

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

Module – 21
Cyclic Codes
Lecture – 21

(Refer Slide Time: 00:32)

The slide is titled "Outline" and lists the following topics:

- Cyclic Codes
- Generator Polynomial
- Syndrome Polynomial
- Matrix Representation

The slide footer contains the Indian Institute of Technology Delhi logo, the page number "2", and the name "Ranjan Bose, Department of Electrical Engineering".

Hello and welcome to our next module on cyclic codes. Let us start with a quick outline for today's lecture. We will start looking at the details of cyclic codes how to construct them, how to look at their properties. We will introduce the notion of generator polynomial, the syndrome polynomial. And finally move on to the matrix representation. Please recall that cyclic codes are a subclass of the linear block codes.

(Refer Slide Time: 00:57)

Information Theory, Coding and Cryptography

Recap

- Ring and Fields
- Polynomials
- Division Algorithm
- Cyclic Codes
- Examples

Indian Institute of Technology, Delhi 3 Ranjan Bose
Department of Electrical Engineering

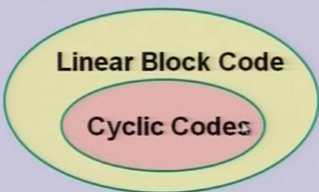
But first let us have a very quick recap as to what you have studied so far. We have talked about rings and fields, and polynomials, and how these mathematical tools help us construct cyclic codes.

(Refer Slide Time: 01:11)

Information Theory, Coding and Cryptography

Cyclic Codes

- A code C is **cyclic** if
 - C is a linear code, and,
 - any **cyclic shift** of a codeword is also a codeword, i.e., if the codeword $a_0a_1\dots a_{n-1}$ is in C then $a_{n-1}a_0\dots a_{n-2}$ is also in C .



Indian Institute of Technology, Delhi 4 Ranjan Bose
Department of Electrical Engineering

We have seen what is a cyclic code. Let us redefine it for clarity. A code C is cyclic, if number one C is a linear code; and any cyclic shift of a valid codeword all the results in another valid code word. So, if $a_0 a_1 \dots a_{n-1}$ is a valid codeword then a

cyclic shift means a n minus 1 comes in the front and then each one get shifted by 1 to the right and that is also valid codeword.


So, if you see the first point says that if this is a linear block code then cyclic code is a subclass. All that we have studied regarding linear block codes holds true for cyclic codes, but we have introduced an additional algebraic constraint that is the linear cyclic shift of any valid codeword is another valid codeword consequently this is a subclass and it is a much stronger code.

(Refer Slide Time: 02:16)

Information Theory, Coding and Cryptography

Example

- The binary code $C_1 = \{0000, 0101, 1010, 1111\}$ is a cyclic code.
- However $C_2 = \{0000, 0110, 1001, 1111\}$ is not a cyclic code, but is *equivalent* to the first code.
- Interchanging the third and the fourth components of C_2 yields C_1 .

 Indian Institute of Technology,
Delhi5Ranjan Bose
Department of Electrical Engineering


We have looked at some examples in the last lecture. And here is a quick-one we have a valid cyclic code here with 4 codewords. And if you can quickly verify, it is a linear block code the all 0 codeword is a valid codeword then sum of any two codeword gives another valid codeword. And more importantly any cyclic shift also results in a valid codeword.

(Refer Slide Time: 02:40)

Information Theory, Coding and Cryptography

Polynomials

- A **polynomial** is a mathematical expression
$$f(x) = f_0 + f_1x + \dots + f_mx^m,$$
where the symbol x is called the indeterminate and the coefficients f_0, f_1, \dots, f_m are the elements of $GF(q)$.
- The coefficient f_m is called the leading coefficient.
- If $f_m \neq 0$, then m is called the **degree** of the polynomial, and is denoted by $\deg f(x)$.
- A polynomial is called **monic** if its leading coefficient is unity.
- **Example:** $f(x) = 3 + 7x + x^3 + 5x^4 + x^6$ is a monic polynomial over $GF(8)$.
- The degree of this polynomial is 6

 Indian Institute of Technology, Delhi 6 Ranjan Bose
Department of Electrical Engineering


We then introduced the notion of polynomials, and why they are important. Polynomials in general are represented as follows $f(x)$ is equal to f_0 plus f_1x plus so and so forth till f_mx^m , where f_m is called the leading coefficient, and m is called the degree of the polynomial. If f_m which is the leading coefficient is 1, then it is a monic polynomial.

(Refer Slide Time: 03:10)

Information Theory, Coding and Cryptography

Division Algorithm

- The **division algorithm** states that, for every pair of polynomial $a(x)$ and $b(x) \neq 0$ in $F[x]$, there exists a unique pair of polynomials $q(x)$, the quotient, and $r(x)$, the remainder, such that $a(x) = q(x)b(x) + r(x)$, where $\deg r(x) < \deg b(x)$.
- The remainder is sometimes also called the **residue**, and is denoted by $R_{b(x)}[a(x)] = r(x)$.
- **Two important properties of residues are**
- $R_{f(x)}[a(x) + b(x)] = R_{f(x)}[a(x)] + R_{f(x)}[b(x)]$, and
- $R_{f(x)}[a(x) \cdot b(x)] = R_{f(x)}\{R_{f(x)}[a(x)] \cdot R_{f(x)}[b(x)]\}$
where $a(x)$, $b(x)$ and $f(x)$ are polynomials over $GF(q)$.

 Indian Institute of Technology, Delhi 7 Ranjan Bose
Department of Electrical Engineering

We also looked at the division algorithm, where you find out that for a pair of polynomials $a(x)$ and $b(x)$ where $b(x)$ is not equal to 0 in $F[x]$. $F[x]$ is a set of polynomials,

then you can have a unique pair of polynomials $q(x)$ and $r(x)$, which are nothing but the quotient polynomial and the remainder polynomial, such that $a(x)$ is equal to $q(x)$ times $b(x)$ plus $r(x)$, where degree of this remainder polynomial must necessarily be less than the divisor $b(x)$. So, we did this example and sometimes we also called the remainder as the residue.


And two important properties that we will use frequently are that the sum of two polynomials the residue of it is the sum of the residues. And similarly we have a property for the product of two polynomials. We have seen that polynomials can be added, subtracted, multiplied and divided. And it is seen that the coefficients follow the Galva field over which the arithmetic is carried out. So, please note that $a(x)$, $b(x)$ and $f(x)$ are defined over some Galva field, and it is important because multiplication, addition, division all will change because if the Galva field changes the arithmetic changes.

(Refer Slide Time: 04:33)

Information Theory, Coding and Cryptography

Irreducible

- **Definition:** A polynomial $f(x)$ in $F[x]$ is said to be **reducible** if $f(x) = a(x) b(x)$, where $a(x)$, $b(x)$ are elements of $f(x)$ and $\deg a(x)$ and $\deg b(x)$ are both smaller than $\deg f(x)$.
- If $f(x)$ is not reducible, it is called **irreducible**.
- A monic irreducible polynomial of degree at least one is called a **prime polynomial**.



Indian Institute of Technology,
Delhi

8

Ranjan Bose
Department of Electrical Engineering

We then introduce the notion of irreducible, which is came to whether it is a prime polynomial, whether you can factorise it or not. So we found out that if a polynomial $f(x)$ is contained in the set of polynomials capital F of x , F of x is said to be reducible if it can be written as a product of two polynomials right. So you can factorise $f(x)$ but this is a product follows the rule of the Galva field over which we carry out the multiplication. So, it is possible that sum $f(x)$ is reducible in some Galva field but irreducible in another

galva field. And if the irreducible polynomial also happens to be monic that is the leading coefficient is one, it is also called as a prime polynomial.

(Refer Slide Time: 05:20)

Information Theory, Coding and Cryptography

Field

- **The ring $F[x]/f(x)$ is a field if and only if $f(x)$ is a prime polynomial in $F[x]$.**
- Example: Consider $p(x) = x^3 + x + 1$ over $GF(2)$.
- Since, $p(0) \neq 0$ and $p(1) \neq 0$, the polynomial is irreducible in $GF(2)$.
- Since it is also monic, $p(x)$ is a **prime polynomial**.
- Here we have $n = 3$, so we can use $p(x)$ to construct a field with $2^3 = 8$ elements, i.e., $GF(8)$.
- The elements of this field will be $0, 1, x, x + 1, x^2, x^2 + 1, x^2 + x, x^2 + x + 1$, which are all possible polynomials of degree less than $n = 3$.

Indian Institute of Technology,
Delhi

9

Ranjan Bose
 Department of Electrical Engineering


We also introduced the notion of a field where the ring capital F of x by f of x is a field if and only if f of x is a prime polynomial in F x. So, we look at this example p of x it is a prime polynomial, and we can construct a field G F 8 with 8 elements using this p x. Please note that when you carry out capital F of x divided by a small f of x. And in this case we are putting at p of x then the residues all the reminders have a degree 2 or less. So, if the degrees are 2 or less, then we have 8 distinct polynomials here.

(Refer Slide Time: 06:17)

Information Theory, Coding and Cryptography

Observations

- $x^n = 1 \pmod{x^n - 1}$.
Hence, any polynomial, modulo $x^n - 1$, can be reduced simply by replacing x^n by 1, x^{n+1} by x and so on.
- A codeword can *uniquely* be represented by a polynomial.
We can use a polynomial to represent the locations and the values of all the element in the codeword.
- For example, the codeword $c_1 c_2 \dots c_n$ can be represented by the polynomial $c(x) = c_0 + c_1x + c_2x^2 + \dots + c_nx^n$.
- As another example, the codeword over $\mathbb{GF}(8)$, $c = 207735$ can be represented by the polynomial $c(x) = 2 + 7x^2 + 7x^3 + 3x^4 + 5x^5$.

 Indian Institute of Technology, Delhi 10 Ranjan Bose
Department of Electrical Engineering

We made some observations. First and foremost is that if you have x^n and you take it mod $x^n - 1$, it equals to 1. So, we will use this term modulo $x^n - 1$ that is this is a polynomial, and we will divide our products, editions etcetera by $x^n - 1$, and whatever is the remainder is what we will deal with. So, x^n , x^{n+1} when you do modulo $x^n - 1$, it is unity.

Please note we also observed that a codeword can uniquely be represented by polynomials. This is how we connect our polynomials to codewords. For example, if your codeword is c_1, c_2, \dots, c_n and you can write it as $c_0 + c_1x + c_2x^2 + \dots + c_nx^n$ and so forth. So, the codeword can be $c_0, c_1, c_2, \dots, c_n$.


For example, c is equal to 207735 this is my codeword. And I have an equivalent polynomial representation as $2 + 7x^2 + 7x^3 + 3x^4 + 5x^5$ and so forth. So, 2 corresponds to the first coefficient, 0 corresponds to the missing 0 into x so that we do not write, and then $7x^2$ corresponds to the third, so in a way the degree of the x the power of x actually represents the location of the coefficient. And these coefficients can belong to the Galois field under consideration here it is $\mathbb{GF}(8)$.

(Refer Slide Time: 07:59)

Information Theory, Coding and Cryptography

Observations

- Multiplying any polynomial by x corresponds to a single cyclic right-shift of the codeword elements.
- More explicitly, in R_n , by multiplying $c(x)$ by x we get
$$\begin{aligned}x \cdot c(x) &= c_0x + c_1x^2 + c_2x^3 + \dots + c_{n-2}x^{n-1} + c_{n-1}x^n \\ &= c_{n-1} + c_0x + c_1x^2 + c_2x^3 + \dots + c_{n-2}x^{n-1}.\end{aligned}$$
- Recall $x^n = 1 \pmod{x^n - 1}$.
Hence, any polynomial, modulo $x^n - 1$, can be reduced simply by **replacing x^n by 1, x^{n+1} by x and so on.**

 Indian Institute of Technology, Delhi 11 Ranjan Bose
Department of Electrical Engineering

So, why are we looking at polynomials in the context of cyclic codes. First observation is if you multiply any polynomial by x , it tantamounts to the cyclic right-shift of the codeword elements, because you simply raise the power of x by 1 for each one of those x s. So, if you had $c(x)$ as $c_0 + c_1x + \dots$ and so forth, then $x \cdot c(x)$ is nothing but $c_0x + c_1x^2 + c_2x^3 + \dots$ each x has gone up. But now your x^n is the highest power, but my $c(x)$ if it were to represent a codeword should have the highest power x^{n-1} . Because the n element from c_0 up to c_{n-1} .

So, in order to put this last coefficient in the front to make it a cyclic code, we have to take modulo $x^n - 1$. So, just the operation of taking this polynomial, and if I divide it by $x^n - 1$, I simply get this c_{n-1} in the front and my cyclic shift operation is complete. So, there are still n elements in my codeword, but they have all undergone a cyclic shift. So, please note we have used this property. So, any polynomial, modulo $x^n - 1$ can be reduced simply by replacing x^n by 1, x^{n+1} by x and so forth.


(Refer Slide Time: 09:40)

Information Theory, Coding and Cryptography

Back to Cyclic Codes

- A code C in R_n is a cyclic code if and only if C satisfies the following conditions: $F[x]/f(x)$ is R_n

$$a(x), b(x) \in C \Rightarrow a(x) + b(x) \in C$$
$$a(x) \in C \text{ and } r(x) \in R_n \Rightarrow a(x)r(x) \in C.$$

 Indian Institute of Technology,
Delhi12Ranjan Bose
Department of Electrical Engineering

So, we have redefined cyclic codes, cyclic codes in R_n is a cyclic code if and only if C satisfies the condition that this collection of polynomials capital F of x divided by $f(x)$ is in R_m . So, if $a(x)$ and $b(x)$ are two polynomials belonging to C . So suddenly we are talking about polynomials as element of a code, no problem because polynomials have a one-to-one correspondence with codewords.

So, if 2 codewords belong to C , I can equivalently state that $a(x)$ which corresponds to codeword 1, and $b(x)$ which corresponds to codeword 2, they both belong to C . And consequently sum of two any codewords here in this case codeword polynomials is also in C . Similarly if I take $a(x)$ and take any other polynomial $r(x)$, and then I am multiply them then also we get it as a valid codeword.


(Refer Slide Time: 10:50)


Information Theory, Coding and Cryptography

Generating Cyclic Codes

- The following steps can be used to generate a cyclic code:
- Take a polynomial $f(x)$ in R_n .
- Obtain a set of polynomials by multiplying $f(x)$ by all possible polynomials in R_n .
- The set of polynomials obtained above corresponds to the set of codewords belonging to a cyclic code.
- The blocklength of the code is n .

13

 Indian Institute of Technology, Delhi

 Ranjan Bose
Department of Electrical Engineering

We then looked at how to generate cyclic codes. And the steps are given as follows take any polynomial f of x in R_n . And obtain a set of polynomials by multiplying $f x$ by all possible polynomials in R_n . The set of polynomials obtained above correspond to the set of codewords. So, basically we have some kind of a notion of a generator polynomial; and the block length will be n .


(Refer Slide Time: 11:26)


Information Theory, Coding and Cryptography

Generator Polynomial

- Let C be a (n, k) non zero cyclic code in R_n .
- 1** ▪ Then there exists a unique monic polynomial $g(x)$ of the smallest degree in C ,
- 2** ▪ the cyclic code C consists of all multiples of the **generator polynomial** $g(x)$ by polynomials of degree $k - 1$ or less.
- 3** ▪ $g(x)$ is a factor of $x^n - 1$.

14

 Indian Institute of Technology, Delhi

 Ranjan Bose
Department of Electrical Engineering

So, let us now formally introduce the generator polynomial. Let C be an n comma k non zero cyclic code in R_n . Then there exists a unique monic, this is important unique monic

polynomial g of x of the smallest degree C that is there could be many polynomials which can do the job. But we are talking about the one with the smallest degree and that we defined as g of x .

The cyclic code C consists of all multiples of the generator polynomial g of x the polynomial of degree k minus 1 or less. And most importantly g of x is a factor of x raise n minus 1. Now, this is probably the most important observation here, if we state that g of x is a factor of x is n minus 1. That means, we have a now recipe to create all possible cyclic codes. All we have to do is factorise x raise power n minus 1 in the appropriate Galva field, and each factor is potentially the generator polynomial of a cyclic code. So, all we have to do is learn to factorise x raise power n minus 1 and we have the valid generator polynomials.

So, let us look at this 1, 2, 3 properties. And we will try to see whether we can prove them quickly and will look at it intuitively. So number 1, this a unique and monic polynomial g of x of smallest degree, number 2 that if a cyclic code C consists of all multiples of the generator polynomial g of x by polynomials of degree k minus 1 or less. So if I multiply g of x with any arbitrary polynomial of degree k minus 1, then I get a valid codeword polynomial. And finally, the most important one g of x is a factor of x raised by n minus 1. So let us look at 1, 2, 3 individually.

(Refer Slide Time: 13:45)

Information Theory, Coding and Cryptography

Generator Polynomial

1

- There exists a unique monic polynomial $g(x)$ of the smallest degree in C
- Suppose both $g(x)$ and $h(x)$ are monic polynomials in C of smallest degree.
- Then $g(x) - h(x)$ is also in C , and has a smaller degree.
- If $g(x) = h(x)$, then a suitable scalar multiplier of $g(x) - h(x)$ is monic, and is in C , and is of smaller degree than $g(x)$.
- This gives a contradiction.

Indian Institute of Technology, Delhi

15

Ranjan Bose
Department of Electrical Engineering

So, number 1 says that there exists a unique monic polynomial g of x of the smallest degree C . So, to prove or disprove this, we have to see that suppose both g of x and h of x are monic polynomials in C of the smallest degree, fine, I mean either this is true or g of x and h of x have been able to find one more candidate, and then it will no longer be unique.

Then g of x minus h of x is also in C . Why, because g of x is a valid polynomial you multiply it with 1, it should give itself and any multiplication with a polynomial of g of x should give you a valid codeword polynomial. So, g of x valid and h of x is another candidate, so the difference of these two should also be a valid codeword polynomial. Therefore, we say that g of x minus h of x is also in C . But please note both of them are monic and if you subtract them then the degree goes down. And therefore, we violate the smallest degree constraint. So, this gives a contradiction.


(Refer Slide Time: 14:56)

Information Theory, Coding and Cryptography

2

Generator Polynomial

- The cyclic code C consists of all multiples of the **generator polynomial** $g(x)$ by polynomials of degree $k - 1$ or less
- Let $a(x) \in C$.
- Then, by division algorithm, $a(x) = q(x)g(x) + r(x)$, where $\deg r(x) < \deg g(x)$.
- But $r(x) = a(x) - q(x)g(x) \in C$ because both word on the right hand side of the equation are codewords.
- However, the degree of $g(x)$ must be the minimum among all codewords.
- This can only be possible if $r(x) = 0$ and $a(x) = q(x)g(x)$.
- Thus, a codeword is obtained by multiplying the generator polynomial $g(x)$ with the polynomial $q(x)$.
- For a code defined over $GF(q)$, there are q^k distinct codewords.
- These codewords correspond to multiplying $g(x)$ with the q^k distinct polynomials, $q(x)$, where $\deg q(x) \leq (k - 1)$.



Indian Institute of Technology,
Delhi

16

Ranjan Bose
Department of Electrical Engineering

We now look at number point number 2, the cyclic code C consists of all multiples of the generator polynomial g of x the polynomials of degree k minus 1 or less. So, let us say a of x is an element of C all right, so it is a valid codeword polynomial. Then we know that a of x can be always written as sum q of x quotient polynomial times x g of x plus r of x , with degree of r of x should be necessarily be less than g of x . So we can always right it like this from the division algorithm.

But if you flip it around and put $r(x)$ then $r(x)$ is nothing but $a(x) - q(x)g(x)$ right. But this $r(x)$ must be an element of C must be a valid codeword polynomial. Why because $a(x)$ we started off with a valid $g(x)$ and $q(x)$ right, so they are also valid codeword polynomials. With a degree of $g(x)$ must be minimum amongst all codewords right, but this is only possible if $r(x) = 0$. And consequently $a(x)$ must be equal to some polynomial $q(x)$ of x into $g(x)$. So that means multiplying $g(x)$ by some arbitrary polynomial $q(x)$ of x should be equal to $a(x)$ the valid codeword polynomial.

So, for a code defined over $GF(q)$, there are q^k distinct codewords. And these codeword correspond to multiplying this generator polynomial $g(x)$ with the q^k distinct polynomials, $q(x)$, where degree of $q(x)$ is less than $k - 1$. So, in this way we show that I can generate the entire code space by simply multiplying my $g(x)$ with any arbitrary polynomial of degree $k - 1$ or less.


(Refer Slide Time: 17:07)

Information Theory, Coding and Cryptography

Generator Polynomial

3

- $g(x)$ is a factor of $x^n - 1$
 - By division algorithm, $x^n - 1 = q(x)g(x) + r(x)$, where $\deg r(x) < \deg g(x)$.
 - Or, $r(x) = \{(x^n - 1) - q(x)g(x)\} \text{ modulo } (x^n - 1) = -q(x)g(x)$.
 - But $-q(x)g(x) \in C$ because we are multiplying the generator polynomial by another polynomial $-q(x)$.
 - Thus, we have a codeword $r(x)$ whose degree is less than that of $g(x)$.
 - This violates the minimality of the degree of $g(x)$, unless $r(x) = 0$.
 - Which implies $x^n - 1 = q(x)g(x)$, i.e., $g(x)$ is a factor of $x^n - 1$.



Indian Institute of Technology,
Delhi

17

Ranjan Bose
Department of Electrical Engineering

We now look at the third property $g(x)$ is a factor of $x^n - 1$. So, by division algorithm, $x^n - 1$ should be written as $q(x)g(x) + r(x)$, where the degree of this remainder polynomial should be less than the degree of $g(x)$ right. But you can again take it on the other side and $r(x)$ can be written as $x^n - 1 - q(x)g(x)$ of x again modulo $x^n - 1$. But if you take it comes out to be $-q(x)g(x)$. But please note $q(x)$ is a polynomial $-q(x)$ is also a polynomial, so if you

multiply it with g of x , we also end up getting another valid codeword polynomial, hence it is an element of C .

But then we have quickly generated a codeword r of x whose degree is less than that of g of x by this right. And this violates the minimality of degree of g of x which we showed earlier unless r of x is 0, which means if you put r of x is 0 here, then x raised to n minus 1 can always be written as the product of g of x times some q of x consequently g of x must necessarily be a factor of x raised n minus 1.

And therefore, it is you factorise x raised n minus 1 and you get q x times g of x , g of x is definitely a generator polynomial of a cyclic code and so it is q of x . Because it itself is a factor of x raised n minus 1. So, we tried for one and we got the other one free.

(Refer Slide Time: 19:01)

The slide is titled "Generator Polynomial" in a large blue font. Above the title, in a smaller font, is "Information Theory, Coding and Cryptography". Below the title, there are three bullet points:

- **Note 1:** A cyclic code C may contain polynomials other than the generator polynomial which also generate C .
- But the polynomial with the minimum degree is called the generator polynomial.
- **Note 2:** The degree of $g(x)$ is $n - k$

At the bottom of the slide, there is a footer with the Indian Institute of Technology Delhi logo on the left, the number "18" in the center, and "Ranjan Bose Department of Electrical Engineering" on the right.

So, a cyclic code C may contain polynomials other than the generator polynomial which can also generate C . So, but the polynomial with the minimum degree is called the generator polynomial. This we had mentioned earlier. And the degree of g of x is n minus k . And therefore, when you multiply with another polynomial of degree k minus 1, you get up a resultant polynomial with a degree n minus 1. So, if you look at it, we explicitly we can write it.

(Refer Slide Time: 19:39)

$$C(x) = a(x) \cdot g(x)$$

\downarrow \downarrow \downarrow
 deg (n-1) (k-1) (n-k)

$$C_0 + C_1x + C_2x^2 + \dots + C_{n-1}x^{n-1}$$

$$g_0 + g_1x + g_2x^2 + \dots + g_{n-k}x^{n-k}$$

$C_0, C_1, C_2, \dots, C_{n-1}$

$\leftarrow n \rightarrow$

ETSC, IIT DELHI

So, we say that any codeword polynomial C of x is nothing but some arbitrary polynomial a of x times g of x . But please note that this is a codeword polynomial, so its degree is n minus 1 . Because the degree n minus 1 states that I have got C_0 plus plus C_1x plus C_2x^2 plus so and so forth to $C_{n-1}x^{n-1}$. So, I have from here C_0, C_1, C_2 up to C_{n-1} . So, I have a linear block code of block length n . So, degree should be n minus 1 .

Now, if you look at this polynomial the degree here should be k minus 1 , we just now saw. And here in order to ensure that the degree here is n minus 1 , here the degree should be n minus k . Therefore, the product of a polynomial with degree k minus 1 and degree n minus k , so this would add up and you will get a polynomial of degree n minus 1 . So, consequently we should be able to write in general g_0 plus g_1x plus g_2x^2 and so and so forth till $g_{n-k}x^{n-k}$. So, we come back towards slides and we note that degree of g of x is n minus k .


(Refer Slide Time: 21:46)

Information Theory, Coding and Cryptography

Example

- To find all the binary codes of blocklength 3, we first factorize $x^3 - 1$.
- Note that for $GF(2)$, $1 = -1$, since $1 + 1 = 0$.
- Hence, $x^3 - 1 = x^3 + 1 = (x + 1)(x^2 + x + 1)$.

Generator Polynomial	Code (polynomial)	Code (binary)
1	$\{R_3\}$	$\{000, 001, 010, 011, 100, 101, 110, 111\}$
$(x + 1)$	$\{0, x + 1, x^2 + x, x^2 + 1\}$	$\{000, 011, 110, 101\}$
$(x^2 + x + 1)$	$\{0, x^2 + x + 1\}$	$\{000, 111\}$
$(x^3 + 1) = 0$	$\{0\}$	$\{000\}$



*Indian Institute of Technology,
Delhi*

19

*Ranjan Bose
Department of Electrical Engineering*

Let us do a quick example to understand what we did so far. Suppose we have to find all binary codes of block length 3. So, block length 3 means n is equal to 3. So, we have to factorise x raised to power n minus 1 where n is equal to 3 and which Galois field should we consider binary code means $GF(2)$. So, two things are important, the block length must be fixed, otherwise we do not know what x raised to n what does n mean. And once we factorise we definitely must know where we do our math, so we need to know the Galois field. So, given the block length and the Galois field we are ready to go. So, we have to factorise $x^3 - 1$ over $GF(2)$.

So, if you factorise it, you get these 3 generator polynomials. These are the 3 factors that you can write out. Because you can write $x^3 - 1$ is equal to $x^3 + 1$ simply because over $GF(2)$ $1 + 1 = 0$, so 1 is equal to -1 all minuses minus signs can be replaced by plus signs. So, $x^3 + 1$ is equal to $(x + 1)(x^2 + x + 1)$. So, it is actually 1 is always a factor $(x + 1)(x^2 + x + 1)$. So, I have got 1.

Well, 1 is also a valid generator polynomial because it is a factor of $x^3 - 1$, 1 is a factor of everything. But it gives a trivial code it gives really 000001 and so and so forth. If you must be appears to you can verify that this is a linear code, and it is also a cyclic code any cyclic shift is a valid code. But this is really a trivial example it has no interest in properties.

But then you look at factor x minus 1, this 1 leads to the code 0. So, one of the polynomials if you look at x minus 1 it will give you these four codes polynomial code words. And if you have x square plus x plus 1, you have got this two, so this is a reputation code. So, x square plus x plus 1 is a generator polynomial for a reputation code of block 3. Now x cube plus 1 it itself is a factor. And so, this again gives a useless code.

(Refer Slide Time: 24:40)

Information Theory, Coding and Cryptography


Using $g(x)$

- A simple encoding rule to generate the codewords from the generator polynomial is

$$c(x) = i(x)g(x),$$
- where $i(x)$ is the information polynomial, $c(x)$ is the codeword polynomial and $g(x)$ is the generator polynomial.
- We have seen already that there is a one to one correspondence between a word (vector) and a polynomial.
- The error vector can be also represented as the error polynomial, $e(x)$.
- Thus, the received word at the receiver, after passing through a noisy channel can be expressed as

$$v(x) = c(x) + e(x).$$

20



*Indian Institute of Technology,
Delhi*

*Ranjan Bose
Department of Electrical Engineering*

So, let us see how we use g of x . So, as we have mentioned any codeword polynomial can be generated using an information polynomial i of x and a g of x . Now, since any vector can be effectively represented as a polynomial the error vector can also be represented as an error polynomial. So, when we send our valid codeword through an error prone channel, we receive sometimes a received vector which is not the same as c of x . In fact, we receive v of x as c of x plus an error vector e of x . So, any sequence of bit is which can be represented as an error sequence can be represented also as an error polynomial.

(Refer Slide Time: 25:42)


Information Theory, Coding and Cryptography

Syndrome Polynomial

- We define the **Syndrome Polynomial**, $s(x)$ as the remainder of $v(x)$ under division by $g(x)$, i.e.,

$$\begin{aligned} s(x) &= R_{g(x)}[v(x)] = R_{g(x)}[c(x) + e(x)] \\ &= R_{g(x)}[c(x)] + R_{g(x)}[e(x)] = R_{g(x)}[e(x)]. \end{aligned}$$

- This is because $R_{g(x)}[c(x)] = 0$.

 Indian Institute of Technology, Delhi 21 Ranjan Bose
Department of Electrical Engineering

Now, we introduce the syndrome polynomial. Please note syndrome decoding was very much a part of linear block codes and here we have just the parallel definition. So, we define the syndrome polynomial s of x as a remainder of nu x under the division by g of x . What does it mean, if you have s of x as a syndrome polynomial, you take the received vector nu of x , and take divide by g of x whatever is the remainder is the syndrome polynomial.

And if you can write it explicitly, it is R g of x which is you taking the remainder with respect to g of x , and what is v of x , you send the valid codeword polynomial c of x plus e of x . But it can be represented using this properties of residues as R g c of x plus R g e of x . But we know that whenever we divide a valid codeword polynomial by a generated polynomial you get 0 ok. It is a perfect multiple that is how you generate a c of x . So, this first term goes out of the window and what you get is R g of x e of x .


(Refer Slide Time: 27:05)

Information Theory, Coding and Cryptography

Example

- Consider the generator polynomial $g(x) = x^2 + 1$ for **ternary** cyclic codes (i.e., over $GF(3)$)
- Let the **blocklength** $n = 4$.
- Here we are dealing with cyclic codes, therefore, the highest power of
- Since $n = 4$, k must be 2 since $g(x)$ is $n - k$.
- So, we are going to construct a (4, 2) cyclic ternary code.
- There will be a total of $q^k = 3^2 = 9$ codewords.

🔍 📄

 Indian Institute of Technology, Delhi 22 Ranjan Bose
Department of Electrical Engineering

So, let us look at a quick example, consider a generator polynomial g of x is equal to x square plus 1 for a ternary cyclic code over $GF(3)$. So, first thing that we need to verify is whether this g of x is indeed a factor of x raised to power n minus 1 over $GF(3)$. But for that I need to know the block length n is equal to 4. So, first thing we need to do is can we actually factorise x raised to a 4 minus 1 over $GF(3)$. And we should be able to get g of x is equal to x square plus 1 as one of the valid factors alright.

So, well it really is quite obvious because I can take x raised to power 4 minus 1 equal to x square plus 1 into x squared minus 1. So, clearly x raised to a 2 plus 1 is indeed a factor. So, we have cyclic codes here n is equal to 4, the k must be 2 because the degree of g of x is n minus k . So, n minus k is 2, n is 4, so k must indeed be 2. So, what we are trying to look at is a 4 comma 2 cyclic ternary code ok. This is what we have set out to do to construct a 4 comma 2 cyclic ternary code with this generator polynomial g of x given as x squared plus 1.

Now, if it is ternary and k is equal to 2 then there are total of q raised to power k valid codewords q is 3 to k is 2, so there are 9 codewords in our lookup table this is what we were looking at.

(Refer Slide Time: 29:14)

Information Theory, Coding and Cryptography

Example

- Ternary cyclic code constructed using $g(x) = x^2 + 1$.

i	i(x)	c(x) = i(x)g(x)	c
00	0	0	0000
01	1	$x^2 + 1$	0101
02	2	$2x^2 + 2$	0202
10	x	$x^3 + x$	1010
11	$x + 1$	$x^3 + x^2 + x + 1$	1111
12	$x + 2$	$x^3 + 2x^2 + x + 2$	1212
20	2x	$2x^3 + 2x$	2020
21	$2x + 1$	$2x^3 + x^2 + 2x + 1$	2121
22	$2x + 2$	$2x^3 + 2x^2 + 2x + 2$	2222

Indian Institute of Technology, Delhi
23
Ranjan Bose
Department of Electrical Engineering

So, if you write at the table using the generator polynomial, you take g of x and take all possible i x , which is information polynomials of length k . So, i is equal to 0 1 2 alright, because it is a ternary x , x plus 1, x plus 2, $2x$, $2x$ plus 1, $2x$ plus 2 this is a all possible polynomials of degree k minus 1 or less. What is k , k is 2, k minus 1 is 1, so degree 1 or less. So, this is all I can do alright. So, explicitly if you want to see you will looking at your i of x .

(Refer Slide Time: 30:10)

$(4, 2)$

$GF(3)$
 $k=2$

$i(x) \rightarrow (k-i)$ $n=4$

$()x + ()$

$\begin{matrix} & \downarrow & & \downarrow & & \downarrow & & \downarrow \\ 0 & 1 & 2 & 0 & 1 & 2 \end{matrix}$

9 information polynomials.

↓ LUT

9 codewords

0000
 0101
 0202
 ...
 2121
 2222

ETSC, IIT DELHI

And i of x can have degree k minus one. And therefore, I have got 2 places to fill. And this first one can be 0, 1 or 2 and the second place can also be 0, 1 or 2. And therefore, we have 9 polynomials, and this would correspond to 9 codewords. These are clearly information polynomials and this is the lookup table.

So, if you go back you have everything starting from 0 0, 0 1. Please remember we have k equal to 2 and we are working at $G F 3$. So, we have 0 0, 0 1, 0 2 and we can go right up to 2 1 and 2 2. And on the other hand, when we have the codewords c , there we look at 0 0 0 0 because n is equal to 4, and we are looking at 4 comma 2 cyclic code. So, you get 0 1 0 1 and you go right up to 2 1 2 1 and 2 2 2 2.

So, we come back to our slide and look at the entire table and we make a very interesting observation. This cyclic code has done nothing, but repeated the first two symbols one more time. So, 0 0 becomes 0 0 0 0, 0 1 becomes 0 1 0 1, 1 2 becomes 1 2 1 2. So, it is in some sense some kind of a repetition code just repeating the entire block information block one more time. Looks to be pretty trivial, but we have to see whether it is a useful code or not, or whether it is just a toy example.

(Refer Slide Time: 32:50)

Information Theory, Coding and Cryptography

Example

- Ternary cyclic code constructed using $g(x) = x^2 + 1$.

i	$i(x)$	$c(x) = i(x)g(x)$	c
00	0	0	0000
01	1	$x^2 + 1$	0101
02	2	$2x^2 + 2$	0202
10	x	$x^3 + x$	1010
11	$x + 1$	$x^3 + x^2 + x + 1$	1111
12	$x + 2$	$x^3 + 2x^2 + x + 2$	1212
20	$2x$	$2x^3 + 2x$	2020
			2121
			2222

▪ It can be seen that the cyclic shift of any codeword results in another valid codeword.

23

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

But first note that it is a valid cyclic code sum of any two codewords is another valid codeword you can do that math it is over $G F 3$, so modulo 3 arithmetic can work. So, if I take for example, 0 2 0 2 and add it up to 1 0 1 0 I get 1 2 1 2 which is present right here, and if I can take 2 1 2 1 add it up to 2 2 2 2, 2 plus 2 you can do a math for 1. So, 1 0 1 0


which is another valid codeword right here; so, you can check that sum of any two codeword is a another valid codeword all 0 codeword is present. So, it is a definitely a linear block code, but most interestingly any cyclic shift of a valid codeword is another valid codeword. So, it is indeed a cyclic code that has been generated using this generate a polynomial $x^2 + 1$.

(Refer Slide Time: 33:51)

Information Theory, Coding and Cryptography

Example

- By observing the codewords we find that the minimum distance of this code is 2 (there are four non-zero codewords with the minimum Hamming weight = 2).
- Therefore, this code is capable of **detecting one error** and **correcting zero errors**.
- Observing the fact that the codeword polynomial is divisible by the generator polynomial, we can *detect* more number of errors than suggested by the minimum distance of the code.



Indian Institute of Technology,
Delhi

24

Ranjan Bose
Department of Electrical Engineering

So, you continue with this example and 2 are dismay we find that the Hamming weight is 2. See hamming distance and hamming weight should be the same because it is a linear block code, so I just pick up these 2 candidates 0 1 0 1 or 0 2 0 2; 2 non-zero elements. So, weight is 2. So, clearly your hamming weight for this minimum weight for this linear block code which is cyclic is 2.

So, these are equal to 2 means at best it can detect one error and correct zero errors. So, is it useful. So, we make another observation, we observe that the codeword polynomial if it is valid should be divisible by the generator polynomial. Say for possibly we can detect more number of errors then suggested by a minimum distance calculation, how is that.


(Refer Slide Time: 34:59)

Information Theory, Coding and Cryptography

Example

- Assume that $g(x) = x^2 + 1$ and the transmitted codeword is the all zero codeword.
- Therefore, the received word is the error polynomial, i.e.,
$$v(x) = c(x) + e(x) = e(x).$$
- At the receiver end, an error will be detected if $g(x)$ fails to divide the received word $v(x) = e(x)$.
- Now, $g(x)$ has only two terms.
- **So if the $e(x)$ has odd number of terms, i.e., of the number of errors are odd, it will be caught by the decoder!**

25

 Indian Institute of Technology, Delhi

Ranjan Bose
Department of Electrical Engineering

So, assume that g of x is being used as a generator polynomial and without losing any without loss of generality we can say that the transmitted codeword is in all zero codeword. It is a linear block code. So, you can always do an example with an all zero codeword and generalize it.

So, the received word is nothing but the error polynomial because c of x is 0 we are sending the all zero codeword. So, the received vector is the e of x . Now in order to detect where please note at talking about the detection problem not the correction problem we want to detect whether there is an error or not. So, detection will require us to divide the received vector by g of x , that is to divide e of x by g of x .

Now if it is a 0 then we get the conclusion as it is a valid codeword even though e of x is an error or if we look at the e of x carefully if it has odd number of terms with no matter what you do division by g of x which is only two terms, will leave behind one term. It is just basic division in order to cancel out all the terms of e of x right. We should have an even number because each time g of x when I divide I cancel out I try to cancel or two of them.

So, if you have e of x containing odd number of terms, I will always end up with a non zero answer. Regardless of how many such terms are, but please note each term corresponds to one error, so we are actually going to possibly detect more number of errors than one as suggested by the minimum distance.


(Refer Slide Time: 37:10)

Information Theory, Coding and Cryptography

Example

- For example, if we try to divide $e(x) = x^3 + x + 1$ by $g(x)$, we will always get a remainder.
- In the case of the (4, 2) cyclic code with $g(x) = x^2 + 1$, the $d^* = 2$, suggesting that it can detect $d^* - 1 = 1$ error.
- However, by this simple observation we find that it can **detect** any odd number of errors $\leq n$.
- **In this case it can detect 1 error or 3 errors, but not 2 errors.**
- Cyclic codes are very powerful for detecting errors !

26

 Indian Institute of Technology,
Delhi

Ranjan Bose
Department of Electrical Engineering

Let us carry this example forward. So, suppose we have 3 errors, so the error polynomial looks like $x^3 + x + 1$, it means that the error is at location 1 location 2 and location 4. Please note it is a (4, 2) codeword, so there are 4 locations for error to happen because the codeword is of the length 4 and is equal to 4. So, here we have 1 1 0 1 is the polynomial responding to this error vector.

So, we have the cyclic code $g(x)$ is equal to $x^2 + 1$ and d^* is equal to 2, so even though theoretically it seems we can only detect 1 error, if you try dividing $e(x)$ by $g(x)$ we will get a remainder a non zero answer and consequently we will raise a flag and say that yes indeed we have detected an error.

So, the theory says that not more than 1 sure 2 errors can never be detected, but 3 errors yes it can detect, 4 errors probably not. So, this gives us a glimpse as to how strong we can make our error control codes in terms of using cyclic codes in terms of detecting errors. So, again we are talking about detecting not correct, so they can be used for checks most likely we call them cyclic redundancy check. So, CRC codes are very popular in detecting errors.

(Refer Slide Time: 38:56)


Information Theory, Coding and Cryptography

Matrix Representation

- Suppose C is a cyclic code with generator polynomial $g(x) = g_0 + g_1x + \dots + g_rx^r$ of degree r .
- Then the generator matrix of C is given by

$$G = \begin{bmatrix} g_0 & g_1 & \dots & g_r & 0 & 0 & 0 & \dots & 0 \\ 0 & g_0 & g_1 & \dots & g_r & 0 & 0 & \dots & 0 \\ 0 & 0 & g_0 & g_1 & \dots & g_r & 0 & \dots & 0 \\ \vdots & \vdots & & & & & & \vdots & \\ 0 & 0 & 0 & 0 & 0 & g_0 & g_1 & \dots & g_r \end{bmatrix}$$

$\xleftarrow{\hspace{10em}} n \text{ columns} \xrightarrow{\hspace{1em}}$
 $\updownarrow k = (n - r) \text{ rows}$

 Indian Institute of Technology, Delhi
27
Ranjan Bose
Department of Electrical Engineering

So, we now shift gears slightly and we come back to a original flavour of linear block codes. If cyclic codes are indeed subclass of linear block codes then we should be able to represent them in terms of a generator matrix. So, we have a standard way to represent a generator matrix for a cyclic code. Please note the structure. So, g_0, g_1 up to g_r and so and so forth with 1 cyclic shift and so and so forth and we have k rows and n columns.


(Refer Slide Time: 39:39)

Information Theory, Coding and Cryptography

Matrix Representation

- The $(n - r)$ rows of the matrix are linearly independent because of the **echelon form of the matrix**.
- These $(n - r)$ rows represent the codewords $g(x), xg(x), x^2g(x), \dots, x^{n-r-1}g(x)$.
- Thus, the matrix can generate these codewords.
- Now, to prove that the matrix can generate *all* the possible codewords, we must show that every possible codeword can be represented as linear combinations of the codewords $g(x), xg(x), x^2g(x), \dots, x^{n-r-1}g(x)$.
- We know that if $c(x)$ is a codeword, it can be represented as

$$c(x) = q(x)g(x) \text{ for some polynomial } q(x).$$

 Indian Institute of Technology, Delhi
28
Ranjan Bose
Department of Electrical Engineering

So, please note that the n minus r rows of the matrix are linearly independent because of the echelon form of the matrix please note that. So, it is a simple observation, so they are

linearly independent. And these code words are actually the rows are actually the codeword. How is that well g of x is always a valid codeword polynomials, so the corresponding vector which is $g_0, g_1, 0$ up to n . So, the length is n , but the degree of g of x is n minus k . So, their r goes up to n minus k only.

So, we have the first one is a valid codeword, the second one is 1 cyclic shift, third one is another factory shift up to k cyclic shifts and that constitutes a entire generator matrix. So, we can write that g of x valid x g of x another valid codeword x square g of x of course, as a cyclic shift by 2 and so and so forth, and sum of any 2 of them each 1 is a valid codeword is also valid codeword. So, indeed g of x and all of it is saltlick shifted versions which form the rows can generate the entire code space. So, indeed we have g c of x is q of x time g of x for some polynomial q of x .

(Refer Slide Time: 41:28)


Information Theory, Coding and Cryptography

Matrix Representation

- Since the degree of $c(x) < n$ (because the length of the codeword is n), it follows that the degree of $q(x) < n - r$.
- Hence,

$$q(x) \cdot g(x) = (q_0 + q_1x + \dots + q_{n-r-1}x^{n-r-1})g(x)$$

$$= q_0g(x) + q_1xg(x) + \dots + q_{n-r-1}x^{n-r-1}g(x).$$
- Thus, any codeword can be represented as a linear combination of $g(x), xg(x), x^2g(x), \dots, x^{n-r-1}g(x)$.
- This proves that the matrix G is indeed the generator matrix.
- We also know that the dimensions of the generator matrix is $k \times n$.
- Therefore, $r = n - k$, i.e., the degree of $g(x)$ is $n - k$.

 Indian Institute of Technology,
Delhi

29

Ranjan Bose
Department of Electrical Engineering

So, we are explicitly writing that any polynomial times g of x can be written as sum of shifted codewords. And therefore, you can prove that the matrix G is indeed a generator matrix just because the rows are valid codeword polynomials. And we know that the dimension of the generator matrix is k cross n ; and the degree of g of x is n minus k as we established.

(Refer Slide Time: 42:07)

Information Theory, Coding and Cryptography

Summary

- Cyclic Codes
- Generator Polynomial
- Syndrome Polynomial
- Matrix Representation
- Examples

Indian Institute of Technology, Delhi 30 Ranjan Bose
Department of Electrical Engineering

So, with that we come to the end of this lecture. So, let us quickly summarise what we have read so far and studied. So, we have looked at cyclic codes in more detail. We have introduced the interesting concept of a generator polynomial what should be its degree, why it generates the entire code space. And then we looked at syndrome polynomial and why it is possible to detect more number of errors than predicted by the d^* philosophy.

And finally, we looked at the matrix representation and this is where we will continue in the subsequent lecture. We also looked at some examples to show that yes it can indeed do better than the error detection predicted by d^* . With that we come to the end of this lecture.