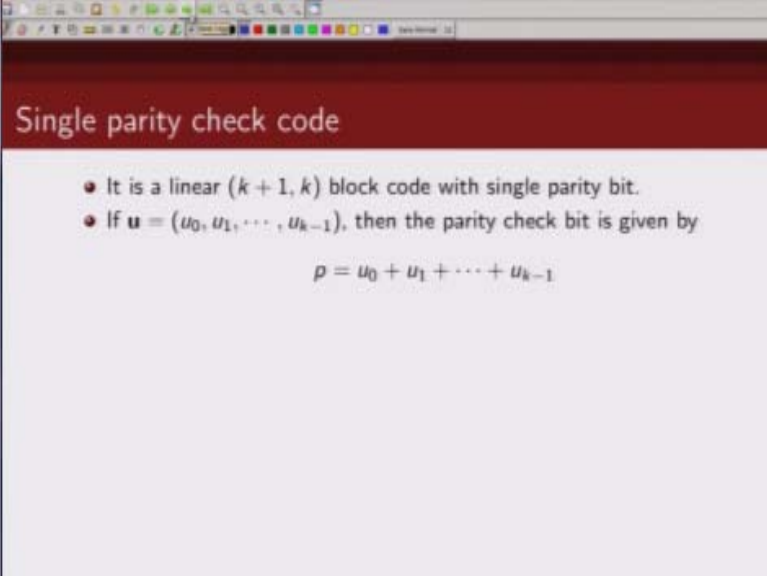
And we can see the minimum distance of this code is n , because there are only two code words all 0 code word and all ones code word. So it can correct $\lfloor n/2 \rfloor$ of that it can correct so many errors, it can correct so many errors and it can detect $n-1$ errors, because any error pattern a weight less than n would not change it into any valid code word so all error pattern up to $n-1$ can be detected by this repetition code.

(Refer Slide Time: 16:33)



Let us look at another example of a linear block code this single parity check code as the name suggest we are adding one single parity check bit.

(Refer Slide Time: 16:48)



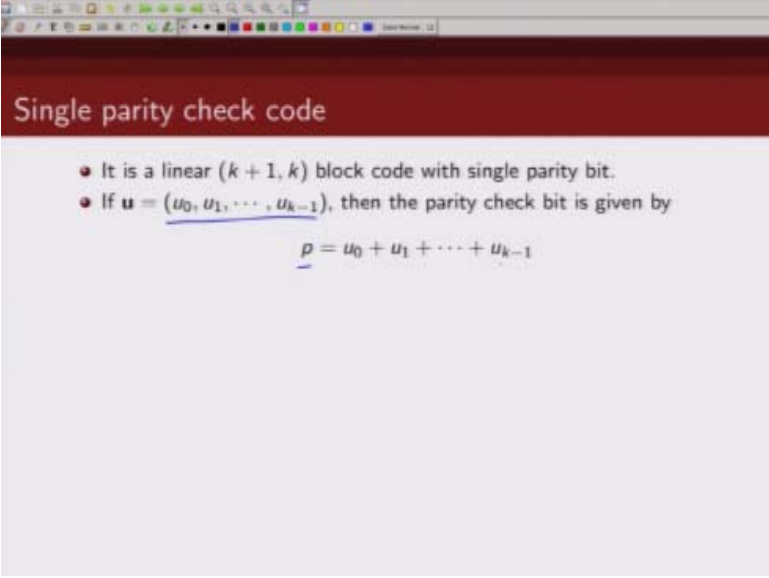
Single parity check code

- It is a linear $(k+1, k)$ block code with single parity bit.
- If $\mathbf{u} = (u_0, u_1, \dots, u_{k-1})$, then the parity check bit is given by

$$p = u_0 + u_1 + \dots + u_{k-1}$$

So the information sequence length is k and n here is given by $k+1$ and how do we generate this single parity check bit, this is as follows.

(Refer Slide Time: 17:04)



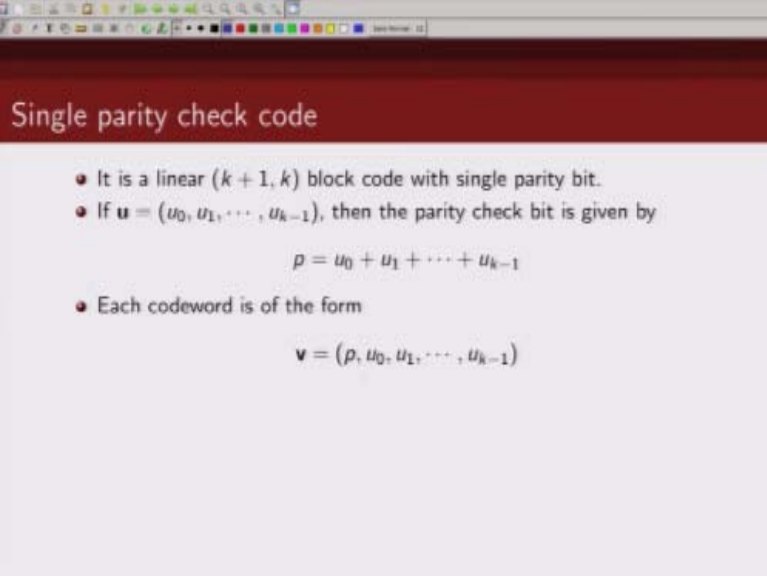
Single parity check code

- It is a linear $(k+1, k)$ block code with single parity bit.
- If $\mathbf{u} = (u_0, u_1, \dots, u_{k-1})$, then the parity check bit is given by

$$p = u_0 + u_1 + \dots + u_{k-1}$$

So if your information sequence is given by this so you have a k bit information sequence let us call it $u_0, u_1, u_2, \dots, u_{k-1}$ then we generate this additional parity bit in this fashion so p is equal to $u_0 + u_1 + u_2 + \dots + u_{k-1}$. So in other words if information sequence is even parity that means some of them basically add up to 0 then p will be 0 or else if the information sequence has odd parity p will be 1. So as you can make out this single parity check code will always have even weight code words.

(Refer Slide Time: 17:50)



Single parity check code

- It is a linear $(k + 1, k)$ block code with single parity bit.
- If $\mathbf{u} = (u_0, u_1, \dots, u_{k-1})$, then the parity check bit is given by

$$p = u_0 + u_1 + \dots + u_{k-1}$$

- Each codeword is of the form

$$\mathbf{v} = (p, u_0, u_1, \dots, u_{k-1})$$

So each cord word I can write in this particular fashion So I have these

(Refer Slide Time: 17:54)

Single parity check code

- It is a linear $(k + 1, k)$ block code with single parity bit.
- If $\mathbf{u} = (u_0, u_1, \dots, u_{k-1})$, then the parity check bit is given by

$$p = u_0 + u_1 + \dots + u_{k-1}$$

- Each codeword is of the form

$$\mathbf{v} = (p, u_0, u_1, \dots, u_{k-1})$$

\uparrow
parity bit

K information bits and 1 parity bit.

(Refer Slide Time: 18:06)

Single parity check code

- It is a linear $(k+1, k)$ block code with single parity bit.
- If $\mathbf{u} = (u_0, u_1, \dots, u_{k-1})$, then the parity check bit is given by
$$p = u_0 + u_1 + \dots + u_{k-1}$$
- Each codeword is of the form
$$\mathbf{v} = (p, u_0, u_1, \dots, u_{k-1})$$

• The generator matrix for the single parity check code in systematic form is given by

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

So I can write the same thing in the form of generator matrix.

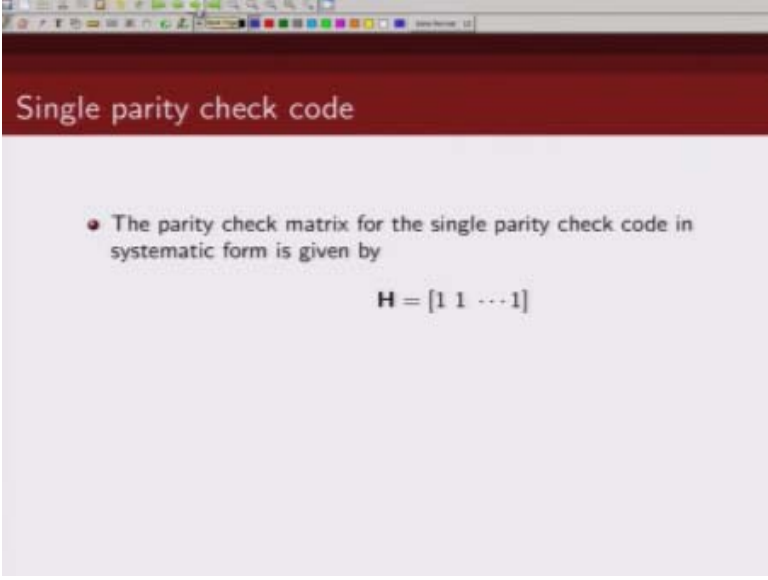
(Refer Slide Time: 18:11)

Single parity check code

- It is a linear $(k+1, k)$ block code with single parity bit.
- If $\mathbf{u} = (u_0, u_1, \dots, u_{k-1})$, then the parity check bit is given by
$$p = u_0 + u_1 + \dots + u_{k-1}$$
- Each codeword is of the form
$$\mathbf{v} = (p, u_0, u_1, \dots, u_{k-1})$$
- The generator matrix for the single parity check code in systematic form is given by
$$\mathbf{G} = \begin{bmatrix} 1 & 1 & 0 & 0 & \dots & 0 \\ 1 & 0 & 1 & 0 & \dots & 0 \\ 1 & 0 & 0 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & 0 & 0 & 0 & \dots & 1 \end{bmatrix} \quad \mathbf{v} = \mathbf{uG}$$

As we know our coded sequence can be written as input times generator matrix so you can see first bit is a parity bit which is sum of all u_i 's, so first column will be all ones and then this will be a identity matrix because the second bit is u_0 , third bit is u_1 , u_2 and so on. So this is a identity matrix okay

(Refer Slide Time: 18:48)



Single parity check code

- The parity check matrix for the single parity check code in systematic form is given by

$$\mathbf{H} = [1 \ 1 \ \dots \ 1]$$

Now we can write down the parity check matrix for this also.

(Refer Slide Time: 18:54)

Single parity check code

- It is a linear $(k+1, k)$ block code with single parity bit.
- If $\mathbf{u} = (u_0, u_1, \dots, u_{k-1})$, then the parity check bit is given by
$$p = u_0 + u_1 + \dots + u_{k-1}$$
- Each codeword is of the form
$$\mathbf{v} = (p, u_0, u_1, \dots, u_{k-1})$$
- The generator matrix for the single parity check code in systematic form is given by
$$\mathbf{G} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & \dots & 0 \\ 1 & 0 & 1 & 0 & 0 & \dots & 0 \\ 1 & 0 & 0 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & 0 & 0 & \dots & 1 \end{bmatrix}$$
$$\mathbf{v} = \mathbf{uG}$$
$$\mathbf{G} = [\mathbf{P} : \mathbf{I}]$$
$$\mathbf{H} = [\mathbf{I} : \mathbf{P}^T]$$

As you can see this is this generator matrix is in systematic form. So what would be the corresponding -- this is of the form like this I have P: I so parity check matrix will be I: \mathbf{P}^T right. So what I have here is

(Refer Slide Time: 19:25)

Single parity check code

- The parity check matrix for the single parity check code in systematic form is given by

$$\mathbf{H} = [1 \ 1 \ \cdots \ 1]$$

This is my \mathbf{P}^T and this is my \mathbf{I} .

(Refer Slide Time: 19:34)

Single parity check code

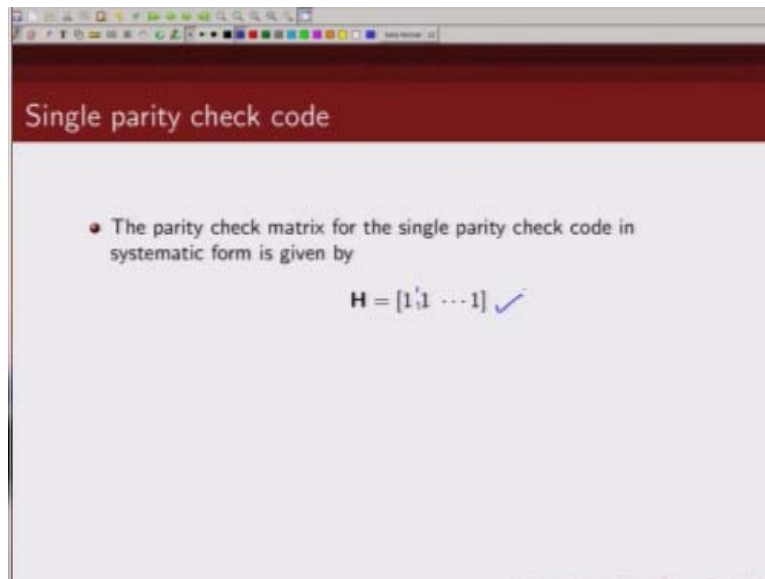
- It is a linear $(k+1, k)$ block code with single parity bit.
- If $\mathbf{u} = (u_0, u_1, \dots, u_{k-1})$, then the parity check bit is given by
$$p = u_0 + u_1 + \dots + u_{k-1}$$
- Each codeword is of the form
$$\mathbf{v} = (p, u_0, u_1, \dots, u_{k-1})$$
- The generator matrix for the single parity check code in systematic form is given by

$$\mathbf{G} = \begin{bmatrix} \overset{\text{P}}{1} & \overset{\text{I}}{1} & 0 & 0 & 0 & \dots & 0 \\ 1 & 0 & 1 & 0 & 0 & \dots & 0 \\ 1 & 0 & 0 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & 0 & 0 & \dots & 1 \end{bmatrix}$$

$\mathbf{v} = \mathbf{u}\mathbf{G}$
 $\mathbf{G} = [\mathbf{P} : \mathbf{I}]$
 $\mathbf{H} = [\mathbf{I} : \mathbf{P}^T]$

This part is my -- this part is my P and this part is my I, so parity check matrix will be $\mathbf{I} \mathbf{P}^T$

(Refer Slide Time: 19:48)



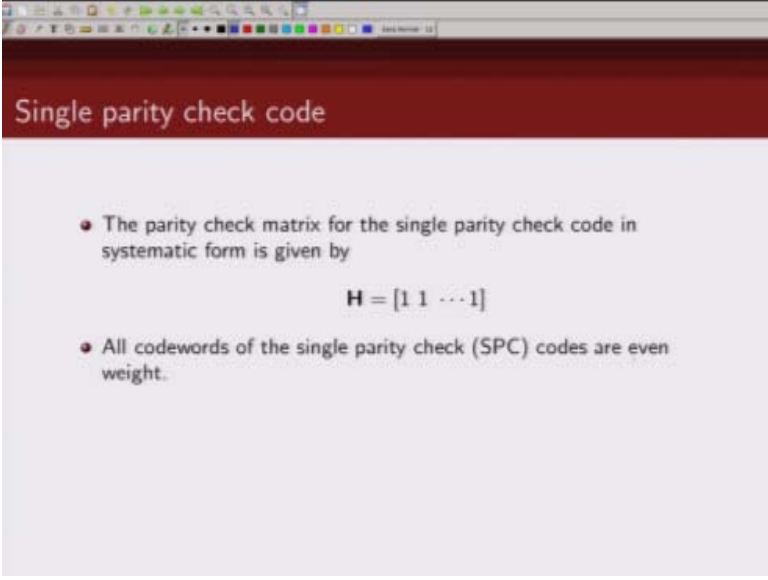
Single parity check code

- The parity check matrix for the single parity check code in systematic form is given by

$$\mathbf{H} = [1^t \mid 1] \quad \checkmark$$

So that is this.

(Refer Slide Time: 19:54)

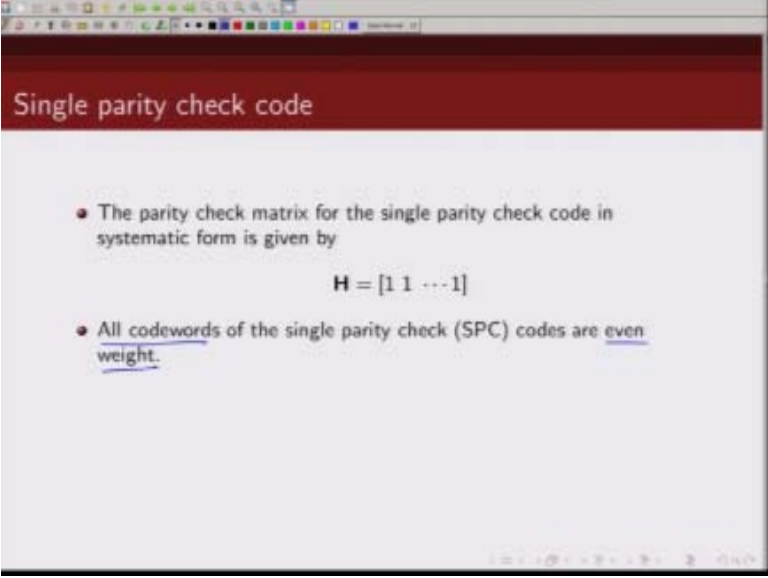


Single parity check code

- The parity check matrix for the single parity check code in systematic form is given by
$$\mathbf{H} = [1 \ 1 \ \dots \ 1]$$
- All codewords of the single parity check (SPC) codes are even weight.

As I mentioned a very interesting property of single parity check codes.

(Refer Slide Time: 20:00)



Single parity check code

- The parity check matrix for the single parity check code in systematic form is given by
$$\mathbf{H} = [1 \ 1 \ \dots \ 1]$$
- All codewords of the single parity check (SPC) codes are even weight.

All code words of single parity check codes are of even weight.

(Refer Slide Time: 20:07)

Single parity check code

- The parity check matrix for the single parity check code in systematic form is given by
$$\mathbf{H} = [1 \ 1 \ \dots \ 1]$$
- All codewords of the single parity check (SPC) codes are even weight.
- Minimum distance of SPC code is 2.

And we can see the minimum distance of the code is 2 how simple, very simple way to check is look at the columns of the parity check matrix what is the minimum number of columns that add up to 0 in this case 2, any 2 columns will add up to 0 so the minimum distance of this code is 2.

(Refer Slide Time: 20:31)

Single parity check code

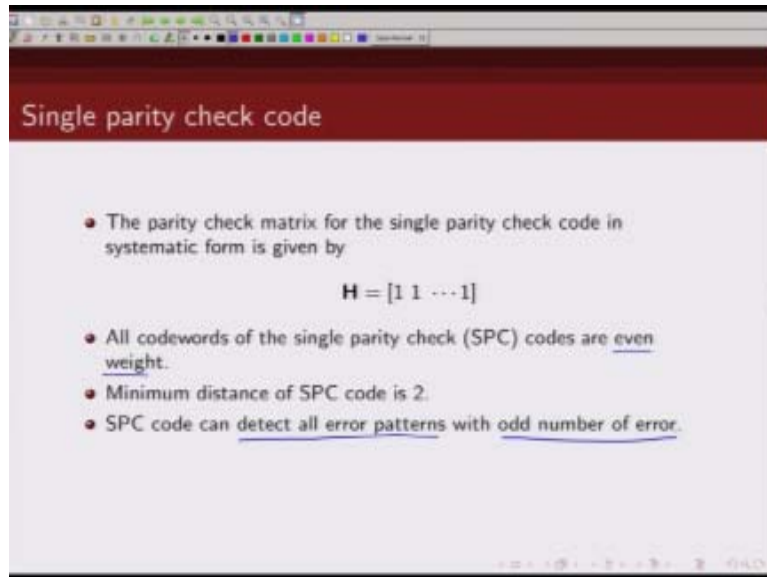
- The parity check matrix for the single parity check code in systematic form is given by

$$\mathbf{H} = [1 \ 1 \ \dots \ 1]$$

- All codewords of the single parity check (SPC) codes are even weight.
- Minimum distance of SPC code is 2.
- SPC code can detect all error patterns with odd number of error.

Now since all code words have even weight any odd code, any odd error pattern can be detected by single parity check code, because any odd pattern, any odd weight pattern is not a valid code word.

(Refer Slide Time: 20:54)



Single parity check code

- The parity check matrix for the single parity check code in systematic form is given by
$$\mathbf{H} = [1 \ 1 \ \dots \ 1]$$
- All codewords of the single parity check (SPC) codes are even weight.
- Minimum distance of SPC code is 2.
- SPC code can detect all error patterns with odd number of error.

So a single parity check code can detect all error patterns which has even number of errors, odd numbers of errors right. So as long as error pattern has odd weight single parity check code can detect it.

(Refer Slide Time: 21:15)

Single parity check code

- The parity check matrix for the single parity check code in systematic form is given by
$$\mathbf{H} = [1 \ 1 \ \dots \ 1]$$
- All codewords of the single parity check (SPC) codes are even weight.
- Minimum distance of SPC code is 2.
- SPC code can detect all error patterns with odd number of error.
- The $(n, n-1)$ SPC code and $(n, 1)$ repetition code are dual to each other.

And it is not very difficult to see that a single parity check code and repetition codes are dual to each other you can see basically the parity check matrix of single parity check code is same --

(Refer Slide Time: 21:32)

Single parity check code

- The parity check matrix for the single parity check code in systematic form is given by

$$\mathbf{H} = [1 \ 1 \ \dots \ 1]$$

- All codewords of the single parity check (SPC) codes are even weight.
- Minimum distance of SPC code is 2.
- SPC code can detect all error patterns with odd number of error.

(Refer Slide Time: 21:35)

Repetition code

- A repetition code of length n is a linear $(n, 1)$ block code.
- It consists of two codewords, all zero codeword $\mathbf{0} = (0, 0, \dots, 0)$ and all one codeword $\mathbf{1} = (1, 1, \dots, 1)$.
- Codeword is obtained by repeating the information bit n times.
- Generator matrix is given by

$$\mathbf{G} = [1 \ 1 \ \dots \ 1]$$

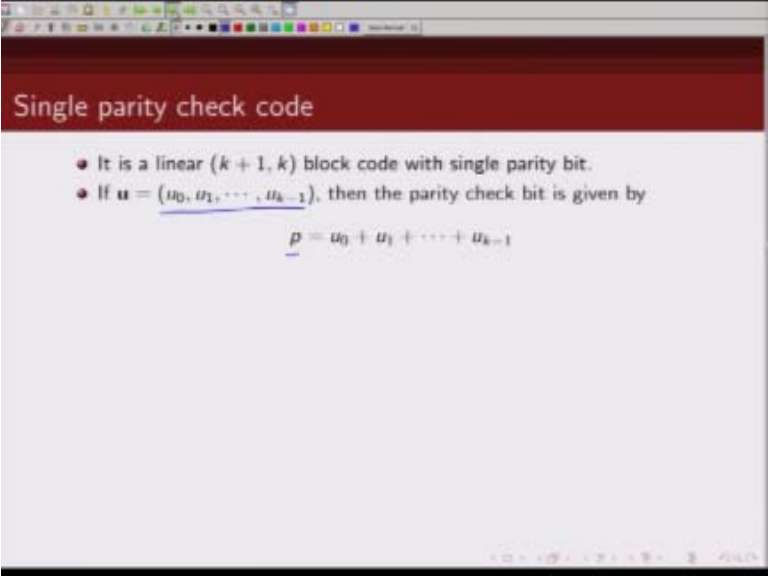
- Decoding is based on majority decision of n coded bits.
- Minimum distance of the code is n .

Handwritten notes in blue ink:

- Next to "Minimum distance of the code is n ":
 $\left. \begin{array}{l} \frac{n-1}{2} \\ n-1 \end{array} \right\} \begin{array}{l} \text{correct} \\ \text{detect} \end{array}$

As generator matrix of the repetition code and similarly the parity check matrix of repetition code is same as.

(Refer Slide Time: 21:43)



Single parity check code

- It is a linear $(k+1, k)$ block code with single parity bit.
- If $\mathbf{u} = (u_0, u_1, \dots, u_{k-1})$, then the parity check bit is given by

$$\underline{p = u_0 + u_1 + \dots + u_{k-1}}$$

The generator matrix of the of the single parity check code.

(Refer Slide Time: 21:44)

Single parity check code

- It is a linear $(k+1, k)$ block code with single parity bit.
- If $\mathbf{u} = (u_0, u_1, \dots, u_{k-1})$, then the parity check bit is given by

$$p = u_0 + u_1 + \dots + u_{k-1}$$

- Each codeword is of the form

$$\mathbf{v} = (p, u_0, u_1, \dots, u_{k-1})$$

\uparrow
parity bit

(Refer Slide Time: 21:44)

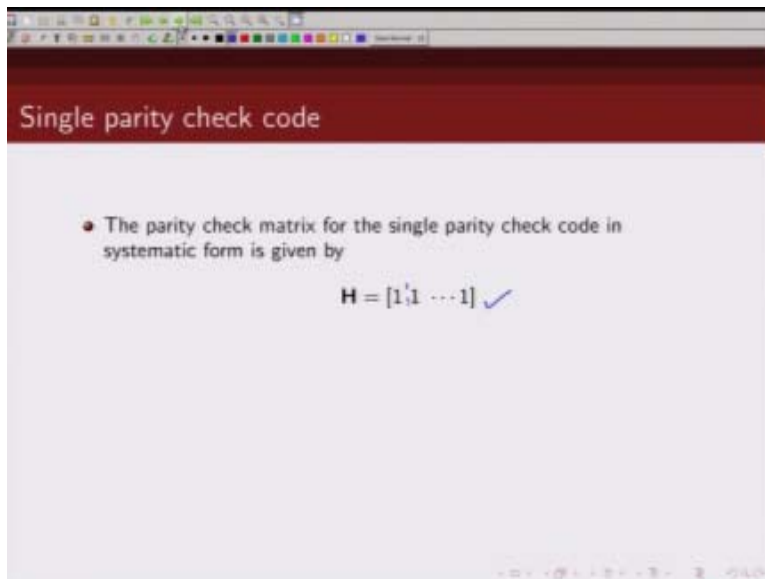
Single parity check code

- It is a linear $(k+1, k)$ block code with single parity bit.
- If $\mathbf{u} = (u_0, u_1, \dots, u_{k-1})$, then the parity check bit is given by
$$p = u_0 + u_1 + \dots + u_{k-1}$$
- Each codeword is of the form
$$\mathbf{v} = (p, u_0, u_1, \dots, u_{k-1})$$
- The generator matrix for the single parity check code in systematic form is given by

$$G = \begin{bmatrix} \overbrace{1}^p & \overbrace{1}^I & 0 & 0 & \dots & 0 \\ 1 & 0 & 1 & 0 & \dots & 0 \\ 1 & 0 & 0 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & 0 & 0 & 0 & \dots & 1 \end{bmatrix}$$

$v = uG$
 $G = [P : I]$
 $H = [I : P^T]$

(Refer Slide Time: 21:45)



Single parity check code

- The parity check matrix for the single parity check code in systematic form is given by

$$\mathbf{H} = [1, 1 \dots 1] \checkmark$$

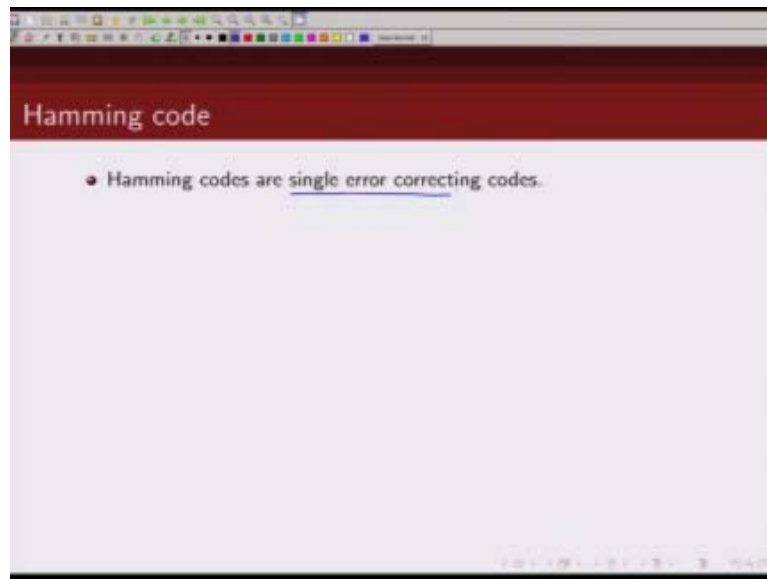
(Refer Slide Time: 21:45)

Single parity check code

- The parity check matrix for the single parity check code in systematic form is given by
$$\mathbf{H} = [1 \ 1 \ \dots 1]$$
- All codewords of the single parity check (SPC) codes are even weight.
- Minimum distance of SPC code is 2.
- SPC code can detect all error patterns with odd number of error.
- The $(n, n-1)$ SPC code and $(n, 1)$ repetition code are dual to each other.

And you can check the dimensions of the single parity check code is $n - 1$ and this dimension is 1. So single parity check code and repetition codes are dual to each other.

(Refer Slide Time: 22:01)



Next we consider hamming codes, so hamming codes are single error correcting codes these are single error correcting codes.

(Refer Slide Time: 22:13)

Hamming code

- Hamming codes are single error correcting codes.
- For any $m \geq 3$, there exist a Hamming code with following parameters:

Code length:	$n = 2^m - 1$
Information bits:	$k = 2^m - m - 1$
Parity bits:	$n - k = m$
Error correcting capability:	$t = 1$
Minimum distance:	$d_{\min} = 3$

And they are described by these parameters. So for any m greater than equal to 3 the code word length is given by this the number of parity bits is equal to m , so number of information bits are $n - k$, $n - m$ so this is $2^m - m - 1$ so these my number of information bits, it has minimum distance of 3 so it can correct single error and it can detect 2 errors. So how do we describe hamming code.

(Refer Slide Time: 22:55)

Hamming code

- Hamming codes are single error correcting codes.
- For any $m \geq 3$, there exist a Hamming code with following parameters

Code length:	$n = 2^m - 1$
Information bits:	$k = 2^m - m - 1$
Parity bits:	$n - k = m$
Error correcting capability:	$t = 1$
Minimum distance:	$d_{\min} = 3$

• The parity check matrix in systematic form

$$H = [I_m : P^T]$$

where the $2^m - m - 1$ columns of P^T consists of all m-tuples of weight 2 or more.

We can describe the hamming code by generator matrix or the parity check matrix. So a parity check matrix of a hamming code consist of all nonzero m-tuples. So if the number of parity bits as I said is m, then code word length is $2^m - 1$ and the entries in the parity check matrix are all non zero m-tuples. Now how may m- tuples do we have, we have total 2^m m-tuples and out of those, so these are like 000, 001 and you can go on up to all ones and this m times m. Now if we removed all zero 1, so total number of nonzero m-tuples is basically given by $2^m - 1$.

So the columns of the parity check matrix of hamming code are nothing but non zero m-tuples, so they are total $2^m - 1$ of them okay. And if you want to write the parity check matrix of a hamming code in a systematic form then we can write it in this way. So parity check matrix will have an identity and then some matrix P, since identity matrix will consist of all patterns all m-tuples of weight 1, P^T should consist of all m-tuples of weight more than 1. So it consist of all m-tuples of weight 2, 3,4 up to m.

(Refer Slide Time: 25:01)

Hamming code

- For $m=3$ the Hamming code is of length $n = 2^3 - 1 = 7$, $k = 2^3 - 3 - 1 = 4$, that has parity check matrix H ,

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

and generator matrix G ,

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

Handwritten notes on the slide:

- $n = 2^m - 1 = 7$
- $n - k = m = 3$
- $k = 7 - 3 = 4$
- $(7, 4)$
- 7 nonzero m-tuples: 001, 010, 011, 100, 101, 110, 111

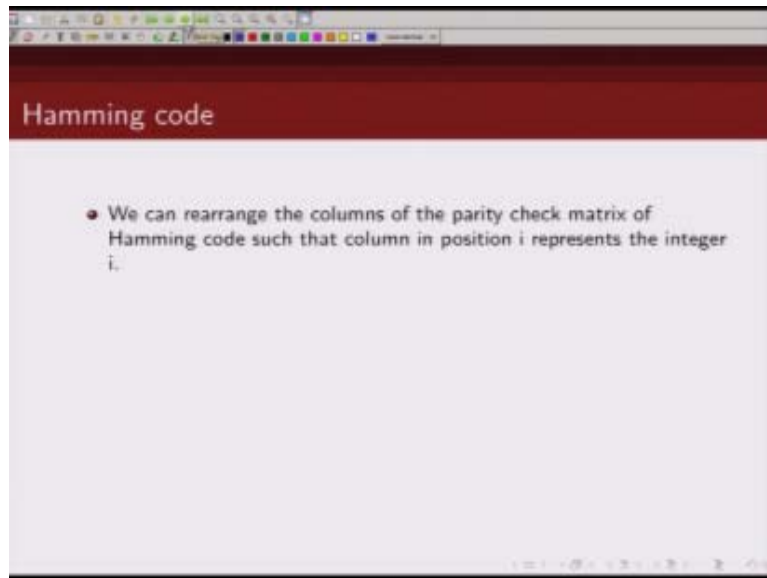
So let us take an example, let us consider $m = 3$. So if $m=3$ we know n is $2^m - 1$ so length of the block code in this case will be 7. Now how many numbers of parity bits, number of parity bits is given by m so this will be 3. So then what is the number of information bits this is $7 - 3$ that is 4. So this is a 7, 4 code. Now as I said the parity check matrix consist of all nonzero m -tuples, so m here is 3 so let us list all nonzero m -tuples so we can write 001, 010, 011, 100, 101, 110, 111 so these are the 7 nonzero m -tuples right.

And if you want to write your parity check matrix in a systematic form what will you do. So this is your I , so I am writing these patterns which are like 1, 2 and 3, I am writing these patterns and then what I write here is the other m -tuples. So you can see this is m -tuple of weight 1, these 3 are m -tuples of weight 1, and then this is a m -tuple of weight 2, weight 2 and this is m -tuple of weight 3. Now since this matrix is of the form I and P , I can write the generator matrix this will be $P^T I$.

So we can write this, so P^T so 1011 will come here as 1011, 1110 will come as 1110 and 0111 will come as 0111. And then this is the identity matrix, so this our generator matrix or this 74 hamming code. The point to be noted here is, once you will fix your m all other code parameters

are fixed, n is fixed right, because n is 2^{m-1} and number of parity bits is equal to m . So for the hamming code once you fix m the other parameters of the code are fixed.

(Refer Slide Time: 28:00)



Now let us see how we can correct errors using hamming code.

(Refer Slide Time: 28:07)

Hamming code

• For $m=3$ the Hamming code is of length $n = 2^3 - 1 = 7$,
 $k = 2^3 - 3 - 1 = 4$, that has parity check matrix **H**.

$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \left[\frac{d-1}{2} \right] = 1$

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

and generator matrix **G**,

$G = [P^T : I]$

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

$n = 2^m - 1 = 7$
 $n - k = m = 3$
 $k = 7 - 3 = 4$
 $(7, 4)$

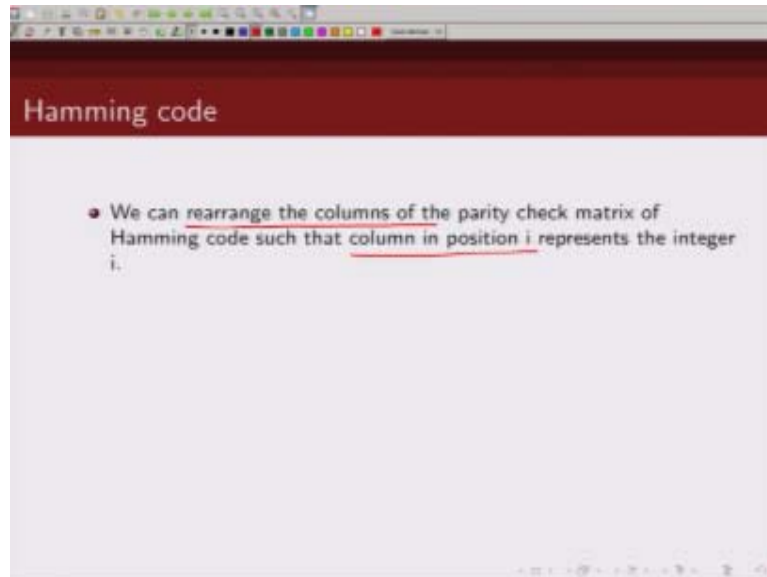
0	0	0	1	1	1	1
0	1	1	0	0	1	1
1	0	1	0	1	0	1

7 - ones

As we said this has minimum distance 3 you can see here. So let us take any 3 columns let us just take this column, column number 1, column number 2, and column number 4 you can see column number 1, 2 and 4 they will add up to 000. So if 3 and that is the minimum number of columns that will add up to 0 so which means the minimum distance of this code is 3.

If the minimum distance of this code is 3, what is the error correcting capability of this code it is $d-1/2$ floor off that and this comes out to be 1. So hamming code can correct single error. So let us see how we can use hamming codes to correct single error.

(Refer Slide Time: 29:13)



So first thing what we will do is we will rearrange the columns of the parity check matrix of hamming code such that column in position i represent integer i .

(Refer Slide Time: 29:32)

Hamming code

- We can rearrange the columns of the parity check matrix of Hamming code such that column in position i represents the integer i .
- For example for $m = 3$, the Hamming code is of length $n = 2^3 - 1 = 7$, $k = 2^3 - 3 - 1 = 4$, that has parity check matrix H .

$$H = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{matrix} \leftarrow \text{LSB} \\ \\ \leftarrow \text{MSB} \end{matrix}$$

So as we said the parity check matrix of hamming code are all nonzero m -tuples. So we will arrange them in such a way such that the i^{th} column represent i^{th} bit. Now what do I mean by that, so let us go back to our same example $m=3$ so in this case $n=7$ and $k=4$. Now note the way I have arranged this, this is my MSB. And this is my LSB least significant bit and the most significant bit.

(Refer Slide Time: 30:15)

Hamming code

- We can rearrange the columns of the parity check matrix of Hamming code such that column in position i represents the integer i .
- For example for $m = 3$, the Hamming code is of length $n = 2^3 - 1 = 7$, $k = 2^3 - 3 - 1 = 4$, that has parity check matrix \mathbf{H} ,

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

- Here the column $(x, y, z)^T$ represents the number $x(2^0) + y(2^1) + z(2^2)$.

So each column represents a number in this particular way.

(Refer Slide Time: 30:18)

Hamming code

- We can rearrange the columns of the parity check matrix of Hamming code such that column in position i represents the integer i .
- For example for $m = 3$, the Hamming code is of length $n = 2^3 - 1 = 7$, $k = 2^3 - 3 - 1 = 4$, that has parity check matrix H .

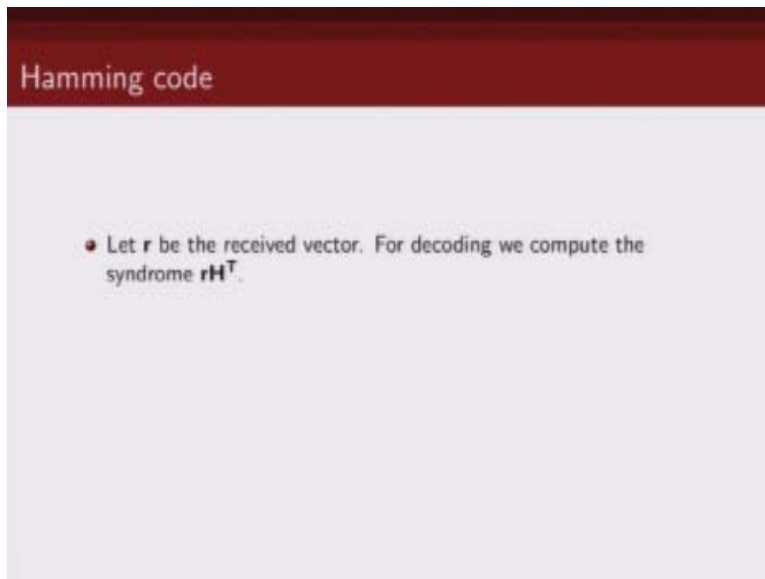
$$H = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

1 2 3 4 5 6 7

- Here the column $(x, y, z)^T$ represents the number $x(2^0) + y(2^1) + z(2^2)$.

So this is a least significant, this a least significant bit, this is the most significant bit. So this is 001 that is binary 1, so this is my column number 1, this is 010 that is 2, 011 that is 3, this is 4, 5, 6, 7. So note what I did was, I rearranged a column of the parity check matrix of hamming code such that now i^{th} column represent i^{th} number basically, so column in i^{th} position represent the integer i , now if I do that.

(Refer Slide Time: 31:08)




Hamming code

- Let \mathbf{r} be the received vector. For decoding we compute the syndrome \mathbf{rH}^T .

Then let us see how this helps us in locating the position of the error.

(Refer Slide Time: 31:18)



Hamming code

- Let \mathbf{r} be the received vector. For decoding we compute the syndrome \mathbf{rH}^T .

So let us say \mathbf{r} is my receive sequence receive vector this is my received n bit vector, and I am interested in knowing whether \mathbf{r} is an error, and if \mathbf{r} is an error let us -- because the single error correcting code, let us say a single error has happened then I am interested in correcting this single error, so what do we do?

(Refer Slide Time: 31:42)



Hamming code

- Let \underline{r} be the received vector. For decoding we compute the syndrome \underline{rH}^T .

The first thing to find out whether error has occurred or not, the first thing that we do is recompute the syndrome, how do you compute the simple syndrome? We take \underline{rH}^T and what is our H?

(Refer Slide Time: 31:56)

Hamming code

- We can rearrange the columns of the parity check matrix of Hamming code such that column in position i represents the integer i .
- For example for $m = 3$, the Hamming code is of length $n = 2^3 - 1 = 7$, $k = 2^3 - 3 - 1 = 4$, that has parity check matrix \mathbf{H} ,

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

1 2 3 4 5 6 7

- Here the column $(x, y, z)^T$ represents the number $x(2^0) + y(2^1) + z(2^2)$.

H is our matrix this matrix which is arranged where their columns are arranged in such a way such that column in i^{th} position represent integer i .

(Refer Slide Time: 32:09)

Hamming code

- Let \underline{r} be the received vector. For decoding we compute the syndrome \underline{rH}^T .

(Refer Slide Time: 32:11)

Hamming code

- Let \mathbf{r} be the received vector. For decoding we compute the syndrome \mathbf{rH}^T .
- If at most one error has occurred, the syndrome would be either the zero vector or a column of \mathbf{H} .

So since is a single error correcting code it can only correct at most one error.

(Refer Slide Time: 32:17)

Hamming code

- Let \mathbf{r} be the received vector. For decoding we compute the syndrome \mathbf{rH}^T .
- If atmost one error has occurred, the syndrome would be either the zero vector or a column of \mathbf{H} .
no error single error

So we assume let us say atmost one error has occurred, atmost of course if there is no error then syndrome would be an all zero vector and if a single error has happened the syndrome would have been a nonzero vector. Now what you will observe is, if a single error happens then that single error will be basically whatever is a syndrome from there we can find out which bit location is in error.

So in case of hamming code if single error has happened and you have arranged your parity check matrix in such a way such that column in position i represent integer i , then the syndrome would be either zero vector in case there is no error or otherwise in case of single error it would be a column of this parity check matrix.

(Refer Slide Time: 33:19)

Hamming code

- Let \mathbf{r} be the received vector. For decoding we compute the syndrome \mathbf{rH}^T .
- If atmost one error has occurred, the syndrome would be either the zero vector or a column of \mathbf{H} .
- When one error has happened, the number represented by the column of the calculated syndrome is the position in codeword which is in error, and since we considered binary code, it can be corrected.

(Refer Slide Time: 33:22)

Hamming code

- Let \mathbf{r} be the received vector. For decoding we compute the syndrome \mathbf{rH}^T .
- If at most one error has occurred, the syndrome would be either the zero vector or a column of \mathbf{H} .
- When one error has happened, the number represented by the column of the calculated syndrome is the position in codeword which is in error, and since we considered binary code, it can be corrected.


So when single error happens the number represented by the column of the calculated syndrome is the position in the code word where the error has happened. So see it so nice, so by looking at the syndrome and comparing it with the column of the parity check matrix that our reordered parity check matrix we can find out the location where the error has happened. And since for a binary code basically if we can locate the error, so the bit can be either zero and one so whatever the bit is we just flip that bit and get our corrected code word.

(Refer Slide Time: 34:06)

Hamming code

- Let \mathbf{r} be the received vector. For decoding we compute the syndrome \mathbf{rH}^T .
- If atmost one error has occurred, the syndrome would be either the zero vector or a column of \mathbf{H} .
- When one error has happened, the number represented by the column of the calculated syndrome is the position in codeword which is in error, and since we considered binary code, it can be corrected.

(Refer Slide Time: 34:08)



The slide has a dark red header with the text "Hamming code" in white. Below the header, on a light gray background, is a single bullet point. The bullet point text is: "Let 0 1 0 1 0 1 0 be a codeword in [7, 4] Hamming code. Suppose we received the vector 0 0 0 1 0 1 0."

Hamming code

- Let 0 1 0 1 0 1 0 be a codeword in [7, 4] Hamming code. Suppose we received the vector 0 0 0 1 0 1 0.

So let us take an example.

(Refer Slide Time: 34:13)

Hamming code

- Let 0 1 0 1 0 1 0 be a codeword in [7, 4] Hamming code. Suppose we received the vector 0 0 0 1 0 1 0.

So let us say this is a code word for this $[7, 4]$ hamming code. And let us assume that there is a single error has happened and this bit got change from 1 to 0, now given this received sequence I am interested in finding out what is my estimated code word first of all I am interested in finding out whether an error has happened or not, and if the error has happened I am interested in correcting that error.

(Refer Slide Time: 34:48)

Hamming code

- Let 0 1 0 1 0 1 0 be a codeword in [7, 4] Hamming code. Suppose we received the vector 0 0 0 1 0 1 0.
- Syndrome is given by

$$\mathbf{s} = \mathbf{rH}^T = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = (010)$$

So as I said the first thing that I am going to do is.

(Refer Slide Time: 34:51)

Hamming code

- Let 0 1 0 1 0 1 0 be a codeword in [7, 4] Hamming code. Suppose we received the vector 0 0 0 1 0 1 0.
- Syndrome is given by

$$\underline{s} = \underline{rH^T} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = (010)$$

I am going to compute the syndrome so what is my r? This is my r, and what is my H?

(Refer Slide Time: 35:02)

Hamming code

- Let 0 1 0 1 0 1 0 be a codeword in [7, 4] Hamming code. Suppose we received the vector 0 0 0 1 0 1 0.

(Refer Slide Time: 35:04)

Hamming code

- Let \mathbf{r} be the received vector. For decoding we compute the syndrome \mathbf{rH}^T .
- If at most one error has occurred, the syndrome would be either the zero vector or a column of \mathbf{H} .
- When one error has happened, the number represented by the column of the calculated syndrome is the position in codeword which is in error, and since we considered binary code, it can be corrected.

(Refer Slide Time: 35:05)

Hamming code

- Let r be the received vector. For decoding we compute the syndrome rH^T .
- If atmost one error has occurred, the syndrome would be either the zero vector or a column of H .
no error single error

H matrix is basically my parity check matrix.

(Refer Slide Time: 35:06)

Hamming code

- Let \mathbf{r} be the received vector. For decoding we compute the syndrome \mathbf{rH}^T .

(Refer Slide Time: 35:07)

Hamming code

- We can rearrange the columns of the parity check matrix of Hamming code such that column in position i represents the integer i .
- For example for $m = 3$, the Hamming code is of length $n = 2^3 - 1 = 7$, $k = 2^3 - 3 - 1 = 4$, that has parity check matrix \mathbf{H} .

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

1 2 3 4 5 6 7

- Here the column $(x, y, z)^T$ represents the number $x(2^0) + y(2^1) + z(2^2)$.

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

Of the hamming code where the columns are arranged in such a way that column in i^{th} position represents i^{th} integer, so if we take \mathbf{H}^T so we will get 100, 010, 110, 001, 101, 011, 111 right? And that is what I get here.

(Refer Slide Time: 35:40)

Hamming code

- Let \mathbf{r} be the received vector. For decoding we compute the syndrome \mathbf{rH}^T .
- If atmost one error has occurred, the syndrome would be either the zero vector or a column of \mathbf{H} .
no error single error

(Refer Slide Time: 35:41)

Hamming code

- Let \mathbf{r} be the received vector. For decoding we compute the syndrome \mathbf{rH}^T .
- If atmost one error has occurred, the syndrome would be either the zero vector or a column of \mathbf{H} .
- When one error has happened, the number represented by the column of the calculated syndrome is the position in codeword which is in error, and since we considered binary code, it can be corrected.

(Refer Slide Time: 35:42)

Hamming code

- Let 0101010 be a codeword in [7, 4] Hamming code. Suppose we received the vector 0001010.

(Refer Slide Time: 35:43)

Hamming code

- Let 0101010 be a codeword in [7, 4] Hamming code. Suppose we received the vector 0001010.
- Syndrome is given by

$$s = rH^T = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = (010)$$

H^T

$001 + 011 = 010$

So this is my r, this is my H^T now H^T so this is 1, 2, 3 this row will participate this row. So my syndrome will be this plus this, this row plus this row, so this will be $001+011$ which is 010. So my syndrome here is 010 now what do I do? I go back and check which column is 010.

(Refer Slide Time: 36:25)

Hamming code

- Let 0 1 0 1 0 1 0 be a codeword in [7, 4] Hamming code. Suppose we received the vector 0 0 0 1 0 1 0.

So if I go back to my parity check matrix I notice that.

(Refer Slide Time: 36:27)

Hamming code

- We can rearrange the columns of the parity check matrix of Hamming code such that column in position i represents the integer i .
- For example for $m = 3$, the Hamming code is of length $n = 2^3 - 1 = 7$, $k = 2^3 - 3 - 1 = 4$, that has parity check matrix H ,

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

1 2 3 4 5 6 7

- Here the column $(x, y, z)^T$ represents the number $x(2^0) + y(2^1) + z(2^2)$.

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

010 is this second column of my parity check matrix, right? So what does it indicate then? It indicates that the second bit of my code word is in error.

(Refer Slide Time: 36:47)

Hamming code

- Let 0101010 be a codeword in [7, 4] Hamming code. Suppose we received the vector 0001010.
- Syndrome is given by

$$\underline{s} = \underline{rH^T} = \left[\begin{array}{ccccccc} 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{array} \right] \begin{array}{c} \left[\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{array} \right] \\ \underline{H^T} \end{array} = (010)$$

$001 + 011 = 010$

So then I can go ahead and correct this error.

(Refer Slide Time: 36:50)

Hamming code

- Let 0 1 0 1 0 1 0 be a codeword in [7, 4] Hamming code. Suppose we received the vector 0 0 0 1 0 1 0.
- Syndrome is given by

$$\mathbf{s} = \mathbf{rH}^T = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = (0 \ 1 \ 0)$$

- The number represented by syndrome is 2, hence the error is in second bit position. Hence estimated codeword is 0 1 0 1 0 1 0.

(Refer Slide Time: 36:52)

Hamming code

- Let 0101010 be a codeword in [7, 4] Hamming code. Suppose we received the vector 0001010.
- Syndrome is given by $\hat{V} = 0101010$

$$s = rH^T = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \underline{(010)}$$

- The number represented by syndrome is 2, hence the error is in second bit position. Hence estimated codeword is 0101010.

So this second bit, from the syndrome I can find out that this syndrome is nothing but the column in the second location. So my second bit is in error and since we are talking about binary code, so then this zero should have been one. So my estimated code word is then 0101010 and you can see that is what I had transmitted, this is the same code word that I have transmitted, you can yourself verify change some other bit location, introduce some other single error and you can -- you will see from the syndrome you can find out the location of your error.

(Refer Slide Time: 37:40)

Shortened Hamming code

- If we delete any l columns from the parity check matrix of a Hamming code, we get shortened Hamming code with following parameters

Code length:	$n = 2^m - l - 1$
Information bits:	$k = 2^m - m - l - 1$
Parity bits:	$n - k = m$
Minimum distance:	$d_{\min} \geq 3$

So it has a very simple decoding algorithm, now as I said before in case of hamming codes, once I fix my m , my code parameters are fixed.

(Refer Slide Time: 37:57)

Shortened Hamming code

- If we delete any l columns from the parity check matrix of a Hamming code, we get shortened Hamming code with following parameters

Code length:	$n = 2^m - l - 1$
Information bits:	$k = 2^m - m - l - 1$
Parity bits:	$n - k = m$
Minimum distance:	$d_{\min} \geq 3$

So for example the examples that we were dealing with, we had taken $m=3$.

(Refer Slide Time: 37:59)

Shortened Hamming code

- If we delete any l columns from the parity check matrix of a Hamming code, we get shortened Hamming code with following parameters

Code length:	$n = 2^m - l - 1$	$m = 3$
Information bits:	$k = 2^m - m - l - 1$	$n = 2^3 - 1 = 7$
Parity bits:	$n - k = m$	$k = 4$
Minimum distance:	$d_{\min} \geq 3$	$(7, 4)$
		$(8, 4)$
		$(7, 3)$
		$(6, 3)$

So once the moment I fix $m=3$ my n is fixed it is 2^{3-1} that is 7, so k is $7-3$ that is 4. So this becomes the $(7, 4)$ code, right? What if I am interested in designing let us say $(8, 4)$ code or let us say I am interested in $(7, 3)$ codes, $(6, 3)$ code. So there are various techniques available to twig the parameters of the code, we are going to talk about some of them, we are not exhaustively covering all possible ways of lengthening or shortening a code.

We just give you just few examples to illustrate this idea of changing the parameters of the code. So the first thing that we are considering is a shortened hamming code, so how do you get a shortened hamming code? You delete l columns from the parity check matrix of your hamming code, now if I delete l columns that means my code word length is decreased by l .

(Refer Slide Time: 39:17)

Shortened Hamming code

- If we delete any l columns from the parity check matrix of a Hamming code, we get shortened Hamming code with following parameters

Code length:	$n = 2^m - l - 1$	$m = 3$
Information bits:	$k = 2^m - m - l - 1$	$n = 2^3 - 1 = 7$
Parity bits:	$n - k = m$	$k = 4$
Minimum distance:	$d_{\min} \geq 3$	$(7, 4)$

$(8, 4)$
 $(7, 3)$
 $(6, 3)$

So my new code word length is then $2^m - 1 - l$ so this is my code word length, now numbers of rows are still the same that is m , so number of parity bits are still same but since I deleted l coded bits so number of information bits also got decreased by l . Now minimum distance of a hamming code is three if I am deleting some columns.

(Refer Slide Time: 39:54)

Shortened Hamming code

- If we delete any l columns from the parity check matrix of a Hamming code, we get shortened Hamming code with following parameters

Code length:	$n = 2^m - l - 1$	$m = 3$ $n = 2^3 - 1 = 7$
Information bits:	$k = 2^m - m - l - 1$	$k = 4$
Parity bits:	$n - k = m$	$(7, 4)$
Minimum distance:	$d_{\min} \geq 3$	$(8, 4)$ $(7, 3)$ $(6, 3)$

$H = [a_1, a_2, \textcircled{a_3}, \textcircled{a_4}, \dots, a_n]$

Then minimum distance cannot be less than what was the original minimum distance of the code, why? Because you can see if I have a parity check matrix here and let us say I have these columns let us call it a_1, a_2, a_n . Now if I am removing some of the columns I can carefully remove those columns which are causing the sum of let say d columns to be linearly dependent. So if I remove some of those columns I can possibly increase my minimum distance.

(Refer Slide Time: 40:37)

Shortened Hamming code

- If we delete any l columns from the parity check matrix of a Hamming code, we get shortened Hamming code with following parameters

Code length:	$n = 2^m - l - 1$	$m = 3$
Information bits:	$k = 2^m - m - l - 1$	$n = 2^3 - 1 = 7$
Parity bits:	$n - k = m$	$k = 4$
Minimum distance:	$d_{\min} \geq 3$	$(7, 4)$

$H = [a_1, a_2, \textcircled{3}, \textcircled{4}, a_n]$

$(8, 4)$
 $(7, 3)$
 $(6, 3)$

And that is why I wrote by whenever you delete columns on the parity check matrix the minimum distance is atleast what was there in the original code.

(Refer Slide Time: 40:48)

Shortened Hamming code

- If we delete any l columns from the parity check matrix of a Hamming code, we get shortened Hamming code with following parameters:
Code length: $n = 2^m - l - 1$
Information bits: $k = 2^m - m - l - 1$
Parity bits: $n - k = m$
Minimum distance: $d_{\min} \geq 3$
- H matrix of (7,4) Hamming code given by
$$H = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$
- Shortened (6,3) Hamming code has a parity check matrix
$$H_1 = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \quad d = 3$$

This is one example of a (7, 4) hamming code, so if I delete let say I delete this if I delete this column what do I get, I get a shortened (6, 3) hamming code. So when I delete this my n becomes six from seven and my information bits also reduce from four to three. This code also has minimum distance three you can see three columns will add up to zero. Let us say this column, this column, and this column add up to zero.

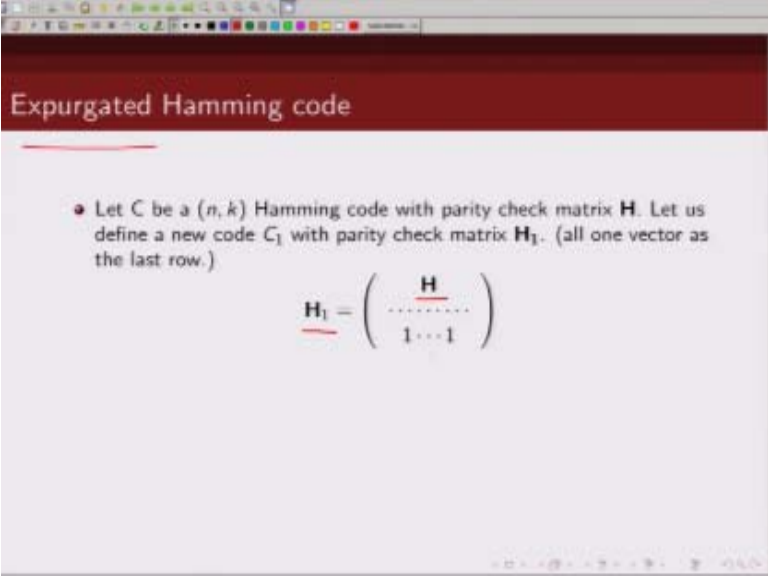
So, minimum distance is still three for this code. So whenever you want a shortened code make sure you remove those columns in a judicial way so that potentially you can increase the minimum distance. In this particular example, if I delete one column I cannot increase the hamming distance, but if I delete more columns from this parity check matrix I can potentially increase the minimum distance of the code.

(Refer Slide Time: 42:02)

Shortened Hamming code

- If we delete any l columns from the parity check matrix of a Hamming code, we get shortened Hamming code with following parameters
 - Code length: $n = 2^m - l - 1$
 - Information bits: $k = 2^m - m - l - 1$
 - Parity bits: $n - k = m$
 - Minimum distance: $d_{\min} \geq 3$
- H matrix of (7,4) Hamming code given by
$$H = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix}$$
- Shortened (6,3) Hamming code has a parity check matrix
$$H_1 = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}$$
 $d = 3$

(Refer Slide Time: 42:02)



The slide has a dark red header with the title "Expurgated Hamming code". Below the header, a bullet point states: "Let C be a (n, k) Hamming code with parity check matrix H . Let us define a new code C_1 with parity check matrix H_1 . (all one vector as the last row.)". Below the text, the matrix H_1 is defined as:

$$H_1 = \begin{pmatrix} H \\ 1 \dots 1 \end{pmatrix}$$

Another class of code is what is called expurgated codes. So what is expurgated hamming code? So if H is the parity check matrix of hamming code then I define a new parity check matrix which has all ones as the last row.

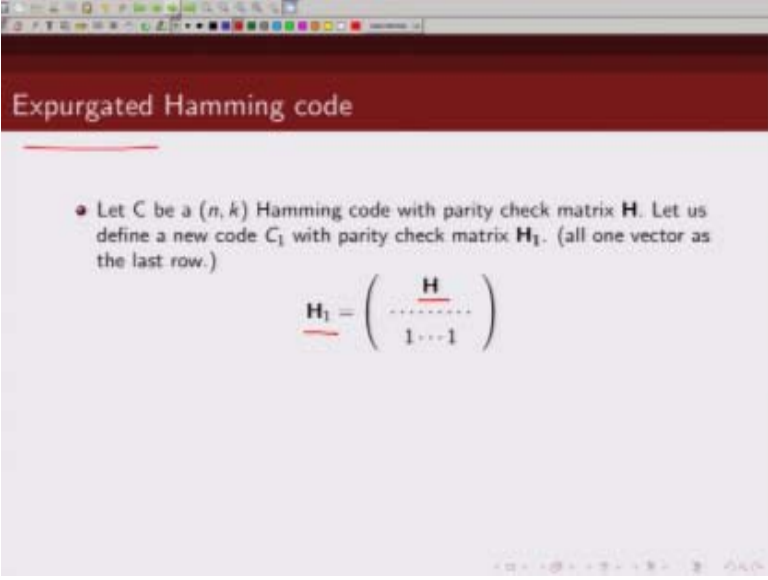
(Refer Slide Time: 42:28)

Shortened Hamming code

- If we delete any l columns from the parity check matrix of a Hamming code, we get shortened Hamming code with following parameters
 - Code length: $n = 2^m - l - 1$
 - Information bits: $k = 2^m - m - l - 1$
 - Parity bits: $n - k = m$
 - Minimum distance: $d_{\min} \geq 3$
- H matrix of (7,4) Hamming code given by
$$H = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix}$$
- Shortened (6,3) Hamming code has a parity check matrix
$$H_1 = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}$$
 $d = 3$

Now note the parity check matrix of a hamming code does not have all ones. So when I add an all one row.

(Refer Slide Time: 42:37)



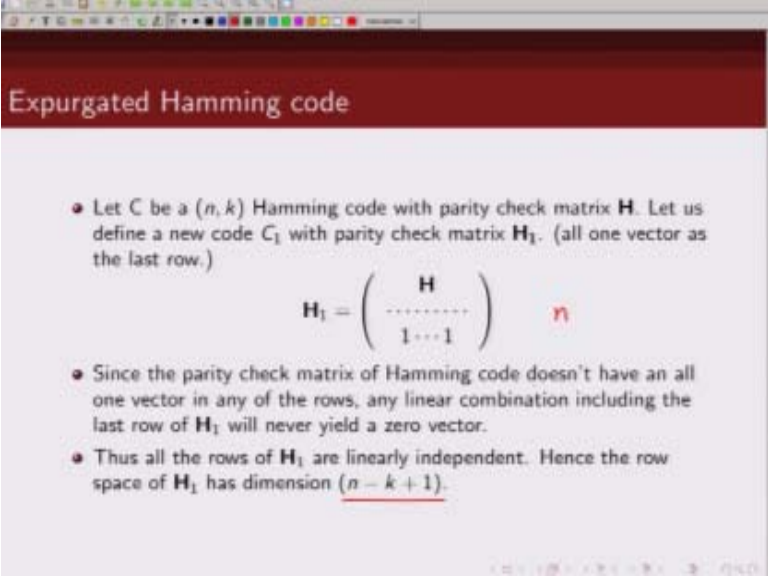
The slide has a dark red header with the title "Expurgated Hamming code". Below the header, a bullet point states: "Let C be a (n, k) Hamming code with parity check matrix H . Let us define a new code C_1 with parity check matrix H_1 . (all one vector as the last row.)". Below the text, the matrix H_1 is defined as:

$$H_1 = \begin{pmatrix} H \\ \text{all one vector} \end{pmatrix}$$

The matrix is shown with the original matrix H in the top row and a new row of all ones at the bottom. The H in the top row is underlined in the original image.

Essentially the rank of this matrix will be one more than the rank of the original matrix H .

(Refer Slide Time: 42:47)



Expurgated Hamming code

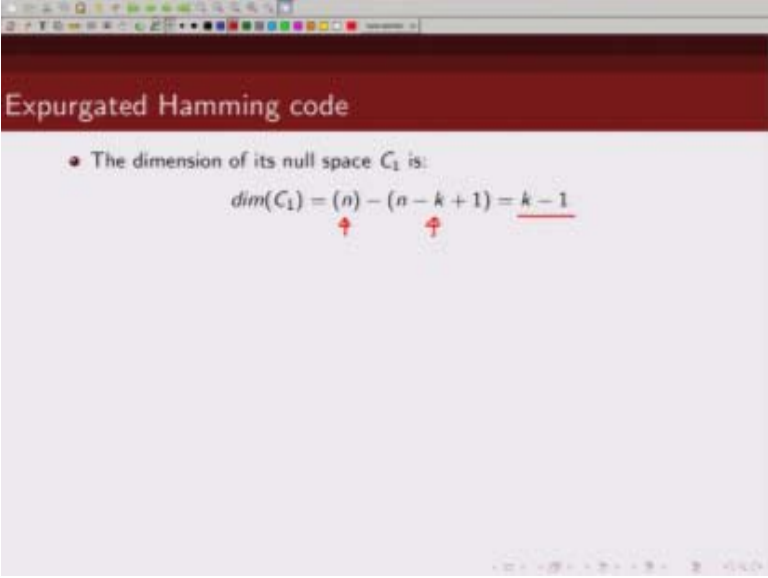
- Let C be a (n, k) Hamming code with parity check matrix H . Let us define a new code C_1 with parity check matrix H_1 . (all one vector as the last row.)

$$H_1 = \begin{pmatrix} H \\ \dots\dots\dots \\ 1 \dots 1 \end{pmatrix} \quad n$$

- Since the parity check matrix of Hamming code doesn't have an all one vector in any of the rows, any linear combination including the last row of H_1 will never yield a zero vector.
- Thus all the rows of H_1 are linearly independent. Hence the row space of H_1 has dimension $(n - k + 1)$.

Because any linear combination of last row with the rows of the earlier parity check matrix H would not be dependent. Hence the rank of this new matrix H_1 is $(n-k+1)$ and what is the number of columns of this matrix that is n . So rank of this matrix is $(n-k+1)$.

(Refer Slide Time: 43:20)



Expurgated Hamming code

- The dimension of its null space C_1 is:

$$\dim(C_1) = (n) - (n - k + 1) = \underline{k - 1}$$

Then what is the dimension of the null space of this or what is the dimension of code sequence here, so that is basically this is n minus dimension of H matrix so that becomes $k-1$.

(Refer Slide Time: 43:37)

Expurgated Hamming code

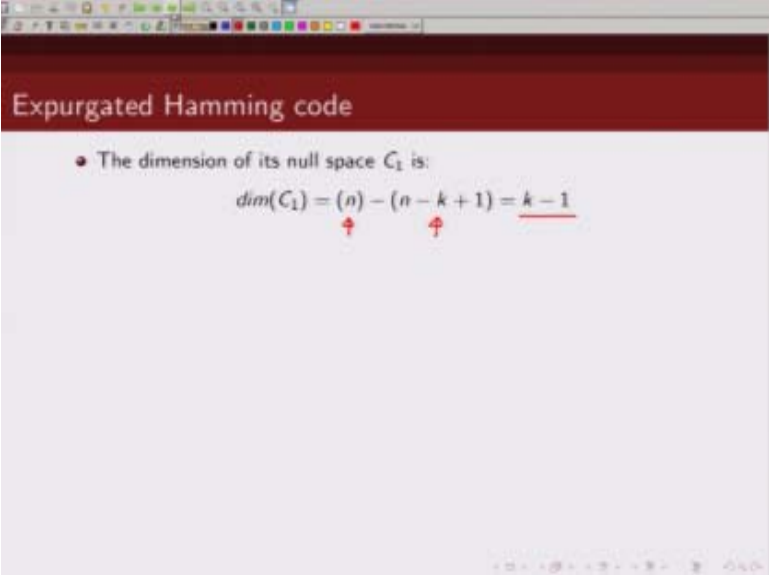
- Let C be a (n, k) Hamming code with parity check matrix H . Let us define a new code C_1 with parity check matrix H_1 . (all one vector as the last row.)

$$H_1 = \begin{pmatrix} H \\ \cdots \cdots \cdots \\ 1 \cdots 1 \end{pmatrix} \quad n$$

- Since the parity check matrix of Hamming code doesn't have an all one vector in any of the rows, any linear combination including the last row of H_1 will never yield a zero vector.
- Thus all the rows of H_1 are linearly independent. Hence the row space of H_1 has dimension $(n - k + 1)$.

So the new code that we derive by adding an all one row in the parity check matrix of the hamming code.

(Refer Slide Time: 43:44)



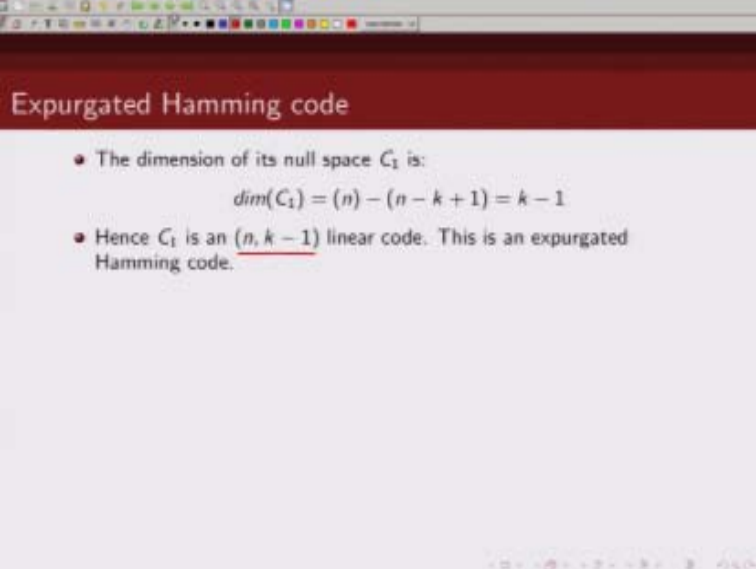
Expurgated Hamming code

- The dimension of its null space C_1 is:

$$\dim(C_1) = (n) - (n - k + 1) = \underline{k - 1}$$

We get a new code which is a $(n, k-1)$ code.

(Refer Slide Time: 43:46)

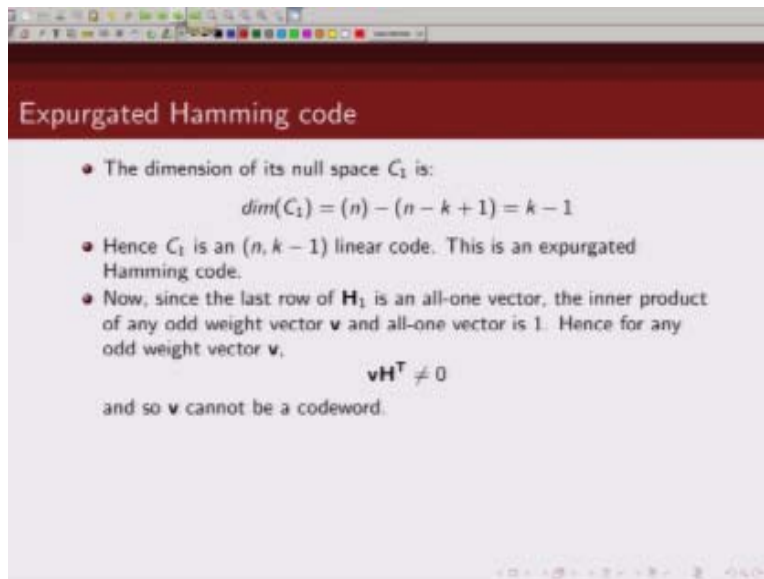


Expurgated Hamming code

- The dimension of its null space C_1 is:
$$\dim(C_1) = (n) - (n - k + 1) = k - 1$$
- Hence C_1 is an $(n, k - 1)$ linear code. This is an expurgated Hamming code.

And this is known as expurgated Hamming code.

(Refer Slide Time: 43:55)

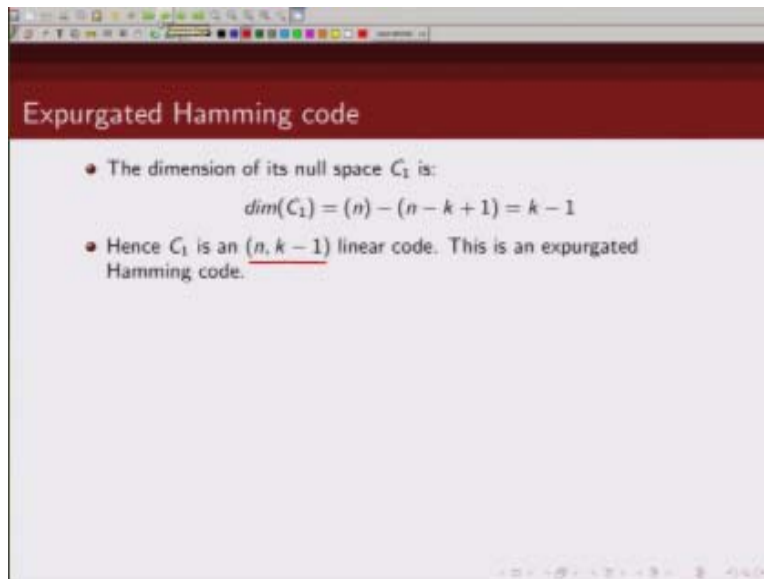


Expurgated Hamming code

- The dimension of its null space C_1 is:
$$\dim(C_1) = (n) - (n - k + 1) = k - 1$$
- Hence C_1 is an $(n, k - 1)$ linear code. This is an expurgated Hamming code.
- Now, since the last row of H_1 is an all-one vector, the inner product of any odd weight vector \mathbf{v} and all-one vector is 1. Hence for any odd weight vector \mathbf{v} ,
$$\mathbf{vH}^T \neq 0$$
and so \mathbf{v} cannot be a codeword.

Now one of the interesting property of this code is, this code contains all even weight code word it is not very difficult to prove.

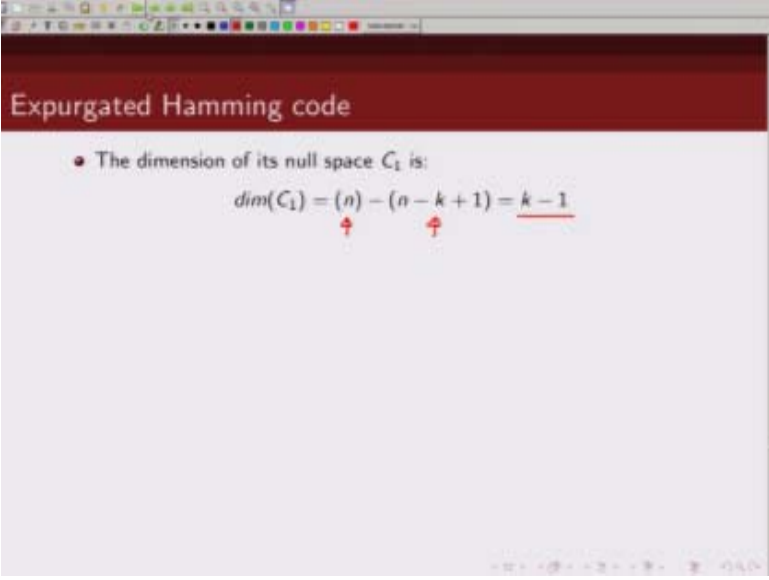
(Refer Slide Time: 44:06)



Expurgated Hamming code

- The dimension of its null space C_1 is:
$$\dim(C_1) = (n) - (n - k + 1) = k - 1$$
- Hence C_1 is an $(n, k - 1)$ linear code. This is an expurgated Hamming code.

(Refer Slide Time: 44:08)



Expurgated Hamming code

- The dimension of its null space C_1 is:

$$\dim(C_1) = (n) - (n - k + 1) = \underline{k - 1}$$

If v is a valid code word.

(Refer Slide Time: 44:10)

Expurgated Hamming code

- Let C be a (n, k) Hamming code with parity check matrix H . Let us define a new code C_1 with parity check matrix H_1 . (all one vector as the last row.)

$$H_1 = \begin{pmatrix} H \\ \boxed{1 \cdots 1} \end{pmatrix} \quad n \quad vH_1^T = 0$$

- Since the parity check matrix of Hamming code doesn't have an all one vector in any of the rows, any linear combination including the last row of H_1 will never yield a zero vector.
- Thus all the rows of H_1 are linearly independent. Hence the row space of H_1 has dimension $(n - k + 1)$.

Then we know vH^T should be zero. And last row of H_1 has all ones, so when we take transpose so the first column will be all ones, and when we do vH_1^T what we will get is let us say the components of v are v_0, v_1, v_2 so v_{n-1} then what you will get is basically.

(Refer Slide Time: 44:42)

Expurgated Hamming code

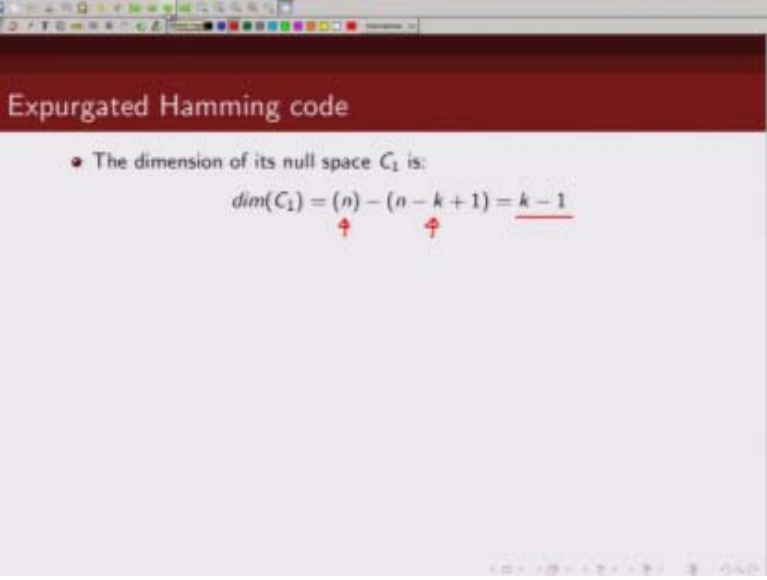
- Let C be a (n, k) Hamming code with parity check matrix H . Let us define a new code C_1 with parity check matrix H_1 . (all one vector as the last row.)

$$H_1 = \begin{pmatrix} H \\ \boxed{1 \cdots 1} \end{pmatrix} \quad n \quad vH_1^T = 0$$

- Since the parity check matrix of Hamming code doesn't have an all one vector in any of the rows, any linear combination including the last row of H_1 will never yield a zero vector.
- Thus all the rows of H_1 are linearly independent. Hence the row space of H_1 has dimension $(n - k + 1)$.

Sum of these components of this code.

(Refer Slide Time: 44:45)



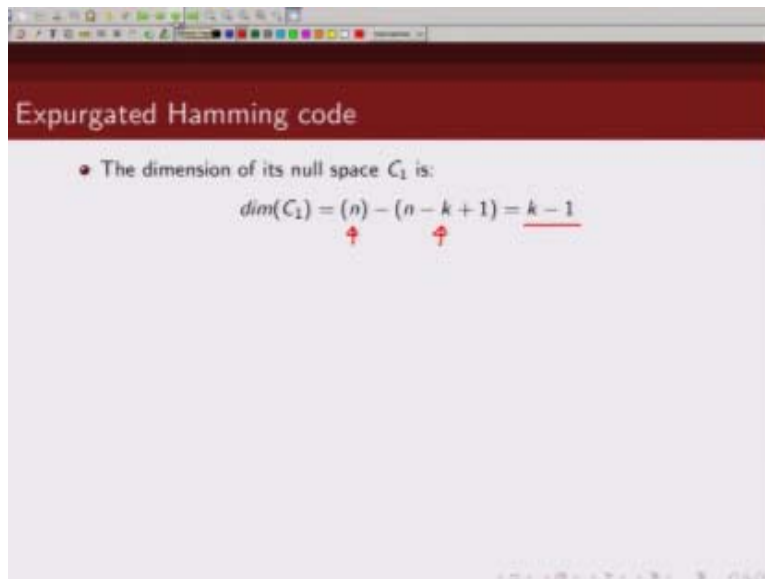
Expurgated Hamming code

- The dimension of its null space C_1 is:

$$\dim(C_1) = (n) - (n - k + 1) = \underline{k - 1}$$

Code word we should add up to zero. And this will happen only when v has even weight.

(Refer Slide Time: 44:54)



The slide features a dark red header with the title "Expurgated Hamming code" in white. Below the header, a bullet point states: "• The dimension of its null space C_1 is:". This is followed by the equation
$$\dim(C_1) = (n) - (n - k + 1) = \underline{k - 1}$$
 where the n in the first term and the n in the second term are each marked with a red upward-pointing arrow. The result $k - 1$ is underlined in red. The slide is presented in a window with a standard toolbar at the top and navigation controls at the bottom.

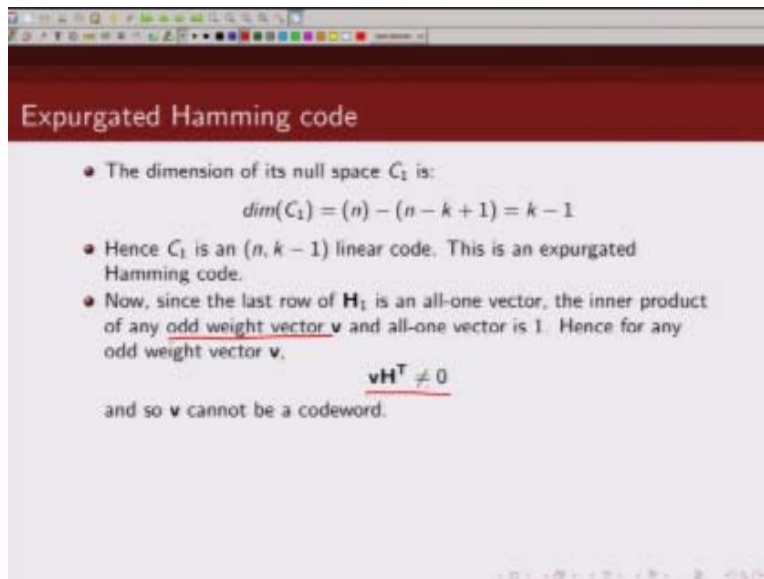
(Refer Slide Time: 44:54)

Expurgated Hamming code

- The dimension of its null space C_1 is:
$$\dim(C_1) = (n) - (n - k + 1) = k - 1$$
- Hence C_1 is an $(n, k - 1)$ linear code. This is an expurgated Hamming code.

So this new code that we generated expurgated code is basically has all even weight code words.

(Refer Slide Time: 45:03)



Expurgated Hamming code

- The dimension of its null space C_1 is:
$$\dim(C_1) = (n) - (n - k + 1) = k - 1$$
- Hence C_1 is an $(n, k - 1)$ linear code. This is an expurgated Hamming code.
- Now, since the last row of H_1 is an all-one vector, the inner product of any odd weight vector \mathbf{v} and all-one vector is 1. Hence for any odd weight vector \mathbf{v} ,
$$\mathbf{vH}^T \neq 0$$
and so \mathbf{v} cannot be a codeword.

So if you have an odd weight vector then \mathbf{vH}^T cannot be 0. Because of this all ones in the parity check matrix.

(Refer Slide Time: 45:19)

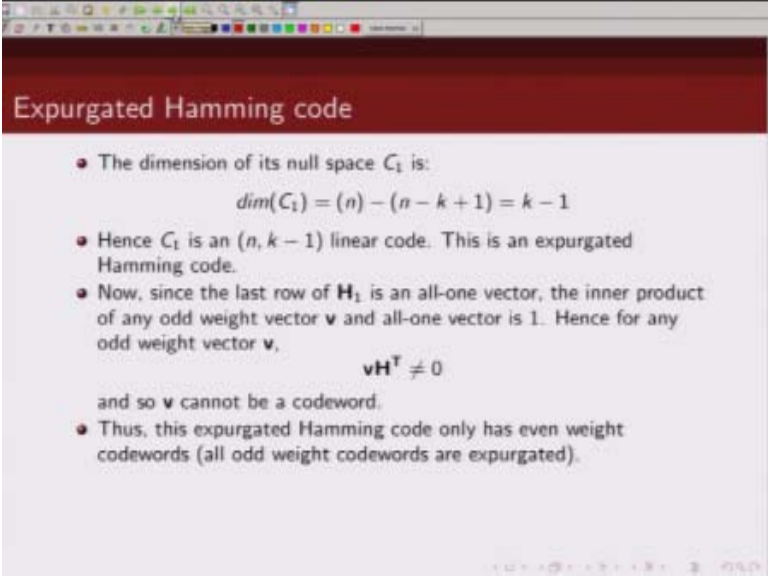
Expurgated Hamming code

- The dimension of its null space C_1 is:
$$\dim(C_1) = (n) - (n - k + 1) = k - 1$$
- Hence C_1 is an $(n, k - 1)$ linear code. This is an expurgated Hamming code.
- Now, since the last row of H_1 is an all-one vector, the inner product of any odd weight vector \mathbf{v} and all-one vector is 1. Hence for any odd weight vector \mathbf{v} ,
$$\mathbf{vH}_1^T \neq 0$$

and so \mathbf{v} cannot be a codeword.

Of this new matrix H_1 expurgated code parity check matrix okay. So \mathbf{vH}_1^T cannot be zero, if \mathbf{v} has odd weight.

(Refer Slide Time: 45:35)

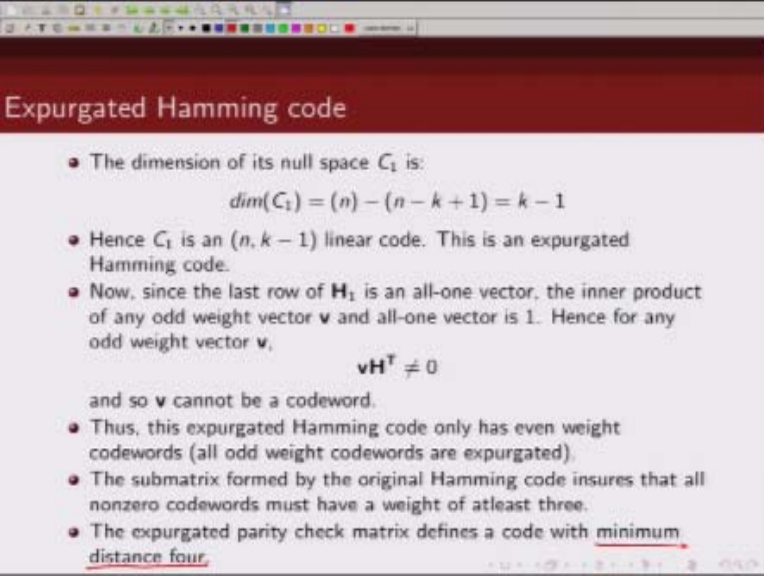


Expurgated Hamming code

- The dimension of its null space C_1 is:
$$\dim(C_1) = (n) - (n - k + 1) = k - 1$$
- Hence C_1 is an $(n, k - 1)$ linear code. This is an expurgated Hamming code.
- Now, since the last row of H_1 is an all-one vector, the inner product of any odd weight vector \mathbf{v} and all-one vector is 1. Hence for any odd weight vector \mathbf{v} ,
$$\mathbf{vH}^T \neq 0$$
and so \mathbf{v} cannot be a codeword.
- Thus, this expurgated Hamming code only has even weight codewords (all odd weight codewords are expurgated).

So you can see that, this is one way of getting rid of odd code words. So by adding an additional all one rows in the parity check matrix of the original code we actually got rid of all odd weight code words. So the minimum distance of this expurgated hamming code would be then four why? Original code has minimum distance three, but now we got rid of all odd weight code words. So the minimum distance of this new code expurgated Hamming code would be four.

(Refer Slide Time: 46:16)

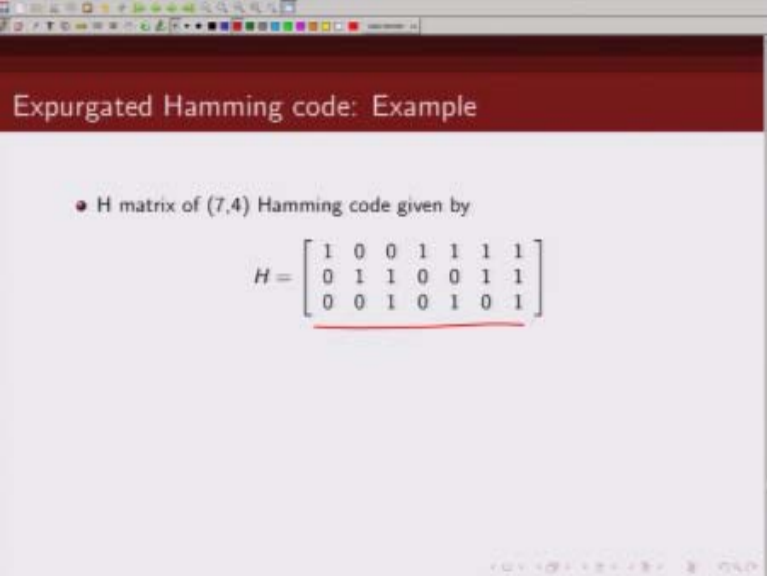


Expurgated Hamming code

- The dimension of its null space C_1 is:
$$\dim(C_1) = (n) - (n - k + 1) = k - 1$$
- Hence C_1 is an $(n, k - 1)$ linear code. This is an expurgated Hamming code.
- Now, since the last row of H_1 is an all-one vector, the inner product of any odd weight vector \mathbf{v} and all-one vector is 1. Hence for any odd weight vector \mathbf{v} ,
$$\mathbf{vH}^T \neq 0$$
and so \mathbf{v} cannot be a codeword.
- Thus, this expurgated Hamming code only has even weight codewords (all odd weight codewords are expurgated).
- The submatrix formed by the original Hamming code insures that all nonzero codewords must have a weight of atleast three.
- The expurgated parity check matrix defines a code with minimum distance four.

So minimum distance is four.

(Refer Slide Time: 46:23)



The slide has a dark red header with the title "Expurgated Hamming code: Example". Below the header, there is a bullet point that reads "H matrix of (7,4) Hamming code given by". Underneath the bullet point, the matrix H is displayed as a 3x7 grid of numbers, enclosed in large square brackets. The matrix is:

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

The bottom row of the matrix is underlined with a red line.

So this is an example, this is the parity check matrix of the original Hamming code.

(Refer Slide Time: 46:30)

Expurgated Hamming code: Example

- H matrix of (7,4) Hamming code given by

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

- Distance-4 expurgated Hamming code has a parity check matrix H_1 given by

$$H_1 = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

And if we add all one rows in the parity check matrix we get the parity check matrix of a expurgated code.

(Refer Slide Time: 46:39)

Extended Hamming code

- Let C be a (n, k) Hamming code with parity check matrix H . Let us define a new code C_1 with parity check matrix H_1 . (all one vector as the last row.)

$$H_1 = \left(\begin{array}{c|c} H & \begin{matrix} 0 \\ \vdots \\ 0 \end{matrix} \\ \hline \begin{matrix} 1 & \dots & 1 \end{matrix} & 1 \end{array} \right)$$

H_{n-k}
 $H_1 = n-k+1$

Finally I will conclude this lecture with another class which is basically extension which called extension of a code. So this is an example of an extended hamming code. So how do I generate an extended Hamming code. So note, so this is the parity check matrix of my original Hamming code. I add a zero so I add an additional column which is zero here. So these are all zero here, and then I add one row which is all one.

And it is not very difficult to see that the rank of the matrix will be if the rank of H is let say $n-k$. the rank of this matrix H_1 will be $n-k+1$. Because the parity check matrix of the Hamming code did not have an all one row.

(Refer Slide Time: 47:36)

Extended Hamming code

- Let C be a (n, k) Hamming code with parity check matrix H . Let us define a new code C_1 with parity check matrix H_1 . (all one vector as the last row.)

$$H_1 = \left(\begin{array}{c|c} H & \begin{matrix} 0 \\ \vdots \\ 0 \\ 1 \end{matrix} \end{array} \right)$$

H_{n-k}
 $H_1 = n-k+1$

And then these are all zeros and this is one so any linear combination of this row with these rows basically would not, it would not change the it will not decrease the Hamming distance essentially.

(Refer Slide Time: 47:49)

Extended Hamming code

- Let C be a (n, k) Hamming code with parity check matrix H . Let us define a new code C_1 with parity check matrix H_1 . (all one vector as the last row.)

$$H_1 = \left(\begin{array}{c|c} H & 0 \\ \hline 1 \dots 1 & 1 \end{array} \right) \quad \begin{array}{l} n \text{ columns} \downarrow 1 \\ n+1 \end{array}$$

- Since the parity check matrix of Hamming code doesn't have an all one vector in any of the rows, any linear combination including the last row of H_1 will never yield a zero vector.
- Thus all the rows of H_1 are linearly independent. Hence the row space of H_1 has dimension $(n - k + 1)$.

So this will have the rank of this matrix would be one more than the rank of the original parity check matrix H . So what we are saying then is all rows of these parity check matrix H_1 are linearly independent. As I said that is because the original H matrix did not have all ones here. And this is one here, this is zero here, so if we take linear combinations of this with any of the rows of the original parity check matrix they would not be linearly dependent.

So all rows are linearly independent hence, the rank of this matrix is $(n-k+1)$. And what is the number of columns the original matrix had n columns, and we added one column, one more column. So number of columns here is $n+1$, so this will define a code of length $n+1$.

(Refer Slide Time: 48:53)

Extended Hamming code

- The dimension of its null space C_1 is:

$$\dim(C_1) = \underbrace{(n+1)}_1 - \underbrace{(n-k+1)}_1 = \underbrace{k}_1$$

So then we can find out what is the dimension of null space or so this is number of coded bits, this is the rank of the new parity check matrix. So number of coded -- so the dimension of null space is k so this will then generate.

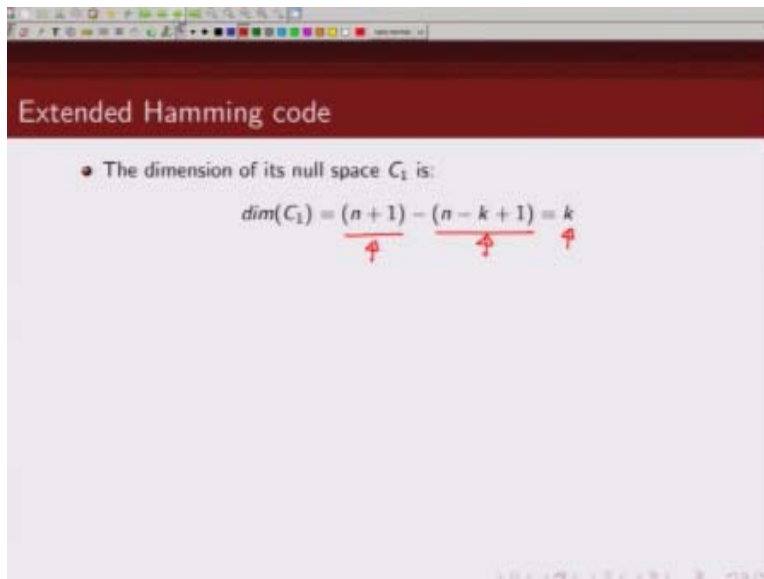
(Refer Slide Time: 49:11)

Extended Hamming code

- The dimension of its null space C_1 is:
$$\dim(C_1) = (n+1) - (n-k+1) = k$$
- Hence C_1 is an $(n+1, k)$ linear code. This is an extended Hamming code.

An $(n+1, k)$ linear block code, and this is known as extension code or extended Hamming code.

(Refer Slide Time: 49:21)



Extended Hamming code

- The dimension of its null space C_1 is:

$$\dim(C_1) = \underbrace{(n+1)}_{\uparrow} - \underbrace{(n-k+1)}_{\uparrow} = \underbrace{k}_{\uparrow}$$

(Refer Slide Time: 49:22)

Extended Hamming code

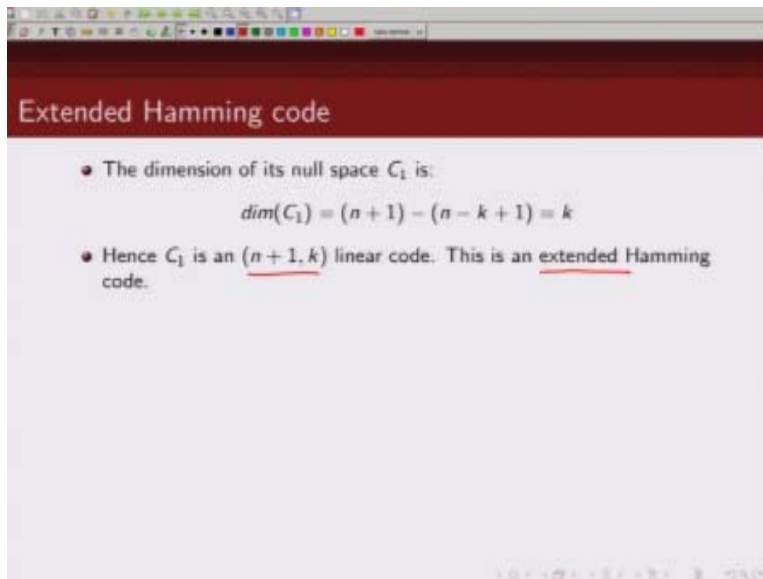
- Let C be a (n, k) Hamming code with parity check matrix H . Let us define a new code C_1 with parity check matrix H_1 . (all one vector as the last row.)

$$\underline{H_1} = \left(\begin{array}{c|c} \begin{array}{c} \text{--- } n \text{ columns ---} \\ H \\ \hline 1 \dots 1 \end{array} & \begin{array}{c} \downarrow 1 \\ 0 \\ \dots \\ 1 \end{array} \end{array} \right) \quad n+1$$

- Since the parity check matrix of Hamming code doesn't have an all one vector in any of the rows, any linear combination including the last row of H_1 will never yield a zero vector.
- Thus all the rows of H_1 are linearly independent. Hence the row space of H_1 has dimension $(n - k + 1)$.

If this H is a matrix a parity check matrix with hamming code the code described by H_1 which is given by this would be extended hamming code.

(Refer Slide Time: 49:34)



Extended Hamming code

- The dimension of its null space C_1 is:
$$\dim(C_1) = (n+1) - (n-k+1) = k$$
- Hence C_1 is an $(n+1, k)$ linear code. This is an extended Hamming code.

(Refer Slide Time: 49:35)

Extended Hamming code

- The dimension of its null space C_1 is:
$$\dim(C_1) = (n+1) - (n-k+1) = k$$
- Hence C_1 is an $(n+1, k)$ linear code. This is an extended Hamming code.
- Now, since the last row of \mathbf{H}_1 is an all-one vector, the inner product of any odd weight vector \mathbf{v} and all-one vector is 1. Hence for any odd weight vector \mathbf{v} ,
$$\mathbf{v}\mathbf{H}^T \neq 0$$

and so \mathbf{v} cannot be a codeword.

And again we can see that the extended hamming code will have only even code words that is because the last row of the parity check matrix contains all ones. So $\mathbf{v}\mathbf{H}^T$ cannot be zero.

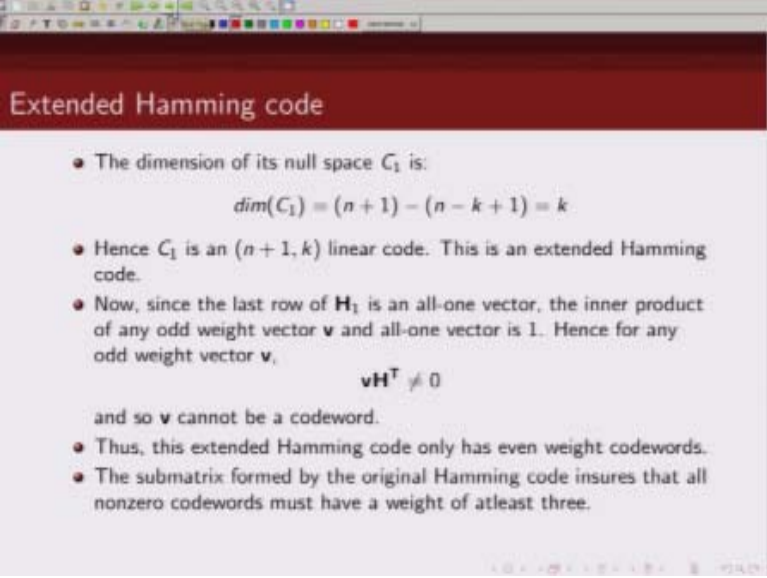
(Refer Slide Time: 49:53)

Extended Hamming code

- The dimension of its null space C_1 is:
$$\dim(C_1) = (n+1) - (n-k+1) = k$$
- Hence C_1 is an $(n+1, k)$ linear code. This is an extended Hamming code.
- Now, since the last row of \mathbf{H}_1 is an all-one vector, the inner product of any odd weight vector \mathbf{v} and all-one vector is 1. Hence for any odd weight vector \mathbf{v} ,
$$\mathbf{v}\mathbf{H}^T \neq 0$$
and so \mathbf{v} cannot be a codeword.

If \mathbf{v} has odd weight.

(Refer Slide Time: 49:56)

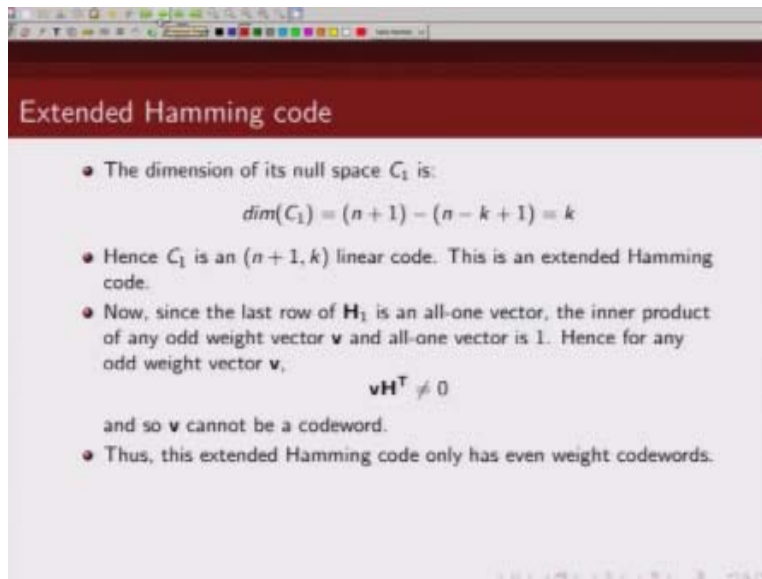


Extended Hamming code

- The dimension of its null space C_1 is:
$$\dim(C_1) = (n+1) - (n-k+1) = k$$
- Hence C_1 is an $(n+1, k)$ linear code. This is an extended Hamming code.
- Now, since the last row of \mathbf{H}_1 is an all-one vector, the inner product of any odd weight vector \mathbf{v} and all-one vector is 1. Hence for any odd weight vector \mathbf{v} ,
$$\mathbf{v}\mathbf{H}^T \neq 0$$
and so \mathbf{v} cannot be a codeword.
- Thus, this extended Hamming code only has even weight codewords.
- The submatrix formed by the original Hamming code insures that all nonzero codewords must have a weight of atleast three.

So the minimum distance of extended hamming code is four. Now please note the some of the techniques that we mentioned here extension, shortening, expurgated that is valid for any other linear block codes too.

(Refer Slide Time: 50:13)



Extended Hamming code

- The dimension of its null space C_1 is:
$$\dim(C_1) = (n+1) - (n-k+1) = k$$
- Hence C_1 is an $(n+1, k)$ linear code. This is an extended Hamming code.
- Now, since the last row of H_1 is an all-one vector, the inner product of any odd weight vector \mathbf{v} and all-one vector is 1. Hence for any odd weight vector \mathbf{v} ,
$$\mathbf{vH}^T \neq 0$$
and so \mathbf{v} cannot be a codeword.
- Thus, this extended Hamming code only has even weight codewords.

(Refer Slide Time: 50:15)

Extended Hamming code

- The dimension of its null space C_1 is:
$$\dim(C_1) = (n+1) - (n-k+1) = k$$
- Hence C_1 is an $(n+1, k)$ linear code. This is an extended Hamming code.
- Now, since the last row of H_1 is an all-one vector, the inner product of any odd weight vector \mathbf{v} and all-one vector is 1. Hence for any odd weight vector \mathbf{v} ,
$$\mathbf{vH}^T \neq 0$$

and so \mathbf{v} cannot be a codeword.

And these are some of the ways in which we can change the code parameters. So with this we conclude this lecture. Thank you.

Acknowledgement

Ministry of Human Resource & Development

Prof. Satyaki Roy

Co-ordinator, NPTEL IIT Kanpur

NPTEL Team

Sanjay Pal

Ashish Singh

Badal Pradhan

Tapobrata Das

Ram Chandra

Dilip Tripathi

Manoj Shrivastava

Padam Shukla

Sanjay Mishra

Shubham Rawat

Shikha Gupta

K. K. Mishra

Aradhana Singh

Sweta

**Ashutosh Gairola
Dilip Katiyar
Sharwan
Hari Ram
Bhadra Rao
Puneet Kumar Bajpai
Lalty Dutta
Ajay Kanaujia
Shivendra Kumar Tiwari**

an IIT Kanpur Production

©copyright reserved