

Information Theory, Coding and Cryptography
Dr. Ranjan Bose
Department of Electrical Engineering
Indian Institute of Technology, Delhi

Module – 15
Linear Block Codes
Lecture – 15

(Refer Slide Time: 00:33)

Information Theory, Coding and Cryptography

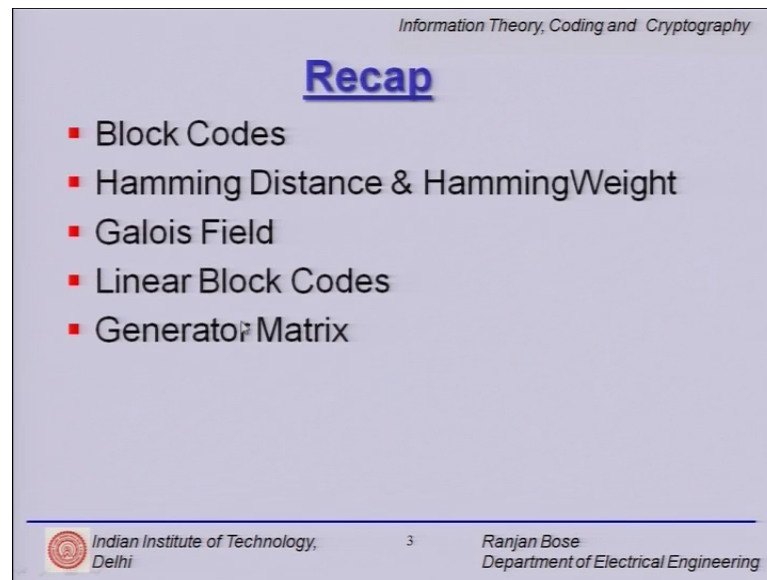
Outline

- Linear Block Codes
- Equivalent Codes
- Generator Matrix
- Parity Check Matrix

Indian Institute of Technology, Delhi 2 Ranjan Bose
Department of Electrical Engineering

Hello, and welcome to next module on Linear Block Codes. Let us start with a brief outline for today's talk. Today we will look at linear block codes. And then, we will define what are equivalent codes. We will pose ourselves the question about efficient generation of linear block codes. And then, we will look at the notion of a generator matrix. Finally, we will enter the decoding side of things, and look at parity check matrix. So, this is the general outline for today's talk.

(Refer Slide Time: 01:06)



Information Theory, Coding and Cryptography

Recap

- Block Codes
- Hamming Distance & Hamming Weight
- Galois Field
- Linear Block Codes
- Generator Matrix

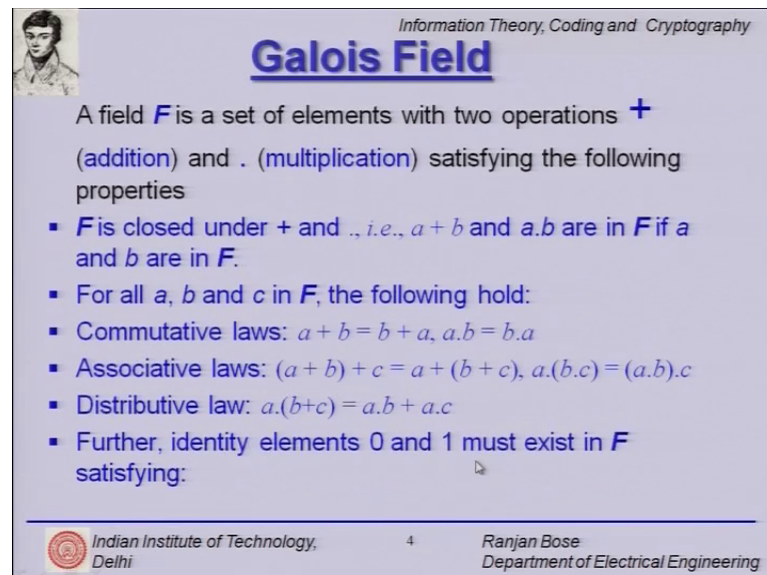
Indian Institute of Technology, Delhi

3

Ranjan Bose
Department of Electrical Engineering

As always we will start with a brief recap of what we have already done. We have looked at block codes. We have defined hamming distance and hamming weights. Then we looked at linear algebra Galois fields, linear block codes. And we introduced this notion of a generator matrix.

(Refer Slide Time: 01:29)



Information Theory, Coding and Cryptography

Galois Field

A field F is a set of elements with two operations $+$ (addition) and \cdot (multiplication) satisfying the following properties

- F is closed under $+$ and \cdot , i.e., $a + b$ and $a \cdot b$ are in F if a and b are in F .
- For all a, b and c in F , the following hold:
- Commutative laws: $a + b = b + a$, $a \cdot b = b \cdot a$
- Associative laws: $(a + b) + c = a + (b + c)$, $a \cdot (b \cdot c) = (a \cdot b) \cdot c$
- Distributive law: $a \cdot (b + c) = a \cdot b + a \cdot c$
- Further, identity elements 0 and 1 must exist in F satisfying:

Indian Institute of Technology, Delhi

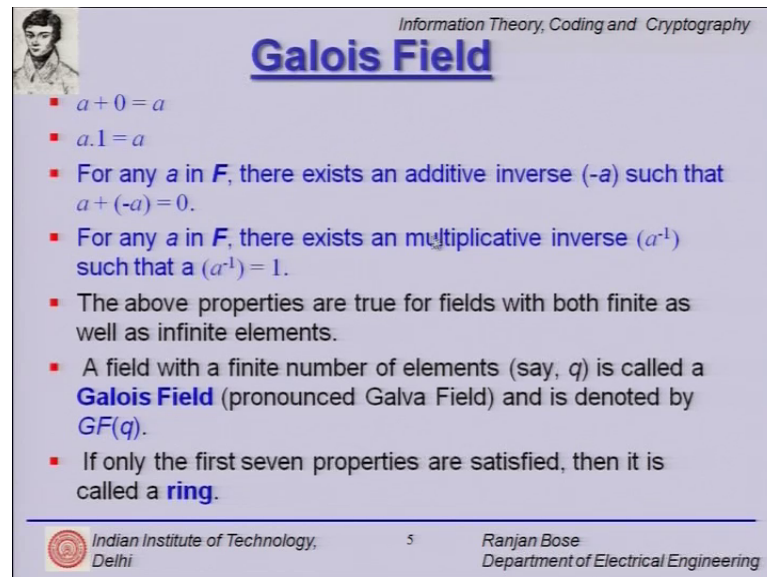
4

Ranjan Bose
Department of Electrical Engineering

So, very quickly, what is a field Galois field, well a field is a set of elements with two operations, addition and multiplication. Defined, and it satisfies the following properties. It is closed under addition, and multiplication. For a b and c contained in the field F we

have the commutative law; the associative law; and the distributive law; we have seen these before. And then, we say that two elements must definitely exist in F which have certain special properties, these two elements are 0 and 1.

(Refer Slide Time: 02:11)



Information Theory, Coding and Cryptography

Galois Field

- $a + 0 = a$
- $a \cdot 1 = a$
- For any a in F , there exists an additive inverse $(-a)$ such that $a + (-a) = 0$.
- For any a in F , there exists an multiplicative inverse (a^{-1}) such that $a \cdot (a^{-1}) = 1$.
- The above properties are true for fields with both finite as well as infinite elements.
- A field with a finite number of elements (say, q) is called a **Galois Field** (pronounced Galva Field) and is denoted by $GF(q)$.
- If only the first seven properties are satisfied, then it is called a **ring**.

Indian Institute of Technology, Delhi

5

Ranjan Bose
Department of Electrical Engineering

What are the properties of 0s and 1. Well, any element a plus the 0 element should give you a . Similarly, any element a into 1 should give you a . And then, for any element a in the field, we have an additive inverse always such that a plus the additive inverse should lead to the element 0. Similarly, we should have a multiplicative inverse a into a inverse should be 1 except for the element 0, all other elements in the field must have a multiplicative in inverse. These properties are true for finite as well as infinite elements in the field.

So, any set of elements say q elements, which satisfy all of these above mentioned properties are called Galois field, so and it is denoted by $GF(q)$, we have seen this before. And in the case, when the last property, the absence of a multiplicative inverse is there that is we do not have a multiplicative inverse for each and every element, but all other properties are satisfied we call it a ring.

So, we now know what is a field, and what is a ring. We will use this very commonly in our study of linear block codes, this is because we have put ourselves the condition that the sum of two code words must be a valid code word. And we have seen observed


before that Galois in a very short period of time made major contributions in mathematics.

(Refer Slide Time: 03:59)

Information Theory, Coding and Cryptography

Generator Matrix

- Any code C is a **subspace** of $GF(q^n)$.
- Any set of **basis vectors** can be used to generate the code space.
- We can, therefore, define a **generator matrix, G** , the rows of which form the basis vectors of the subspace.
- The rows of G will be **linearly independent**.
- Thus, a **linear combination of the rows** can be used to generate the codewords of C .
- The generator matrix will be a **$k \times n$ matrix** with **rank k** .
- Since the choice of the basis vectors is not unique, *the generator matrix is not unique for a given linear code.*

 Indian Institute of Technology, Delhi
Ranjan Bose
Department of Electrical Engineering

We now quickly revisit the concept of a generator matrix. For that, we decide and made this following observation. Any code C is a subspace of $GF(q^n)$. Let us understand this statement with a simple example.

(Refer Slide Time: 04:33)

$(4, 2)$ Code
 $2^4 = 16$ POSSIBLE WORD
 CHOOSE A SUBSET
 $C = \{ 0000, 0101, 1010, 1111 \}$
 $GF(2^4)$
 $GF(2^2)$

ETSC, IIT DELHI

So, if we take a very simple example, a code which is a lookup table, say 0 0 0 1 1 0 and 1 1 has please recall that k equal to 2, and n equal to 4 here, so this is a 4 comma 2 code.

These are my code words. But, please see that even though only four information words are possible, 0 0 0 1 1 0 and 1 1, simply because my k is equal to 2. They are $2^2 = 4$ possible words. But, what we have done, is we have only chosen a subset; A subset, which is 0 0 0 0, 0 1 0 1, 1 0 1 0, and 1 1 1 1.

Now, clearly I could have picked up any other four words, and put it in the table and that would be yet another code right. A code right a code is a set of code words, so this is my code. So, if you go back to the statement, any code C is a subspace of \mathbb{F}_q^n . And in our binary case, it is \mathbb{F}_2^n , and n is 4. So, we have a subspace. So, code is a subspace of \mathbb{F}_q^n .

So, we go back to our slides, and make the observation that any set of basis vectors can be used to generate the code space, but this is the critical observation. A set of basis vectors can be used to generate this space. We can therefore, define a generator matrix G , the rows of which from the basis vectors for the subspace.

Now, clearly if they have to form the basis vectors, the rows of G would be linearly independent. This means, a linear combination of the rows can be used to generate the code words of C , because they form the basis vectors. Any linear combinations of these basis vectors would generate all other points in their subspace. Now, this generator matrix will be of size k cross n , and it should have a rank k , because there k linearly independent rows, and n is necessarily larger than k .

So, since the choice of the basis vectors is not unique, the generator matrix is obviously, not unique for a given linear code. And therefore, we will shortly see, there could be equivalent codes. Why are the equivalent, in what respect are the equivalent, their equivalent in terms of error correcting capability. We are yet to define that, but remember I aim is to have generator matrices for error correcting codes, our aim is to correct from errors recover from errors. And, so in terms of the error correcting codes, we will be able to have equivalent matrices as we will shortly see. With this background and motivation, we would like to formally define.

(Refer Slide Time: 09:01)


Information Theory, Coding and Cryptography

Generator Matrix

- The generator matrix converts (encodes) a vector of length k to a vector of length n .
- Let the input vector (uncoded symbols) be represented by i .
- The coded symbols will be given by

$$c = iG$$

- where c is called the codeword and i is called the information word.

 Indian Institute of Technology,
DelhiRanjan Bose
Department of Electrical Engineering

The generator matrix converts or encodes a vector of length k to a vector of length n . Clearly, it is a k cross n . Let the input vector of uncoded symbols will be represented by i , so i is 1 cross k . And the coded symbol, then be represented by c equal to i times G so, c is 1 cross n is equal to 1 cross k into this dimension is k cross n . So, we end up with a 1 cross n . Here c is called the codeword, i is called the information word, and G is clearly the generator matrix.


(Refer Slide Time: 09:48)

Information Theory, Coding and Cryptography

Generator Matrix

$$c = iG$$

- The generator matrix provides a concise and efficient way of representing a linear block code.
- The $n \times k$ matrix can generate q^k codewords.
- Thus, instead of having a large look-up table of q^k codewords, one can simply have a generator matrix.

 Indian Institute of Technology,
DelhiRanjan Bose
Department of Electrical Engineering

So, now why is this generator matrix important. We saw last time, generator matrix is indeed, an efficient way to represent the lookup table. We saw that even if it is a more greatly large k then, the lookup table size for a linear block code would grow exponentially with k . And it would very soon become impractical to have such large lookup tables, not only from the storage perspective, but also from search perspective.

On the other hand, a generative matrix generates a code word on the fly. It does not have a storage, it does not save the entire list, it just takes in the input vector, and then generates on the fly the code word, this is the beauty of the generator matrix. The n cross k matrix can generate q as power k a code words in general, we are talking about a non binary general case here. So, this elements of the codeword could be 0 1 2 3 up to q minus 1 . It is instead of having a large, actually a very large look-up table of q as power k codewords, one can simply have a generator matrix, which is much much more shorter.

(Refer Slide Time: 11:17)

Information Theory, Coding and Cryptography

Example

- Consider a generator matrix $G = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$

$$c_1 = [0 \ 0] \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} = [0 \ 0 \ 0]$$


$$c_2 = [0 \ 1] \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} = [0 \ 1 \ 0]$$

$$c_3 = [1 \ 0] \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} = [1 \ 0 \ 1]$$

$$c_4 = [1 \ 1] \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} = [1 \ 1 \ 1]$$

↘

↙

 Indian Institute of Technology, Delhi
Ranjan Bose
Department of Electrical Engineering

A very quick example; this is a 2 cross 3 generator matrix, so clearly k is equal to 2, and n is equal to 3. It will take a vector of length 2, and generate vectors of length 3. What we have seen already is that the rows of the generator matrix must be linearly independent, which you can check very easily. And they found the basis vectors, which means that I can take the first row multiplied by scalar add it up to the second row, and it will give me a valid codeword. These themselves are two possible codewords.

So, if you look at this example, if I want to generate the 1st codeword, I can just multiply 0 0 with this, and you get 0 0 0s. Similarly, I can get the 2nd codeword by multiplying 0 1, which is the information word with these to get 0 1 0. And same with the third one, 1 0 multiplied with generator matrixes is gives you this one, and similarly, 4. So, I know there are four code words, and these are highlighted here, and each codeword is of a block length n is equal to 3; A very simple example, but an instructive one.

(Refer Slide Time: 12:45)


Information Theory, Coding and Cryptography

Example

- Hence the generator matrix $G = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$

- Generates the code

$$C = \{000, 010, 101, 111\}.$$

 Indian Institute of Technology,
DelhiRanjan Bose
Department of Electrical Engineering

So, what I have been able to do, I have been able to use these six bits, to generate a table, which is 3 into 4 12 bits. So, already it is a highly compressed efficient way to represent this look-up table, it can generate the lookup table ok. Instead of saving these 12 bits, I have just 3 bits, and I will give you exactly this codewords right.

(Refer Slide Time: 13:20)


Information Theory, Coding and Cryptography

Example

- Hence the generator matrix

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

- We observe that this is a (3, 2) code from the fact that the dimension of the generator matrix is 3×2.

 Indian Institute of Technology, DelhiRanjan Bose
Department of Electrical Engineering


So, this is the generator matrix for a n comma k , which is 3 comma 2 code, and the dimension of the generator matrix is 3 cross 2.

(Refer Slide Time: 13:29)

Information Theory, Coding and Cryptography

Equivalent Codes

- Two q -ary codes are called **equivalent** if one can be obtained from the other by one or both operations listed below:
 - (i) Permutation of the components
 - (ii) Permutation of the position of the codeword.
- Suppose a code containing M codewords are displayed in the form of an $M \times n$ matrix, where the rows represent the codewords.
- Operation (i) corresponds to the re-labeling of the symbols appearing in a given column.
- Operation (ii) represents the rearrangements of the columns of the matrix.

 Indian Institute of Technology, DelhiRanjan Bose
Department of Electrical Engineering

Now, we come to the notion of equivalent codes. And we will see, they are very important and practical. So, two q -ary codes are called equivalent, if one can be obtained from the other by one or both operations listed below. What are these operations, permutations of the components, and permutations of the position of the codewords.

So, please note these definitions are general, but can be applied to binary as well q-ary will become binary codes. Suppose the code contains M codewords, so we keep emphasizing code is a set containing M codewords, and this is in the form of a matrix M cross n. Please remember, M is the total number of code words, n is length of each codeword. So, n cross n represents this look-up table.

Now, operation 1 corresponds to re-labeling of the symbols appearing in the given column, this is operation 1 permutations of components all right. And operation 2, represent the rearrangement of the columns of the matrix, this is permutations of the position of the codewords. So, intuitively, this slide tells us intuitively the properties of a code will not change, if we do these things, properties, the error correcting properties. The ability to correct errors distinguish between different code words that will not change if I carry out these two. And hence, this is called the equivalent codes.

(Refer Slide Time: 15:22)

Information Theory, Coding and Cryptography


Example

- Consider the **ternary code** of blocklength 3
- A code whose components $\in \{0, 1, 2\}$

$$C = \begin{cases} 2 & 0 & 1 \\ 1 & 2 & 0 \\ 0 & 1 & 2 \end{cases}$$

- If we apply the **permutation** $0 \rightarrow 2, 2 \rightarrow 1, 1 \rightarrow 0$ to column 2 and $1 \rightarrow 2, 0 \rightarrow 1, 2 \rightarrow 0$ to column 3 we obtain

$$C_1 = \begin{cases} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 2 & 2 & 2 \end{cases}$$



Indian Institute of Technology,
Delhi

Ranjan Bose
Department of Electrical Engineering

Let us understand this using a simple example. So, let us talk about a ternary code of blocklength 3. What do we mean by a ternary code. A code whose components take from elements 0, 1, and 2 so, let us take a simple example. So, my code, which is the set of codewords, it has only three code words, 2 0 1, 1 2 0, and 1 0 1 2. Now, just a mere scrutiny of this lookup table, tells me that this is indeed a 3 comma 1 code. Why is it a 3 comma 1, n is equal to 3 is obvious, because each of these vectors, each of these code

words is of length 3. But, why is it 1, k is equal to 1; k is equal to 1 means, the information word length is 1 that is either a 0 comes in, or 1 comes in, or a 2 comes in.

So, if I were to appraise this code in terms of a look-up table, I would do the following; I have here k equal to 1, and I have got n equal to 3. So, what we do is possible inputs are 0, 1, and 2, and writing out this example now. And then, 0 is represented by 2, 0, 1, 1 gets represented by 1, 2, 0, and 2 gets represented by 0, 1, 2. Now, if you look at this example that I am writing out, you will see that this is not a linear code, it is a block code, but the all 0 code word is missing.

(Refer Slide Time: 17:40)

$k=1$		$n=3$
0	→	2 0 1
1	→	1 2 0
2	→	0 1 2

$GF(3)$

And on the other hand, if you try to add any two codewords, and here, the arithmetic will be over $GF(3)$. Sum of any two codewords is not another valid code word. For example, if you add these two, 2 plus 1 0, because it is a Galois 3 edition. So, modular arithmetic will hold, because 3 is a prime, 2 plus 0 is 2, 1 plus 0 is 1, so sum of these two codewords is not a valid code word. So, it defies both the properties of a linear block code, nonetheless it is it is a valid code it is a block code.

Now, the question is, can we find an equivalent code for this. So, we apply these permutations that we discussed in the previous slide. Let us say 0 becomes 2, because what is a 0 it is an element; what is 2, it is another element, and nobody is bigger or smaller than anything anybody else, it is just a collection of symbols. So, 0 becomes 2, 2

becomes 1, 1 becomes 0, and you have to apply these two transformations to columns 2 and 3, and suddenly you can come up with the following equivalent code.


(Refer Slide Time: 19:26)

Information Theory, Coding and Cryptography

Example

$$C = \begin{Bmatrix} 2 & 0 & 1 \\ 1 & 2 & 0 \\ 0 & 1 & 2 \end{Bmatrix} \quad C_1 = \begin{Bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 2 & 2 & 2 \end{Bmatrix}$$

- The code C_1 is equivalent to a repetition code of length 3.
- Note that the original code is **not linear**, but is equivalent to a linear code.

 Indian Institute of Technology, Delhi
Ranjan Bose
Department of Electrical Engineering

So, let us observe these two before and after case. Before applying those permutations, and after applying those permutations. We see, that code C_1 , which is an equivalent code is nothing but a repetition code. 0 becomes 0 0 0, 1 becomes 1 1 1, and 2 as an input becomes 2 2 2, it is nothing but a ternary repetition code.


The other important observation is that the original code clearly is not linear, we just checked. But, this equivalent code is linear, what does it mean, well all 0 codeword is a valid code word. The sum of any two codewords, 1 plus 2 0, 1 plus 2 0, 1 plus 2 so, some of these two is this, and some with is 1 1 1 with 0 0 0 is self and so and so far. So, you can check that any pair, when added together yields another valid code word. So, C_1 , which is equivalent is a linear block code, whereas the first one is not.

(Refer Slide Time: 20:44)

Information Theory, Coding and Cryptography

Definition: Equivalent Codes

- Two linear q -ary codes are called **equivalent** if one can be obtained from the other by one or both operations listed below:
 - Multiplication of the components by a non-zero scalar
 - Permutation of the position of the codeword.

 Indian Institute of Technology,
DelhiRanjan Bose
Department of Electrical Engineering

So, let us go to a formal definition. Two linear q -ary codes are called equivalent, if one can be obtained from the other by one or both operations listed below. Multiplication of the components by a non-zero scalar permutation of the position of the code words ok; so, this is what we are stated earlier, but now we are formally defining, how we can declare two q -ary codes to be equivalent.


Please note already we have seen value in declaring or making equivalent codes, because there are very nice techniques available to handle a generate linear block codes, but not so for general block codes. And if you can convert a general block code to a specifically to a linear block code we can make our lives much simpler.

(Refer Slide Time: 21:45)

Information Theory, Coding and Cryptography

Equivalent Linear Codes

- Two $k \times n$ matrices generate equivalent linear (n, k) codes over $GF(q)$ if one matrix can be obtained from the other by a **sequence** of the following operations:
 - Permutation of rows
 - Multiplication of a row by a non scalar
 - Addition of a scalar multiple of one row to another
 - Permutation of columns
 - Multiplication of any column by a non-zero scalar.

 Indian Institute of Technology,
DelhiRanjan Bose
Department of Electrical Engineering

Now, let us talk about equivalent linear codes. So, first we have looked at a non-linear code, we need equivalent to a linear code. Now, I can have one linear code being very equivalent to another linear code. The moment we have a linear code, we have the luxury of the generator matrix that is an important observation. If you have a linear code, you can represent it using a generator matrix.

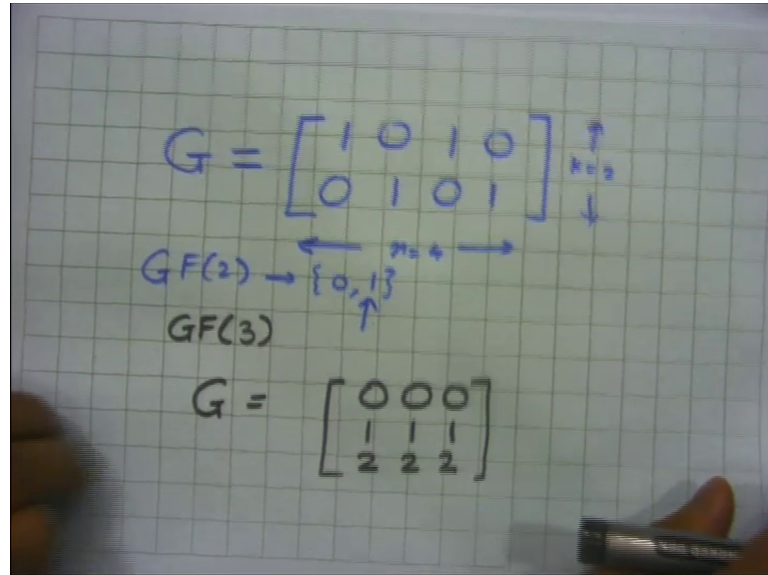
Now, we are talking about two linear codes, and we want to show the equivalence. So, we have two k cross n matrices, this is these are generating matrices, code 1, code 2, and they are suppose to generate equivalent linear codes over $GF(q)$, if one of the generator matrix can be obtained from the other generator matrix by a sequence of the following operations. So, we defining five operations ok. And by conducting these five operations in any sequence and combination, if I can convert one of the generator matrix into another, then these two are termed as equivalent linear code. So, the sequence could be anything ok.

And it could be just the permutation of rows one, permutation of rows can lead to the other generator matrix. Multiplication of a row by a non scalar ok; Addition of a scalar multiple of one row to the other, take to those add them up put it as a third row. Permutations of column, interchange column three and five we end up with an equivalent linear code. Multiplication by any column by non-zero scalar, all these operations in any

sequence if leads from one matrix to the other, then those two matrices, generator matrices are called equivalent linear codes.

Student: (Refer Time: 23:59).

(Refer Slide Time: 24:14)


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

$GF(2) \rightarrow \{0, 1\}$
 $n=4$
 $k=2$

$$G = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 2 & 2 & 2 \end{bmatrix}$$

$GF(3)$

So, here, we are talking about a vector versus a scalar. So, let us show this as an example, the question being asked is what is a scalar here. So, if you look at this an example of a simple generator matrix, in this case we have two rows, so k equal to 2, and n equal to 4. In this case, it is a code over $GF(2)$, so the two elements in the set 0 and 1. And we have already ruled out multiplication by 0. So, multiplication by a scalar is multiplying any of the rows by 1. So, it really does not change that.

But, if you want to make this example, non trivial; let us talk about $GF(3)$. And I have got another generator matrix, which we saw in the last class. Well, in this case, this is not the matrix we are talking about, but this is the code that we talked about.

(Refer Slide Time: 25:40)

$$n=3$$
$$C = \{000, 111, 222\}$$
$$k=1$$
$$G = [1 \ 1 \ 1]$$
$$k \times n$$
$$1 \times 3$$
$$GF(3) \rightarrow \{0, 1, 2\}$$
$$G_1 = [2 \ 2 \ 2]$$
$$\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$$

ETSC, IIT DELHI

So, let us talk about this code, this code is a lookup table, and can be written also as. So, here n is equal to 3, and k is equal to 1 therefore, we had three elements that look-up table. So, if you have to have the generator matrix, the size should be k cross n 1 cross 3. So, the generator matrix is nothing but 1 1 1.

How does it work, well you multiply it with 0, it becomes 0 0 0 0; if you multiplied with 1, becomes 1 1 1; if you multiplied by 2 it becomes 2 2 2, so you have this generator matrix, generate it. Coming back to the question, what is a scalar, now since this is over $GF(3)$, we have the elements 0, 1, and 2. So, I can multiply this G by a scalar 2. So, I can get G_1 , if I multiply this row by 2, I have the multiplication table for $GF(3)$, so I get 2 2 2.

Now, clearly this is generator matrix one, and is the other generator matrix. These two are equivalent, simply because I multiplied one of the rows with a scalar, a vector would be a so a vector could be like 1 2 1 this could be a vector, we do not entertained multiplication by a vector, it will change the dimension and other things. Just a scalar multiplication means, you multiply with any one of the elements, which is non-zero. So, I multiplied with the 1, which leaves it unchanged or a multiply it with 2. So, the point to be noted is, we have just created these two as lineal codes, which are equivalent by scalar multiplication of this row by 2. So, the scalar is 2.

(Refer Slide Time: 28:14)


Information Theory, Coding and Cryptography

Systematic Form

- A generator matrix can be reduced to its **Systematic Form** (also called the **standard form** of the generator matrix) of the type

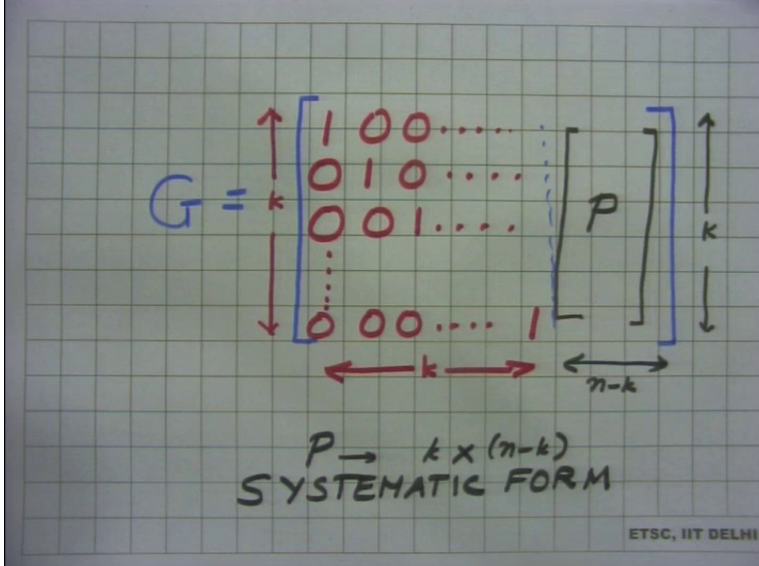
$$G = [I | P]$$

- where I is a $k \times k$ identity matrix and
- P is a $k \times (n - k)$ matrix, called the **Parity Matrix**.

 Indian Institute of Technology, DelhiRanjan Bose
Department of Electrical Engineering

Now, we come back to our slides. And we define another interesting method of representing the linear block codes. A generator matrix can be reduced to its systematic form, which is also called as the standard form of the type G is partitioned into an identity matrix I , and matrix P , which is called the parity matrix. So, G if you note is k cross M , and I is a k cross k identity matrix, and P is a k cross n minus k matrix.

(Refer Slide Time: 29:15)



$P \rightarrow k \times (n-k)$
SYSTEMATIC FORM

ETSC, IIT DELHI

So, if you now go back to the drawing board, and have a look you have this G matrix, which we now define, in terms of an identity matrix first. So, we have a $1\ 0\ 0$. So, this is


k, and this is k, but we left with is now n minus k on one side, and again this is k. So, my P matrix is of the dimension k cross n minus k, this is called the systematic form.

(Refer Slide Time: 30:31)

Information Theory, Coding and Cryptography

Systematic Form

- A generator matrix can be reduced to its **Systematic Form** of the type $G = [I | P]$
- **Proof:**
- The k rows of any generator matrix (of size $k \times n$) are **linearly independent**.
- Hence, by performing **elementary row operations and column permutations** it is possible to obtain an equivalent generator matrix in a row echelon form.
- This matrix will be of the form $[I | P]$.

 *Indian Institute of Technology,
Delhi* *Ranjan Bose
Department of Electrical Engineering*

So, a generator matrix can be reduced to its systematic form of this type by those basic row and column operations. And the proof is pretty intuitive, the k rows of a generator matrix are linearly independent, we saw that is they form the basis vectors. Hence, a performing elementary row operations and column permutations, remember there k rows, so you will be able to place a single 1, and all 0s in the first few columns, and thereby obtain the row echelon form.

And the matrix will be of the form I partition P ok, because very simply you have got rank k matrix, the k rows. So, first k rows of the matrix, it can easily be converted into linearly independent, so all of them will have a single 1 which does not overlap with the following columns.

(Refer Slide Time: 31:39)

Information Theory, Coding and Cryptography


Example

- Consider the generator matrix of a (4, 3) code over $GF(3)$:

$$G = \begin{bmatrix} 0 & 1 & 2 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 2 & 2 & 1 \end{bmatrix}$$

- Let us represent the i^{th} row by r_i and the j^{th} column by r_j . Upon replacing r_3 by $r_3 - r_1 - r_2$ we get (note that in $GF(3)$, $-1 = 2$ and $-2 = 1$ because $1 + 2 = 0$)

$$G = \begin{bmatrix} 0 & 1 & 2 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 2 & 0 \end{bmatrix}$$

 Indian Institute of Technology, Delhi

Ranjan Bose
Department of Electrical Engineering

So, let us look at an example. Let us consider the generator matrix for a 4 comma 3 code over $GF(3)$; what does it mean, it means that the size of the generator matrix is $k=3$ by $n=4$. And $GF(3)$ represents that the elements of the generator matrix will be 0 or 1 or 2. And all the arithmetic, and the addition in the multiplication will be carried over $GF(3)$, which will be modular 3 arithmetic.

So, the example is as follows. G is again a 3×4 generator matrix. And it tells you that the k value is 3. Now, we can start doing a basic row and column operations, and this requires a little bit of thinking observation and see, because right now there is no single 1 in any column. What are we trying to do, you observe the first column 0 1 1, what we need is 0 0 1 a single 1 in three columns. And then, interchange of columns is very easy, we have discussed that before there only two equivalent codes. So, we are trying to make an equivalent code starting from this jumble of numbers.

So, if we will represent the i^{th} row by r_i , and j^{th} column by r_j . What we can do is, replace r_3 last row by $r_3 - r_1 - r_2$. So, I am doing this basic operations, so r_3 , then r_1 first 1, and r_2 . But, please remember it is equivalent to same r_3 plus 2 times r_1 plus 2 times r_2 , because minus 1 is equal to 2 and $GF(3)$; why is that, because 1 plus 2 is 0, so 2 is equal to minus 1 or minus 2 is equal to 1.

So, either I do this statement $r_3 - r_1 - r_2$ or I can say that $r_3 + 2r_1 + 2r_2$. So, again 2 r_1 is multiplying the row 1 by scalar 2, and then again taking the row 2

multiplying it with another scalar, which is 2, and adding them up. So, addition of rows are permitted, you still end up with an equivalent code, we do that.

And if you do that, basic operation you get the first 2 rows untouched, because I have only replaced r_3 . And if you do, so first rows are the same r_3 is changed by doing these operations, you can check that. And still we have far from done, how did it help us. Well look at the first column, first column helped me get rid of this one. So, in this first column there is only single 1, and rest are all 0s. So, parts of the jumble, the jigsaw puzzle are fitting together, this will form my second column of my identity matrix.

Look at this last column 1 0 0, the previous operation led me to a single 1 in this column. So, this will hopefully form the first column of my identity matrix. But, those are the 2 once, I still need to get a 0 0 1 somewhere, I need to knock of this 1 somewhere by carefully thinking, how to scalar multiply this and this.

So, my first bad feeling to knock this off is 1 plus 2 is 0. So, I have to add a 2 here. Now, I observe in the last row that there is a 1 here, so if I multiply by scalar 2 here, so this is will become 0 2 and 2 into 2s modular 3 operation is 1 and 0. So, the movement I have with 2 and I add this to the first row, I will be able to get a 0 here, this will remain unchanged, and this will remain unchanged. So, that will be logical step to go.

(Refer Slide Time: 36:16)

Information Theory, Coding and Cryptography


Example

- Next we replace r_1 by $r_1 - r_3$ to obtain

$$G = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 2 & 0 \end{bmatrix}$$

- Finally, shifting $c_4 \rightarrow c_1, c_1 \rightarrow c_2, c_2 \rightarrow c_3$ and $c_3 \rightarrow c_4$ we obtain the standard form of the generator matrix

$$G = \begin{bmatrix} 0 & 1 & 2 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 2 & 2 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 2 \end{bmatrix} = [I \ P]$$

 Indian Institute of Technology, Delhi
Ranjan Bose
Department of Electrical Engineering

But we can do some other things. So, we can replace r_1 by $r_1 - r_3$, and we can get another kind of a thing. So, we have got $0\ 1\ 0\ 0\ 0\ 1\ 1\ 0\ 0$ has three columns, which are candidates for my identity matrix in the beginning. All I have to do is interchange the columns, the order of the columns can be changed. So, I do that c_4 goes to c_1 c_1 these a columns, column 1 becomes column 2, column 2 becomes so and so forth. And then, at suddenly things fall into place all right.


So, I can take this thing, and do the interchanging, and I can see suddenly that very clearly this is identity matrix in the beginning. And whatever else remains, this $0\ 1\ 2$, which remains is the P , the parity matrix. So, these steps just tells us, how we can use the basic row, and column operations to convert a generator matrix into the systematic form, this is the standard form.

(Refer Slide Time: 37:36)

Information Theory, Coding and Cryptography

Efficient Decoding

- One of the **objectives** of a good code design is to have fast and efficient encoding and decoding methodologies.
- So far we have dealt with the **efficient generation** of linear block codes using a generator matrix.
- Codewords are obtained simply by multiplying the input vector (uncoded word) by the **generator matrix**.
- **Is it possible to detect a valid codeword using a similar concept?**



Indian Institute of Technology,
Delhi

Ranjan Bose
Department of Electrical Engineering

Now, so far we will only talked about efficient representation, and encoding process, we have not talked about decoding. But, one of the stated objectives of a good error control code is an efficient encoding as well as a decoding process. So, let us focus or attention on efficient decoding. We are dealt with efficient generation can we have an equivalent decoding matrix ok, because we cannot say that look.

We have reduced the problem of look-up table at that transmission side, where the receiving, and why do not you have a look-up table, you have received a valid code word look it up and reverse and try to get back the original information world. So, is it possible

to detect a code word is valid quickly; One of the way is, which is inefficient is to go through the long table. The lookup table and check whether it matches, and that is a pain.

(Refer Slide Time: 38:57)

Information Theory, Coding and Cryptography

Efficient Decoding

- Is it possible to detect a valid codeword using a similar, matrix-based approach?
- The answer is yes, and such a matrix is called the **parity check matrix, H** , for the given code.
- For a parity check matrix,


$$cH^T = \mathbf{0}$$

where \mathbf{c} is a valid codeword.

$$c = iG, \text{ therefore, } iGH^T = \mathbf{0}.$$

- For this to hold true for all valid codewords, we must have

$$GH^T = \mathbf{0}.$$

 Indian Institute of Technology, Delhi

Ranjan Bose
Department of Electrical Engineering

So, question is can we have a very nice matrix representation to do the same thing. The answer of course, is yes. This is a question have post, the answer lies in the parity check matrix, typically denoted by H . But, what properties should this parity check matrix have.

Now, word of caution this is called the parity check matrix. But, if you have seen earlier was the parity matrix P will be formed the systematic representation of a generator matrix. So, please note the difference we have seen the parity matrix for the systematic form. And now, we are talking about the parity check matrix H , at the decoding side.

Now, what is a wish list. Despite a check matrix as the name suggests should be able to see, whether the codeword is valid or not. So, my wish list is that any valid code word \mathbf{c} , when multiplied with H transform should give me a $\mathbf{0}$ vector. On the other hand, if \mathbf{c} is not a valid code word, then I should get a nonzero vector here ok. So, I if I can have this going for me, then I indeed it jackpot, and I should be able to very very quickly detect. First step is detection, detect if there is an error or not.

Now, we make a quick observation, what is \mathbf{c} , \mathbf{c} after all is i into G , i is information word, G is the generator matrix. So, if you put that \mathbf{c} equal to iG in this first equation,

you get $\mathbf{G}^T \mathbf{G} \mathbf{H}^T$ equal to 0 for all possible information word. I cannot have it specific to any particular information word, this means that I should have $\mathbf{G} \mathbf{H}^T$ equal to 0, in order to make it general enough for all information word. So, if we have a generator matrix, we should be able to find out an H matrix and also vice versa. If I have an H matrix, I should be able to find a generator matrix.


Now, how do we do that, well one of the bigger longer strenuous way is to represent this H matrix generally by the elements, $H_{11} H_{12} H_{13}$ for different rows; And you already have a G matrix, then you have a set of linear equations, and you solve it. You will realize that this problem is under constraint, and you can have several solutions for H. So, tells us that the H matrix need not be unique, you can have several valid parity check matrices for one particular G matrix, you can check it out ok.

(Refer Slide Time: 42:11)

Information Theory, Coding and Cryptography

Parity Check Matrix

- The size of the parity check matrix is $(n - k) \times n$.
- A parity check matrix provides a simple method of **detecting** whether an error has occurred or not.
- If the multiplication of the received word (at the receiver) with the transpose of H yields a **non-zero vector**, it implies that an error has occurred.
- This methodology, however, will fail if the errors in the transmitted codeword **exceeds** the number of error for which the coding scheme is designed.



Indian Institute of Technology,
Delhi

Ranjan Bose
Department of Electrical Engineering

Now, the size of the parity check matrix is n minus k cross n . This parity check matrix provides a very simple, and elegant way of detecting whether an errors occurred or not. If the multiplication the received word, at the receiver with the transpose of H yields of non-zero vectors, it is raises the flag, it implies that the error has occurred. Now, question is what error, and if you can answer that what error, then we would be able to go back and correct the error ok.

How would this methodology will fail, if the number of errors in the codeword exceeds the number of errors for which the coding scheme is designed. So, this brings us to a

very interesting observation. Error control codes are designed for a certain number of errors. So, if the number of errors are within the limit, everything works well, but if you exceed then all of these decoding mechanisms will start failing.

(Refer Slide Time: 43:19)


Information Theory, Coding and Cryptography

Parity Check Matrix

- Suppose the generator matrix is represented in its systematic form $G = [I | P]$.
- The matrix P is called the **coefficient matrix**.
- Then the parity check matrix will be defined as

$H = [-P^T | I],$
- where P^T represents the transpose of matrix P .
- This is because

$$GH^T = [I | P] \begin{bmatrix} -P \\ I \end{bmatrix} = \mathbf{0}.$$

 Indian Institute of Technology,
Delhi

Ranjan Bose
Department of Electrical Engineering

So, let us just look at a smart way to generate parity check matrix. We have already seen that G matrix can be represented as in the standard systematic form, identity matrix, and a parity matrix right. P sometimes also called as the coefficient matrix, but in general we will call it as the parity matrix.

Now, the parity check matrix can simply be defined as minus P transpose and I, but this I will be n cross n minus k. So, P T represents the transpose of the matrix P. And this is easy to verify if you have your G H transpose, and G is in the systematic form I in I partition P, and you have this minus P transpose, here there is a transpose I; and if you multiply it out, you will end up getting a 0 matrix.

(Refer Slide Time: 45:52)

Information Theory, Coding and Cryptography


Example

- and P^T is given by

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

- Observing the fact that $-1 = 1$ for the case of binary, we can write the parity check matrix as

$$H = [-P^T \mid I]$$
$$= \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

 Indian Institute of Technology,
DelhiRanjan Bose
Department of Electrical Engineering

Now, to get the P transpose just flip it over, and we have now a 3 cross 4 P transpose matrix. Now, we have to put a negative sign. But, we are working with binary codes, binary means, 1 plus 1 is 0, so minus 1 is 1 minus is plus, so minus P transpose is nothing but P transpose. So, we have already got that stuff. All we have to do is append the I n minus k matrix here. Now, n is 7, k is 4, so n minus k is 3.

So, we must append a 3 cross 3 identity matrix at the back, and we are home free. So, H is minus P transpose partition I this we have already figured out. And we put this 3 cross 3 matrix at the back, and we have our H matrix. So, this H matrix should be able to give you a 0 vector output for any valid code word generated by this generator matrix.

(Refer Slide Time: 46:58)

Information Theory, Coding and Cryptography

Summary

- Linear Block Codes
- Equivalent Codes
- Generator Matrix
- Parity Check Matrix
- Examples

Indian Institute of Technology, Delhi

26

Ranjan Bose
Department of Electrical Engineering

So, with this we come to the end of this lecture. Just let us summarize what we have learned so far. We started looking at linear block codes. Then we spent some time looking at equivalent codes, because they are pretty helpful the very handy mathematical tools, to how to make one code equivalent to another code. Then we looked at the notion of generator matrices. And finally, we moved over to the decoding side, and looked at parity check matrix. Of course, we looked at a few examples on the way. With that we come to the end of this lecture.