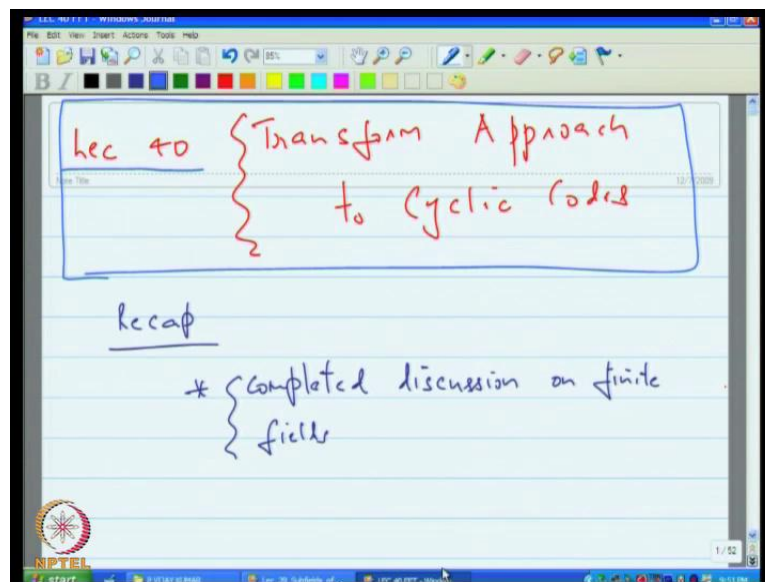


Error Correction Codes
Prof. Dr. P. Vijay Kumar
Department of Electrical Communication Engineering
Indian Institute of Science, Bangalore

Lecture No. # 40
Transform Approach to Cyclic Codes

Here, I am going to say good morning today, because I am taping this lecture in the morning; usually I tape in the afternoon. So, there is a little bit of a change here. So, this is our forty eighth lecture, and the lecture series is supposed to run to about 40 lectures, but I might go 1 or 2 lectures over just to make sure that I complete the material. So, the title of today's lecture is a transform approach to cyclic codes.

(Refer Slide Time: 00:43)



So, what that means is that, we worked very hard in trying to understand finite fields, because that is the mathematical background you need to be able to understand cyclic codes. And now finally, we have reached that point where we are ready to start studying the codes themselves.

(Refer Slide Time: 01:05)

Lec 39 Subfields of a finite field

Defn $\mathbb{F}_p^* = \mathbb{F}_p \setminus \{0\}$ (set of all nonzero elements)

Lemma Let $p \in \mathbb{F}_p^*$, $q \in \mathbb{F}_p^*$. Then $\deg(\text{m.p.}(p)) \leq m$.

Proof (consider $1, p, p^2, \dots, p^{m-1}$ elements)

These cannot all be linearly independent for this would mean that the m elements $\sum_{i=0}^{m-1} a_i p^i = 0$, $a_i \in \mathbb{F}_p$ are all 0, which is impossible since $1 = p^0$. Thus there exists a dependence expression of the form:

$\sum_{i=0}^{m-1} c_i p^i = 0$ with at least one $c_i \neq 0$

$\Rightarrow p$ is a root of $\sum_{i=0}^{m-1} c_i x^i$

Hence $\text{m.p.}(p) \mid \sum_{i=0}^{m-1} c_i x^i$

$\Rightarrow \deg(\text{m.p.}(p)) \leq m$

But, as usual, I will begin with a quick overview of what we discussed last time. So, here is our lecture from last time and the first thing we actually did was we finished completed our discussion on the minimal polynomial, where we showed that minimal polynomial has degree, where we discussed the degree of the minimal polynomial, and noticed that it has the degree m , only when it is the minimal polynomial of a primitive element.

(Refer Slide Time: 00:47)

*** subfields**

*** $(x+y)^p = x^p + y^p$**

*** $\theta \in \mathbb{F}_{p^d} \Leftrightarrow \theta^{p^d} = \theta$**

*** finite fields of every size p^m exist**

*** $x^{p^m} - x = \prod_{d|m} f(x)$**
 $\deg f = d, f \text{ irreducible}$

Our interest is in cyclic

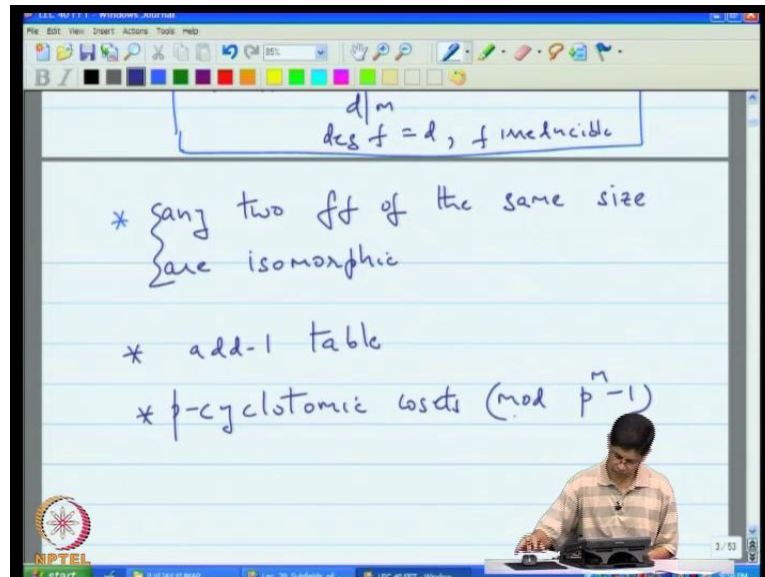
So, let me as I do this, let me put down the summary. So, we completed our discussion, then the next topic that we moved on to on finite fields, so we did a little bit of jumping around in the last lecture. We talked about subfields of a finite field, that is and this picture over here, gives you an excellent overview. It tells you that the subfield's structure is completely determined by this exponent over here.

So, here you have 2^{12} , and so the subfields that it contains are of the form 2^m , where m divides 12. And the other thing that we did was along, as an aside we also discussed a useful lemma, which says that if you take $x + y$ and raise it to the p th power, you get $x^p + y^p$. So, then we say subfield structure; then we did this lemma; then we also tested carried out a test for membership in a subfield, which is that for example, if $\theta^p = \theta$, then it belongs to F_p .

And then we, I sketched very briefly, the proof that, finite fields of every size p^m exists, for every prime p and every m greater than or equal to 1; and, what I did was, because I wanted to make sure that I do not get bogged down in discussing finite fields, I put the detailed proofs in the appendix. So, as you go through your notes, please look through the notes; you will find that whatever theorem we have stated here, is again stated in the appendix, and followed by a complete proof. Then I actually stated a useful lemma, a theorem, which says that, if you take the, if you are looking at particular finite field $x^p = x$ in F_{p^m} , then the polynomial $x^p - x$, factors into the product of the irreducible polynomials, whose degree divides m .

So, we will put that down also. So, that was our next observation and after that, we showed that, any two finite fields of the same size are isomorphic. What isomorphic means is that, basically, the two fields are the same, except that, what you call an element here, maybe different.

(Refer Slide Time: 06:09)



For example, you might call an element beta here, and in that other field, it might appear as alpha cubed; but provided, you recognize that, beta is the same as alpha cubed and you follow through, then you will find, the fields are exactly the same; they add in the same way; they multiply in the same way.

So, I will put that down. Then after that, we discussed the add 1 table of a finite field; and what the add 1 table allows you to do, it basically allows you to work with the elements, with the representation of the elements of the finite field in exponent form. So, you can regard the finite field as powers of some primitive element alpha, and just learn how to add and multiply in that domain; and, that makes it much easier. We also discussed the add 1 table. Then we talked about cyclotomic cosets and so I introduced the definition of cyclotomic cosets, and I think that, it is best illustrated by this example here. So, these are the 2 cyclotomic cosets, mod 15. So, basically, anything in which a is 2 to the, some power times b, then they are equivalent.

So, 1, 2, 4, 8, 16 mod 15 is back to 1. So, these are the cyclotomic cosets. They are basically equivalence classes under the corresponding equivalence relationship. And then we also talked about the coset leaders, the smallest element in a cyclotomic coset is called the coset leader.

(Refer Slide Time: 08:22)

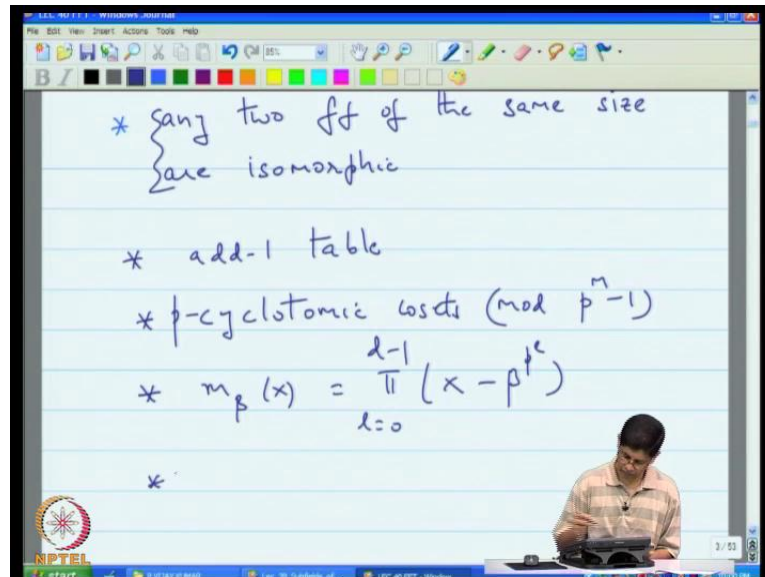
Thm. Let $\beta \in F_{p^m}^*$. Then

$$m_\beta(x) = \prod_{l=0}^{d-1} (x - \beta^{p^l})$$

where d is the smallest integer such that $\beta^{p^d} = \beta$, i.e., F_{p^d} is the smallest subfield in F_{p^m} which β is an element.

Then, there is a theorem and we will not actually prove it here, which says that, the minimal polynomial of an element β , is precisely, the product of all linear factors of the form x minus β to the p to the l , where l goes from 0 to d minus 1 . So, what is this d ? Basically, what this d here is, is that d is the smallest integer such that, β to the p to the d is β ; that is F , the finite field of size p to the d is the smallest subfield in F p to the m of which β is an element; proof is in the appendix.

(Refer Slide Time: 09:06)



* any two fts of the same size
are isomorphic

* add-1 table

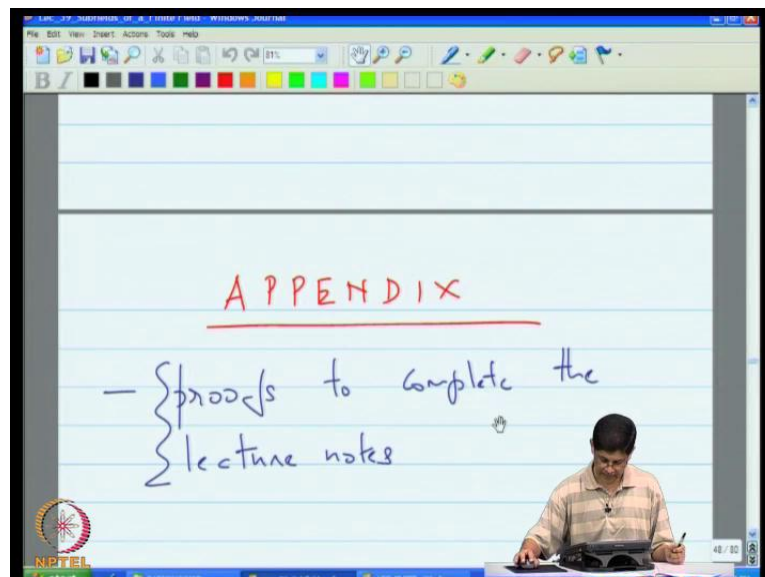
* p -cyclotomic cosets (mod p^n-1)

* $m_p(x) = \prod_{l=0}^{d-1} (x - \beta^{p^l})$

*

So, let us put that down, and the, the powers of beta of the form beta to the p to the l are called the conjugates of beta. So, what this theorem also tells us is that the **the** conjugates of beta also share the same minimal polynomial, because they are zeroes of the same polynomial. And after this you have the appendix.

(Refer Slide Time: 09:50)

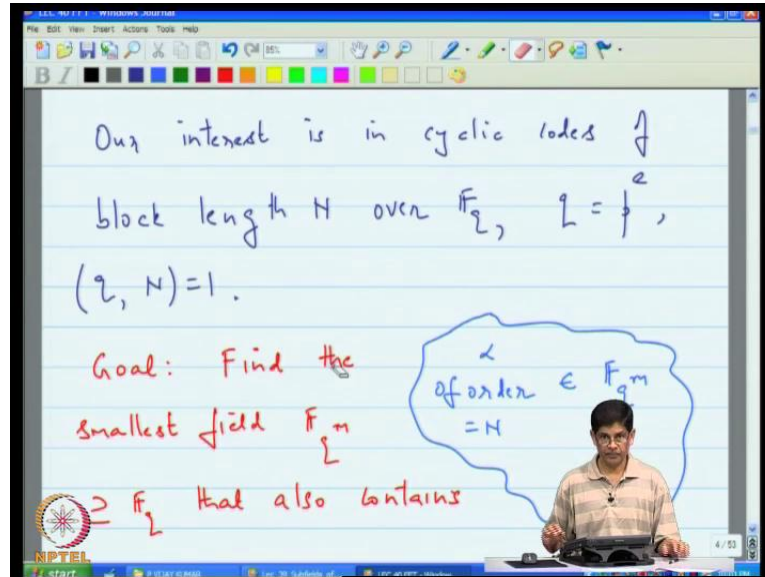


APPENDIX

- {proofs to complete the
lecture notes}

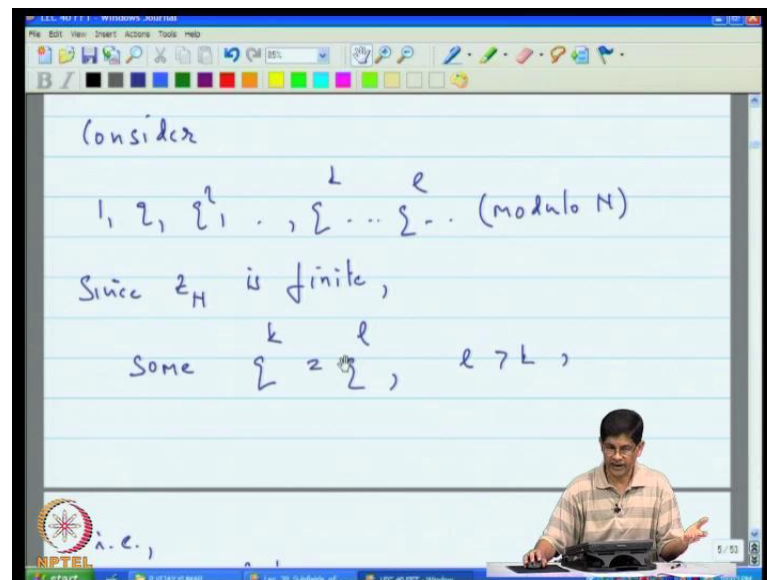
That basically was the last lecture. Now, getting back, so that I guess concludes our summary. I did want to break up the previous lecture into points (()) to rather lengthy lecture. Now, let us move on to today's lecture, and here, our interest is really in, is in cyclic codes.

(Refer Slide Time: 10:17)



So, remember, right in the beginning, we talked about block codes. So, our interest is back in block codes, except, we are interested in a particular class of block codes known as cyclic codes, and we will actually give a formal definition so on. And, the, the codes, now are no longer binary; that they are codes over a finite field of size q , where q is the power of p . And, there is an additional conditional that comes about, just because without this, the theory will become much harder. So, this is for the purpose of tractability, you also require the q and N be relatively prime.

(Refer Slide Time: 12:12)

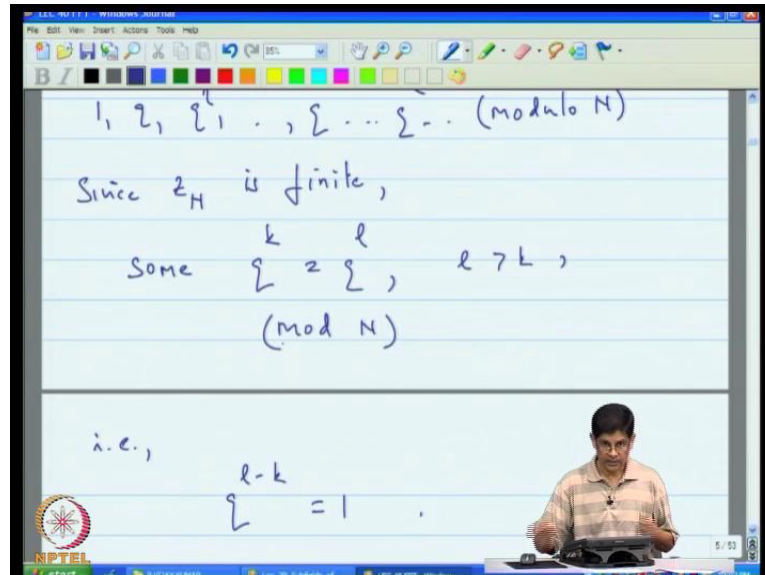


So, our first goal and the reason for this will become clearer a little later, is to find the smallest field. So, you are starting off with a finite field F_q ; that is the field over which you want to build your code. So, it is not, if it is binary, q would be equal to 2. But it may not be binary. For example, the class of Reed-Solomon codes; these codes are not binary, and in fact, the non-binary nature of those codes, gives them their strength. This, let us say, is the alphabet over which you want to build your code, for whatever reason. We will call this the ground field. It turns out that, in order to develop the theory of cyclic codes, what you need, it is a rather peculiar condition, that is, you need an element α , whose multiplicative order is N ; that is the smallest power of α which gives you 1 should be N .

Now, it could be that, this field itself contains such an element α , but often, that is not the case; in which case, what you can do is, you can actually start with the smaller field, and you can build up to a larger field. You can go up from here to here, and you can be sure that, there is a larger field that contains this field, which contains the desired element α . So, that is our goal, is to find this bigger field. And, the way you find it is quite easy, because what you do is, you say, wait a minute, what I can do is I can take 1, q , q square, q cubed, q to the k , q to the l and keep going; and, I do this modulo N . For example, if q was 2, and (()) 1, 2, 4, 8, and so on, and I would do this modulo N ; if N was 15, I would go 1, 2, 4 and 8 and stop. But in general, I would just keep going. Now, since there are only NN residues

modulo N , N at some point this series must repeat. Two elements must be the same modulo N . So, perhaps I should emphasize that, and write, this is true modulo N ; some 2 powers must agree.

(Refer Slide Time: 12:56)



Now, without loss of generality, we can assume that, l is greater than k ; so that is equivalent to saying that, q to the l minus k is equal to 1; again **again**, this is mod **mod** N . So, then now that we know this, it is meaningful to ask this question, what is the smallest power of q , such that, q to the m is equal to 1 mod N .

(Refer Slide Time: 13:16)

i.e.,

$$q^{l-k} = 1 \pmod{N}.$$

Let m be the smallest power of q

s.t. $q^m = 1 \pmod{N}$, m is then

called the multiplicative order of q

\pmod{N} .

The video frame shows a digital whiteboard interface with a toolbar at the top. In the bottom right corner, a person is visible, and a small 'NPTEL' logo is in the bottom left corner of the whiteboard area.

Once you found the smallest integer m , then that particular m is called the multiplicative order of $q \pmod{N}$. It sounds a little bit mysterious, why would you keep doing this? The reason is, the focus is on this equation, over here. This equation is key; because what you are really going to do is, you are going to exploit this equation. What this, remember, our hunt is for this element α , whose multiplicative order is N ; so that is what we are actually hunting for.

(Refer Slide Time: 14:14)

Note: this implies that

$$q^m \equiv 1 \pmod{N}$$

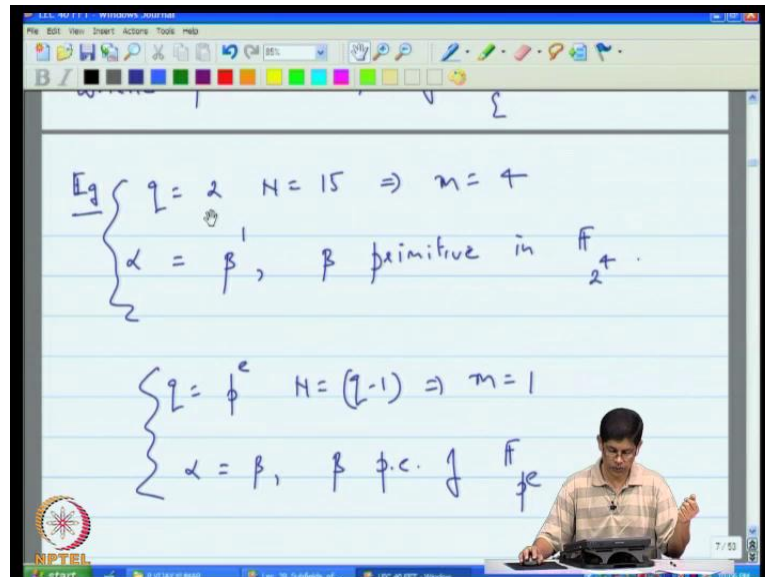
$\Rightarrow N \mid (q^m - 1)$

$\Rightarrow F_q^m$ contains an element of order N , for example, $\alpha = \beta^{\frac{q^m - 1}{N}}$, where β is a p.e. of F_q^m

And, what we have here, is this expression; but that, this expression is telling us that, N divides q to the m minus 1, and that is important, because it tells us that, in the finite field F_q to the m , and we know that, finite fields of every size of this type exist, because q itself is some power of a prime. So, q to the m is also some power of a prime, and therefore, a finite field of this size exists. So, this finite field contains an element of order N , why is that? Well, because you know, let us say, let β be a primitive element of this finite field; then the order of β is $q^m - 1$. So, if you raise β to the power $q^m - 1$ by N , then this element α will actually have order N .

So, that is the reason for actually working this hard. So, again, just to quickly recap, our goal was, you want to build cyclic codes over this field, which we called the ground field; but it turns out that, in order to develop the theory, you need the presence of an element α , whose multiplicative order is N .

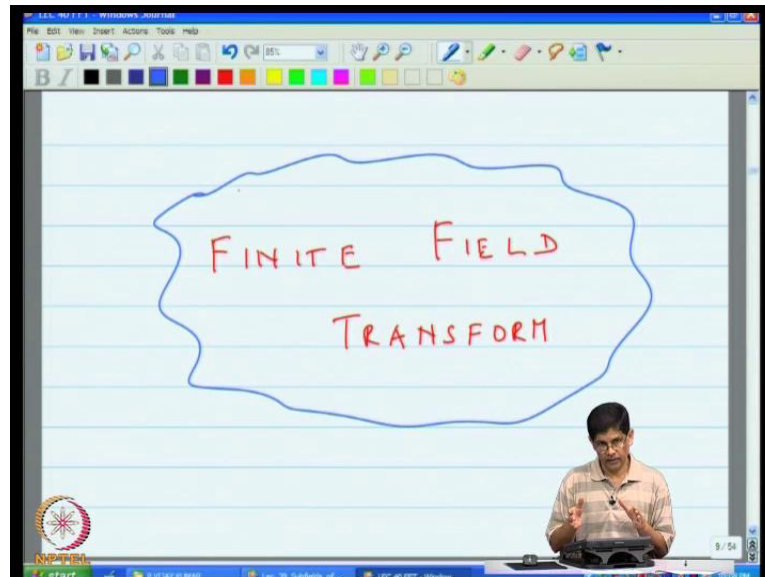
(Refer Slide Time: 15:40)



So, you are forced to enlarge the size of your field and go to a bigger field, which contains the element that you are actually looking for... Let us keep going. Example, for example, when q is 2 and N is 15, then the smallest power of 2, which will give you 1 mod 15 is 4, because 2 to the 4 is 16, which is 1 mod 15. And here, it turns out that, you can simply, you need an element of order 15 but the primitive element in F_{2^4} , is itself of order 15.

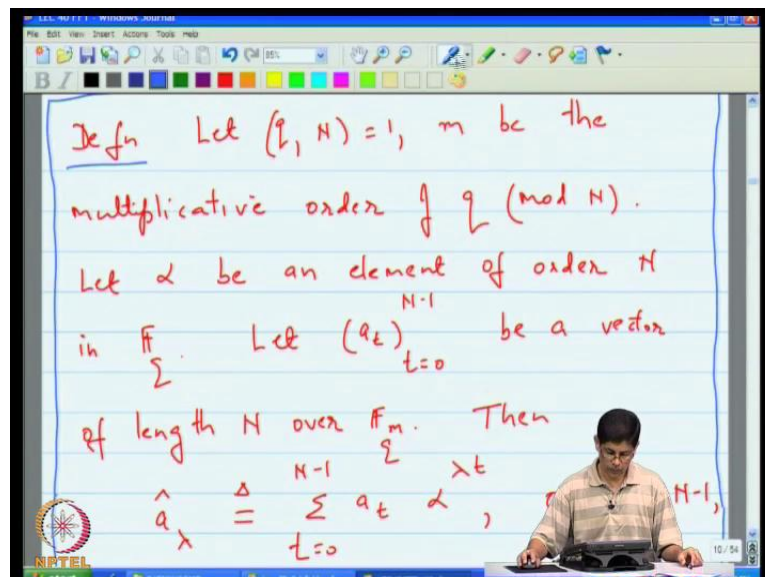
So, there, that does it for you. The second pause, the second example which often crops up with Reed-Solomon codes is that, q is a power of p and N is q minus 1. Note that, the condition that, N and q are relatively prime is satisfied, in which case, you are looking for the smallest power of q , which is equal to 1 mod N , but since q is m plus 1, that is just m equal to 1. In this case, you do not need to enlarge your field, and you can find the element α that you are looking for, within the ground field itself, and you can choose α to be β , where β is any primitive element of the ground field. So, this situation is characteristic of Reed-Solomon codes, whereas this is characteristic of BCH codes. Now, now we are actually going to discuss and perhaps it is worth highlighting this.

(Refer Slide Time: 17:15)



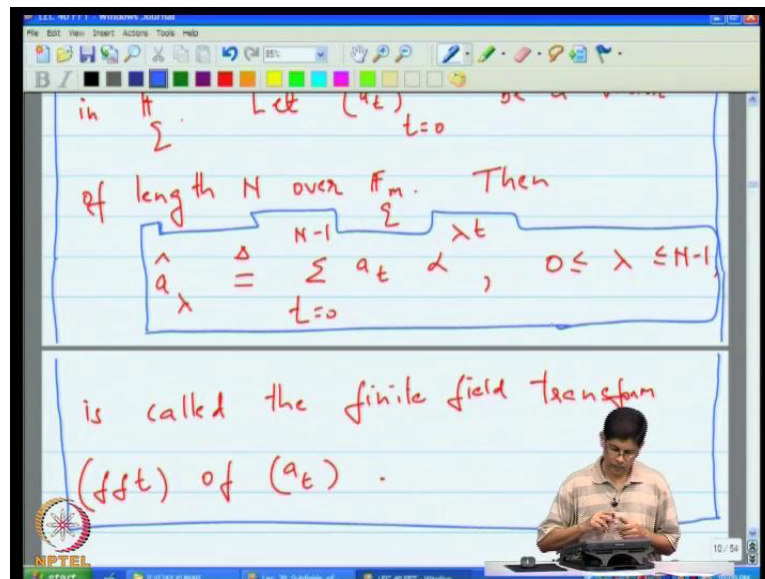
So, I will write finite field transform; so we are beginning a new topic, finite field transform. So, what in **in in** respect to this topic, what our previous topic was all about was, actually trying to find that element alpha, because in constructing the transform, we will need this alpha. So, here is the definition of a finite field transform.

(Refer Slide Time: 18:01)



Let q , let q and N be relatively prime; let m be the multiplicative order of $q \bmod N$; let α be an element of order N , in F_q ; let a_t , t goes from 0 to N minus 1, be a vector of length N over F_q to the m ; then if you set \hat{a}_λ . So, you look at this expression over here; this is called the finite field transform of this, a F_t . Now, if you look at this and think about it, there is a lot of resemblance with the discrete Fourier transform; but I leave it to you, to make that connection.

(Refer Slide Time: 18:59)



So, also, this element that we were looking for α , appears here. So, then the finite field transform is defined according to this expression. So, this is the finite field transform, and so we will be working with this expression many times in the course of this lecture. And analogous to the way you develop properties of the discrete Fourier transform or the Fourier transform, or the Laplace transform or the z transform, let us look at some properties, that this transform enjoys.

(Refer Slide Time: 19:38)

1). Linearity:

$$(a_t) \leftrightarrow (\hat{a}_\lambda) \quad (b_t) \leftrightarrow (\hat{b}_\lambda)$$

$$\Rightarrow (a_t + \theta b_t) \leftrightarrow (\hat{a}_\lambda + \theta \hat{b}_\lambda)$$

all $\theta \in \mathbb{F}_m$.

CYCLIC SHIFTS

2). Let

$$b_t = a_{(t-\tau) \bmod N} \quad 0 \leq \tau \leq N-1$$

for some $0 \leq \tau \leq N-1$.

The first property is that of linearity. What do we mean by that? We mean that, if you take a vector a_t and let us say that, it has a transform \hat{a}_λ and if b_t , it has a transform \hat{b}_λ , then if you take a_t plus θ times b_t , what is θ , where your a and b ... Now, in this discussion, a and b , all belong to the larger field; there is a ground field and there is a big field; the ground field being \mathbb{F}_q and the big field being \mathbb{F}_{q^m} . Here, all our discussion is about the big field, \mathbb{F}_{q^m} . So, a_t and b_t are sequences in the big field and θ also belongs to the big field, then the transform of this sequence here, which is a linear combination of a and b , has this transform, $(())$ a_t is replaced by \hat{a}_λ , b_t by \hat{b}_λ . So, that is linearity, and this is easy to prove. So, I will just leave this to you, to show as an exercise; it is straight forward.

(Refer Slide Time: 21:01)

CYCLIC SHIFTS

2). Let

$$b_t = a_{(t-\tau) \pmod N} \quad 0 \leq t \leq N-1$$

for some $0 \leq \tau \leq N-1$. Then

(b_t) is called a cyclic shift of (a_t) .

In the sequel, we will also assume that subscript of a

It is a simple exercise. Another property, and this property is important, because after all our discussion is about cyclic codes. So, now we want to understand, how the finite field transform behaves under cyclic shifts; meaning that supposing you have a sequence a of t and you are aware of its transform, which is \hat{a} . Now, you construct a second sequence, b which is a cyclic shift of a by τ ; that is the t th element here is t minus τ th element of a . Now, one point to note here is that, when you subtract t minus τ , you are doing this modulo N . So, that is important, that it is because without this modulo N , it would simply be a shift; but because it is mod N , it is called a cyclic shift. So, in the sequel now from now onwards, whenever we do arithmetic in the subscripts...

(Refer Slide Time: 22:12)

are always computed (mod N)
and will therefore simply write
 $(a_{t-\tau})$ in place of $(a_{t-\tau \text{ (mod } N)})$.

Now

$$\sum_{t=0}^{N-1} a_{t-\tau} \alpha^{>t}$$

So, we might say $2t$, or $3t$, or t minus τ , or $t+1$ plus $t+2$; in the subscripts, the subscript arithmetic will always be modulo N . In the sequel, we will always assume that, subscripts of either a or \hat{a} , that the arithmetic is always computed mod N , and we will simply write a of t minus τ in place of this more cumbersome expression.

(Refer Slide Time: 22:30)

Now

$$\sum_{t=0}^{N-1} a_{t-\tau} \alpha^{>t}$$

$$= \sum_{t=0}^{N-1} a_{t-\tau} \alpha^{>(t-\tau)} \alpha^{>\tau}$$

$$= \alpha^{>\tau} \left[\sum_{s=0}^{N-1} a_s \alpha^{>s} \right], \text{ setting } t-\tau = s$$

$$= \alpha^{>\tau} \hat{a}_\tau$$

Now, let us see what happens when we have a cyclic shift. So, when you take a cyclic shift and you take its transform, now what I have done is, I have replaced t by $t - \tau$; I am going to break this up into $\lambda t - \lambda \tau$ and $\lambda \tau$; I am going to pull this out, because it is not a function of t ; and then I have this expression here; and, I got from here to here, simply by replacing $t - \tau$ by s ; and again, this is mod N , reason being that, α is an element of order N . So, all exponent arithmetic is naturally mod N . Now, t ranges from 0 to $N - 1$, and **it is not** have to see that, as t ranges over 0 to $N - 1$, so does s , for any fixed value of τ , because you are doing arithmetic mod N .

(Refer Slide Time: 23:00)

$$\begin{aligned}
 & t=0 \\
 & = \sum_{t=0}^{N-1} a_{t-\tau} \alpha^{\lambda(t-\tau)} \\
 & = \alpha^{\lambda\tau} \left[\sum_{s=0}^{N-1} a_s \alpha^{\lambda s} \right], \quad \text{Setting } t-\tau = s \pmod{N} \\
 & = \alpha^{\lambda\tau} \hat{a}_\lambda.
 \end{aligned}$$

Thus $(a_k) \Leftrightarrow (\hat{a}_k)$

So, change the variables, you have this; and now, if you look at this, **this** is nothing, but of the expression for a hat of lambda. And so that tells you that, if a of t has transform a hat lambda, then its cyclic shift has transform $\alpha^{\lambda \tau}$ times a hat lambda; and this is very similar to the expression that you have, in the case of the discrete Fourier transform. There, it is probably, something like, $e^{-j 2 \pi F \tau}$ times a hat of F , but it is the same kind of relationship here. One other point to note is that, in our definition of the transform, we use the plus sign; we use the plus sign here, whereas, with Fourier-Laplace transforms, you typically encounter a negative sign; but of course, that is not a problem, as long as you are consistent, that works just as fine. It would work fine, even if

you define your discrete Fourier transform that way; it is just a matter of convention; so here we choose to use the positive sign.

(Refer Slide Time: 24:35)

3) INVERSION FORMULA:

$$a_t = (N)^{-1} \sum_{\lambda=0}^{N-1} \hat{a}_{\lambda} \alpha^{-\lambda t}$$

pf. RHS = $(N)^{-1} \sum_{\lambda=0}^{N-1} \sum_{s=0}^{N-1} a_s \alpha^{\lambda(s-t)}$

$$= (N)^{-1} \sum_{s=0}^{N-1} a_s \left[\sum_{\lambda=0}^{N-1} \alpha^{\lambda(s-t)} \right]$$

N (when $s=t$)

$$= \begin{cases} N(s-t) & -1 \\ & \end{cases}$$

Then, there is an inversion formula, as you might expect; if you can go from the time domain to the frequency domain, you would expect that you should be able to come back; that is what the inversion formula tells you, that you can invert the expression a_t by computing this expression over here.

So, this is very similar to the way you would compute it, in the case of the discrete Fourier transform. Now, I have given a proof over here, but the proof is straight forward. It is completely analogous to the discrete Fourier transform, and in this proof, you use the fact that α has multiplicative order N ; because if α did not have that property, this proof would not work. But I have said that, I think, I am going to skip this proof, so that we can keep our momentum. So, the inversion formula, very similar to the case of the discrete Fourier transform; there is that, where you had a plus in the forward transform, you have a minus now.

(Refer Slide Time: 23:13)

$$= 2^{Nt} \left[\sum_{s=0}^{N-1} a_s 2^{Ns} \right], \quad t - \tau = s \pmod{N}$$

$$= 2^{Nt} \hat{a}_\lambda$$

Thus $(a_t) \Leftrightarrow (\hat{a}_\lambda)$

$$\Rightarrow (a_{t-\tau}) \Leftrightarrow 2^{Nt} (\hat{a}_\lambda)$$

And then you have an additional scale factor. Now, this N inverse is really, you should interpret this as the inverse of N , in the finite field F_q to the m .

(Refer Slide Time: 26:02)

4). CONVOLUTION

$$(a_t) \Leftrightarrow (\hat{a}_\lambda) \quad (b_t) \Leftrightarrow (\hat{b}_\lambda)$$

$$\Rightarrow (\hat{u}_\lambda) = (\hat{a}_\lambda \hat{b}_\lambda) \quad \text{where} \quad u_t = \sum_{\tau=0}^{N-1} a_{t-\tau} b_\tau$$

Pf. (Exercise!)

Then, there is also a convolution relationship, again completely analogous with the way it is with the discrete Fourier transform. If a of t has transform \hat{a}_λ and b of t has transform \hat{b}_λ , then if you take the convolution of these two sequences, and the

convolution is given by this expression; now, once again, when I write this is a cyclic convolution, because this t minus τ is interpreted modulo N ; so perhaps, I should emphasize that. So, let us do that.

(Refer Slide Time: 26:41)

4). CYCLIC CONVOLUTION

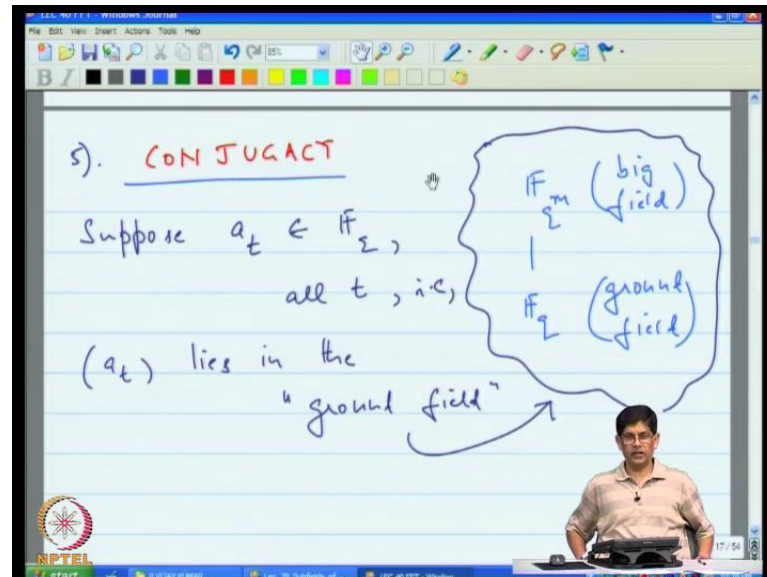
$$(a_t) \Leftrightarrow (\hat{a}_x) \quad (b_t) \Leftrightarrow (\hat{b}_x)$$

$$\Rightarrow (\hat{u}_x) = (\hat{a}_x \hat{b}_x) \quad \text{where} \quad u_t = \sum_{\tau=0}^{N-1} a_{t-\tau} b_{\tau}$$

Pf. (Exercise!)

So, let us write cyclic, so cyclic convolution. So, under cyclic convolution, what happens is that, the transform of the result is nothing, but the product of the individual transforms; and again, completely analogous to the discrete Fourier transform case. I have left the proof as an exercise. So, we have gone through four properties so far, we have seen linearity; we have seen cyclic shifts; we have seen the inversion formula and we have looked at convolution.

(Refer Slide Time: 27:29)

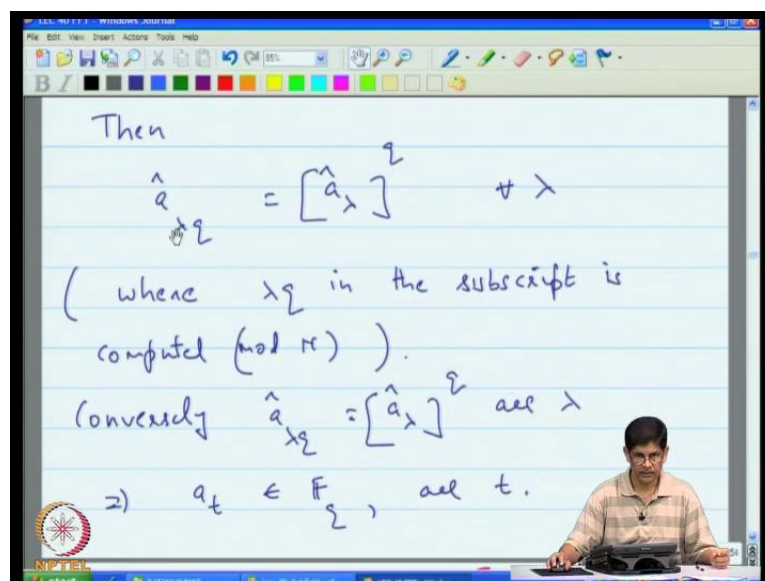


Then, there is another property, that of conjugacy. Now, conjugacy is really something that you already encountered before, in, **in** connection with the discrete Fourier transform; except that it looks a little different. Basically, what conjugacy is saying is that, look, when you take the discrete Fourier transform, you are taking you can take the transform of complex signals, just as easily as you can take the transform of a real signal, because even though your signal may be real, the discrete Fourier transform may be a complex waveform.

But for the particular case, when the function itself is real, then the transform domain, the transform of a real signal satisfies a certain conjugacy property, namely, something to the effect that, F^* of F is F of minus F ; so there is a certain relationship in the transform domain, that comes about, because your time domain signal happens to be real. So, exactly the same thing happens here; meaning that up to now, we have been working entirely in this field, which I have called the big field, that is, your, **your** vector a_t , all these symbols were in the big field, the transform $\hat{\lambda}$, all the symbols were in the big field. But now, we are going to actually investigate, well, what happens if actually my original signal a_t had symbols in here? And, in fact, in practice, that is the case we are going to consider, because our interest is primarily in cyclic codes, but the alphabet is \mathbb{F}_q as I mentioned earlier, right. So, you are really interested in here; you are forced to work with this big field, simply because you would not be able to take transforms in this field simply, because you do not

have an element of order N in the ground field; you have to go to a bigger field. And so what happens? Then, so supposing a_t belongs to F for all t ; that is this sequence lies entirely in the ground field, what can we say about that transform? So, that is the conjugacy relationship.

(Refer Slide Time: 27:38)



It turns out that, in this case, the conjugacy that results is this; if you look at the transform \hat{a} , evaluated at λ_q , then that is the same as \hat{a}_λ , but raised to the power q . Again, λ_q is a computation that you are doing in the subscript, and, as I have pointed out, either with the sequence, or its transform, whenever you do arithmetic in the subscript, this is computed modulo N . And, it turns out that, so what this is saying is that, if my sequence lies entirely in the ground field, then the transform satisfies this conjugacy constraint. But the converse is also true, namely that, if I have a sequence whose transform satisfies this conjugacy constraint, then that sequence must actually be in the ground field.

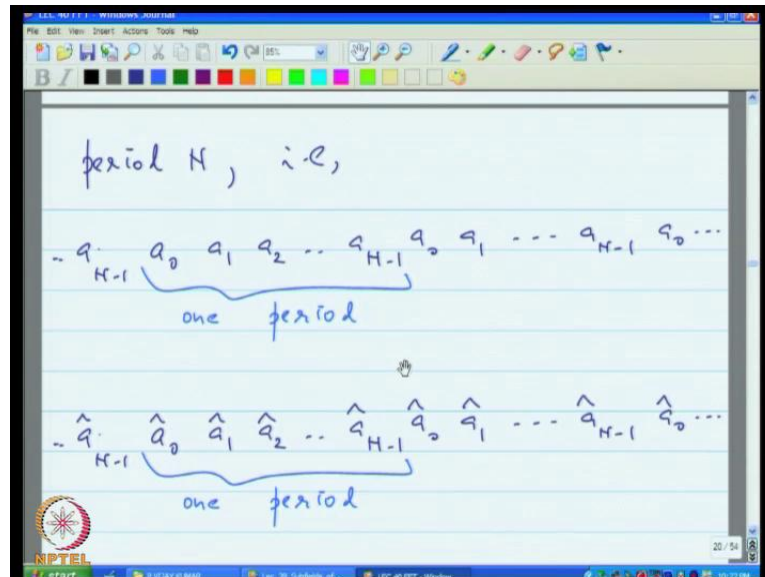
(Refer Slide Time: 30:40)

PERIODIC SEQUENCE INTERPRETATION

since arithmetic in the
subscript of a_t or else \hat{a}_x is
always computed (modulo N), it is often
convenient to think of both
 (a_t) as well as (\hat{a}_x) as being
sequences over \mathbb{Z}_N .

Now, before proceeding further, I have a couple of slides which say, you know, I keep saying that, the subscript arithmetic is modulo N , and, there is an alternative way of thinking about it, which might make it easier to picture, and that is, that you think of all your sequences a_t , now up to now we have thought of a of t ; for example, as a vector, whose length is N ; same thing with the transform; but you can instead think of these, as periodic sequences.

(Refer Slide Time: 31:12)



So, this is the alternate picture, that is, you think of your sequences being periodic with period N . Then it is automatic that all subscripts are computed modulo, all subscripts arithmetic is modulo N , because this is really the N th term in the sequence. But since a N is a 0 , that is the same as computing the subscripts mod N . This is the other way to think about it; that is, that really, although you are working with only a fragment of the sequence, that fragment is, has all the information, because it is one complete period of the sequence; same thing in the transform domain.

(Refer Slide Time: 31:55)

This is consistent with the defn:

$$\hat{a}_\lambda = \sum_{t=0}^{N-1} a_t e^{i\lambda t}$$
$$\hat{a}_{\lambda+N} = \hat{a}_\lambda$$

since α has order N .

Now, you might say that, do I not have to worry about consistency here? You do not, because here is the definition of the transform, and if this is periodic, you are wondering, is this periodic, but you can check, because if you add N to this, then you will be adding N to that; since α has order N , you will find out that, $\hat{a}_{\lambda+N}$ is the same as \hat{a}_λ ; maybe I should just add... So, let me reword this slightly. Since α has order N , alright.

(Refer Slide Time: 33:12)

Then

$$\hat{a}_{\lambda_2} = [\hat{a}_{\lambda}]^2$$

conjugacy relation

(where λ_2 in the subscript is computed (mod N)).

(conversely $\hat{a}_{\lambda_2} = [\hat{a}_{\lambda}]^2$ all λ)

$\Rightarrow a_t \in \mathbb{F}_2$, all t .

With that, now, let us get back to our, our conjugacy relationship. So, remember that, what we are trying to actually say is to prove that, if a of t lies completely in the ground field, that, in the frequency domain, in the transform domain, we have this conjugacy relation; so let me formally call this the conjugacy relation, alright. So, we want to prove this.

(Refer Slide Time: 33:55)

pf (of $\hat{a}_{\lambda_2} = [\hat{a}_{\lambda}]^2$ when $a_t \in \mathbb{F}_2$ all t)

$$\hat{a}_{\lambda_2} = \sum_{t=0}^{N-1} a_t \alpha^{\lambda_2 t}$$

$$= \sum_{t=0}^{N-1} (a_t \alpha^{\lambda t})^2$$

since $a_t^2 = a_t$

$$= \left[\sum_{t=0}^{N-1} a_t \alpha^{\lambda t} \right]^2$$

since $\alpha^2 = \alpha$

And, so this is quite straight forward. So, so let us start with this expression and we will reduce it to this form. So, a hat lambda q, you just follow along with the definition; so in case of lambda, we have lambda q. But then you can regard this as alpha to the lambda t raised to the q th power and also recognizing that, a t, remember a t belongs to the ground field, and a t to the q is therefore, a t; because this is the test for membership in a subfield, if you remember; that, a t belongs to the subfield F q, if and only if, it satisfies the property that, a t to the q is a t. So, we did this in the last lecture, and I... We reviewed that today. So, that means that, you can take this, replace this by a t to the q and therefore, regard this whole thing as being raised to the q eth power. And, since q is the power of the characteristic, you can take this q completely outside this summation. But what is left inside is nothing, but a hat lambda. So, that proves that, a hat of lambda q is a hat lambda raised to the q eth power.

(Refer Slide Time: 35:10)

converse: Next suppose

$$\hat{a}_{\lambda^q} = [\hat{a}_{\lambda}]^q \text{ all } \lambda, \text{ then}$$

$$[a_t]^q = \left[N^{-1} \sum_{\lambda} \hat{a}_{\lambda} \lambda^{-\lambda t} \right]^q$$

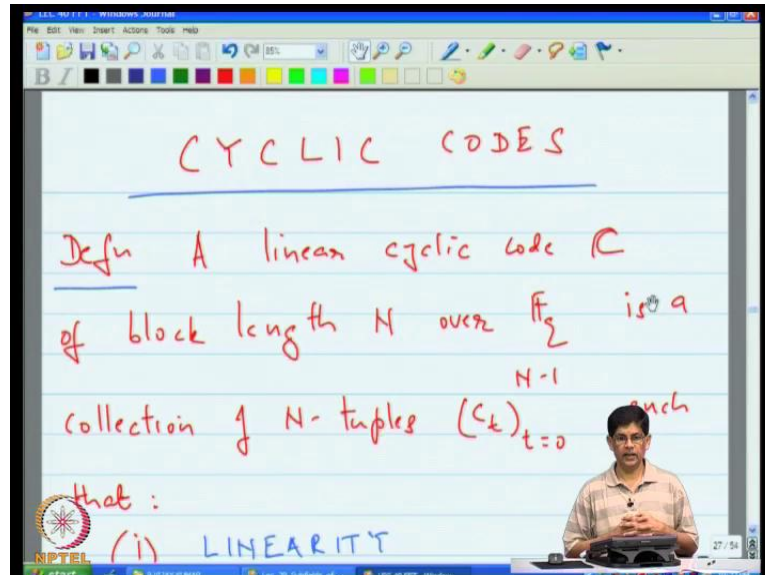
$$= (N^{-1})^q \sum_{\lambda} [\hat{a}_{\lambda}]^q \lambda^{-\lambda q t}$$

$$= N^{-1} \sum_{\lambda} \hat{a}_{\lambda} \lambda^{-\lambda q t}$$

Now, for the converse, so the converse tells you, well, supposing the conjugacy constraint holds, then am I assured that, the sequence belongs to the ground field, and the answer is yes. But again, I think I will just let you read through it on your own, in the interest of making up some time. Basically, the way you show that, this implies that the sequence is in the ground field, is to apply the membership test. What you try to do, is show that, a sub t to the q is a sub t again. And so this is your starting point, and then after some messaging, you can actually bring it to the point of a sub t, which proves that, it is in the ground field. So,

the important thing to keep in mind here, is that, is that, a sequence a of t is in the ground field if and only if, its transform \hat{a} of λ satisfies the conjugacy relationship, that is given here.

(Refer Slide Time: 36:33)



Now, we are ready to talk about cyclic codes. So, this was our goal and I think that, everybody listening to this lecture has been remarkably patient. But that is the way it goes; the big barrier to understanding cyclic codes is finite fields, and we have just gone through finite fields.

So, now, we are ready to actually discuss this, and with requisite background in finite fields, the theory actually is quite straight forward. So, first of all, what do we mean by cyclic codes? So, of course, we are talking about cyclic error correcting codes and these are block codes. The other thing which is not apparent from the name is that, this code is a linear code. So, people talk about cyclic codes all the time. At the back of their mind, they really mean linear cyclic codes; but since cyclic codes are almost invariably linear, in practice, people just, I guess, out of laziness, just call them cyclic codes. So, you should understand, and we will do the same in this class. We will assume that, cyclic codes is a reference to linear cyclic codes. So, what is the definition? So, a linear cyclic code of block length N is a collection of N -tuples, such that, two properties hold. One is of course, it must be a linear

code; so it must satisfy the linearity constraint, which says that, if $c^{(1)}$, $c^{(2)}$; so let me just correct that; that should have been ordinary 2, Arabic numeral 2.

(Refer Slide Time: 38:10)

(i) LINEARITY

$$(c_t^{(1)}), (c_t^{(2)}) \in C$$

$$\Rightarrow (c_t^{(1)} + \theta c_t^{(2)}) \in C, \forall \theta \in \mathbb{F}_q$$

(Thus C is a vector space over \mathbb{F}_q)

(ii) CYCLIC SHIFTS

$$(c_t) = (c_0 c_1 \dots c_{N-1})$$

So, if $c^{(1)}$ and $c^{(2)}$ are two code words, then their linear combination must also be a code word; must also belong to the code. So, if each of these is a code word, then their linear combination is also a code word. This θ here, belongs to \mathbb{F}_q . Now, our codes are over the ground field. So, linearity involves scaling by elements in the ground field. And, if you just think about it, what you are really checking is that, C is a vector space over \mathbb{F}_q . In fact, it is the subspace of \mathbb{F}_q^N , and this is nothing, but the subspace test. So, that is the first condition that you need to meet, that is, it is the linearity condition. The second condition that you need to meet is one of cyclic shifts. You need to verify that, if $c^{(t)}$ is a code word, then some, every cyclic shift is, cyclic shift is kind of like a wraparound shift, is also a code word.

(Refer Slide Time: 39:31)

(thus \mathcal{C} is a vector space over \mathbb{F}_2)

(ii) CYCLIC SHIFTS

$$(c_t) = (c_0 c_1 \dots c_{N-1}) \in \mathcal{C}$$

$$\Rightarrow (c_{t-\tau}) = (c_{N-\tau} c_{N-\tau+1} \dots c_{N-1} c_0 c_1 \dots c_{N-\tau-1}) \in \mathcal{C}$$

(i.e., \mathcal{C} is closed under cyclic shifts).

So, here, in this code word, when you put t equal to 0, you get c of minus tau; but since arithmetic is mod N , that c of N minus tau; or, if you think about it as a periodic sequence, you will come to the same conclusion. So, what we require is that, if this code word belongs to the code, then every wraparound cyclic shift of this code word also belongs to the code; that is, the code is closed under the cyclic shifts.

(Refer Slide Time: 40:18)

(i.e., \mathcal{C} is closed under cyclic shifts).

Given a codeword $(c_t) \in \mathbb{F}_2^N$ in \mathcal{C} ,

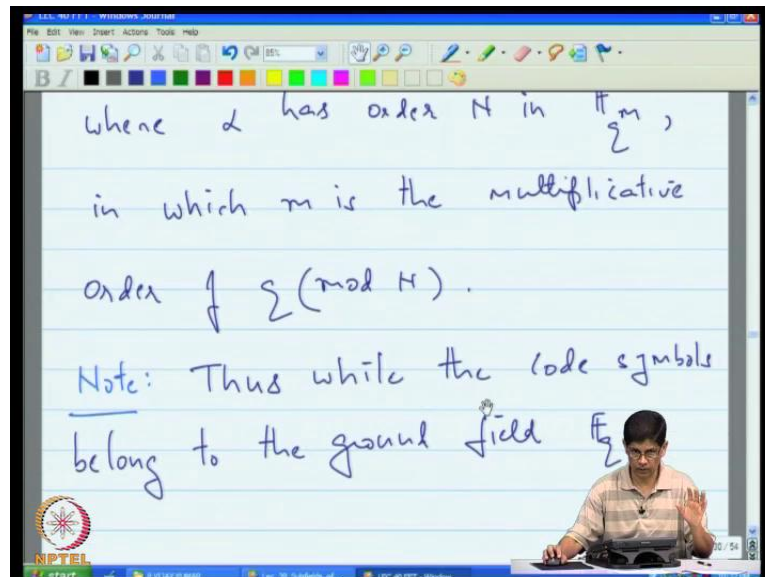
we define the transform (\hat{c}_λ) of

(c_t) via

$$\hat{c}_\lambda = \sum_{t=0}^{N-1} c_t \alpha^{\lambda t}$$

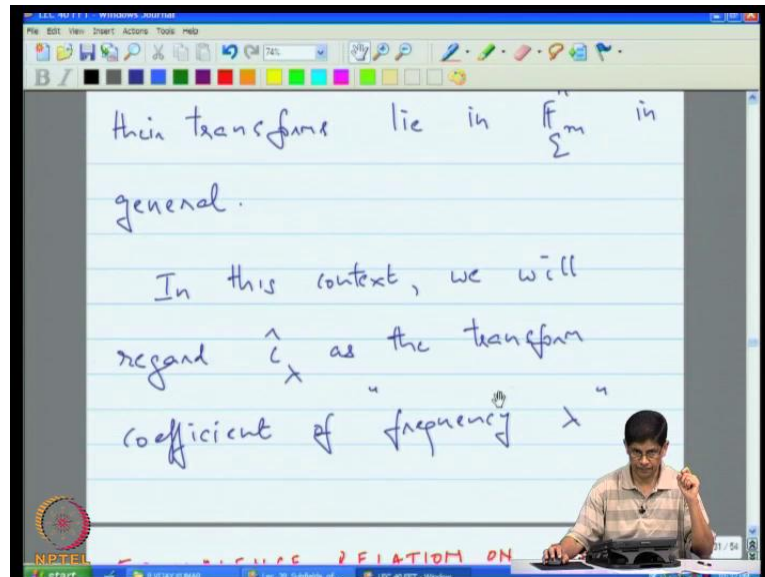
So, that is the definition of a cyclic code. And now, what we will do is, we will go ahead and analyze these codes. And it turns out that, a very convenient means of analyzing them, is using the finite field transforms. My, my own exposure to this view point came from, came many years ago from a noted coding theorist Lloyd Welch. Nowadays, it is a... The other proponent of this view point is Richard (()), and you can find this discussed in his text books on coding theory. Now, given a code word c of t , we define... So, now, I would, I am going to specialize our transform theory to the case of cyclic codes. So, given a code word in a cyclic code, we define the transform of c of t , in this way. So, it is the usual definition, where α has order N in F_{q^m} , in which m is the multiplicative order of $q \bmod N$.

(Refer Slide Time: 41:28)



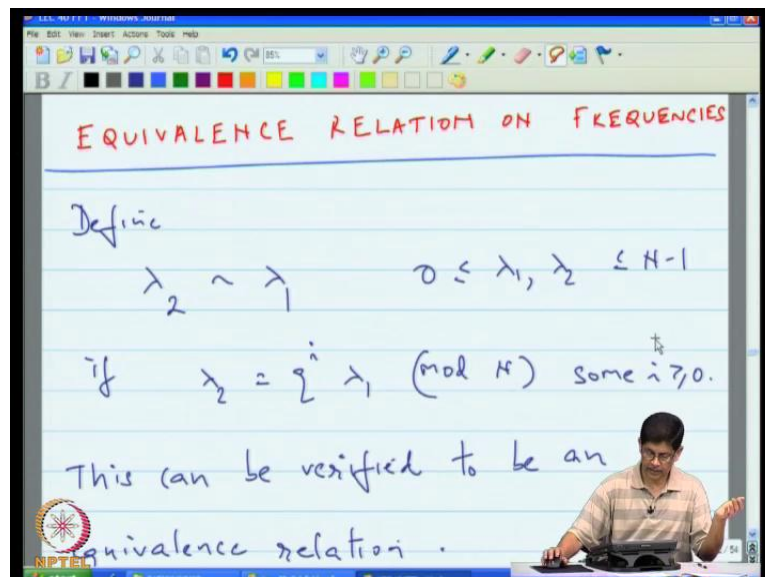
So, there is nothing new here. We already understand how transforms are defined. We understand that, if your sequence is in F_q , in order to find an element of order, of order N , you have to go to a larger field F_{q^m} . So, we are completely familiar with all of this. Yes, so this is just saying something that is obvious; that is, while the code symbols belong to the ground field, the transform coefficients, in general, lie in the big field.

(Refer Slide Time: 42:00)



Now, in this context, we will regard \hat{c}_λ as the transform coefficient, I guess, I need to rewrite this a little bit; is the transform coefficient of c of t at frequency λ . Let me try moving this one last time; there we go. So, it is the transform coefficient of this at frequency λ .

(Refer Slide Time: 43:33)



So, we are going to interpret the subscripts of c hat as frequencies, in the ordinary way you would do in the discrete Fourier transform. Now, it turns out that, it is convenient. So, we discussed cyclotomic cosets in connection with a finite field. So, there is something, there are cyclotomic cosets that also show up here. They are slightly different in terms of their parameters, but the definition is more or less the same. So, in this connection, what we do is, we define two frequencies to be equivalent, lying between 0 and N minus 1; if one is, if λ_2 is λ_1 times some power of q ; and again, this arithmetic is computed modulo N . This can be verified to be an equivalence relationship and in just the way, I remember, when we discussed the p cyclotomic cosets mod p to the N minus 1; there, I wrote down the three, I had three lines which actually showed that, the reflexive, the symmetric, and the transitive properties were satisfied. You can, in exactly the same way, you can verify that, those three properties are satisfied here as well.

So, what that means is that, this is an equivalence relationship and a key property of equivalence relationship is that, it partitions the set of all frequencies. So, you think of the λ s as being frequencies. So, it says that, the frequency domain is now partitioned into equivalence classes. Each of these equivalence classes is called now, a cyclotomic coset. So, the corresponding equivalence classes are called the q cyclotomic cosets mod N . Here is an example.

(Refer Slide Time: 45:20)

Example: $q = 2$, $N = 15$

there are 5 equivalence classes \Rightarrow

coset leader

frequencies

$\{ \lambda \}$

$\lambda_1 \sim \lambda_2$

$\lambda_1 = 2 \lambda_2 \pmod{N}$

coset leader

So, supposing q is 2, and N is 15; so this is going to be our favorite example, favorite set of parameters. Then these, you should think of these, as now being frequencies. So, these, all of these, should now be regarded as frequencies. These are the various frequencies λ . And now, we have an equivalence relationship, in which, in this particular case you defined λ_1 equivalent to λ_2 , if λ_1 is equal to some power of 2 times λ_2 mod N . So, that is our equivalence relationship in this particular case, and if you work it out, you will see that, these are the equivalence classes that you get. In this case, they also happen to be the p cyclotomic cosets mod p to the m minus 1. But when you introduce cyclotomic cosets in connection with finite fields, then you are always dealing with p cyclotomic cosets mod p to the m minus 1. With cyclic codes and the transform domain like this, the, you are a little bit more free to choose these parameters, because N need not be p to the m minus 1, although it is the case here. So, for example, N may have been, in this particular instance, 5 for example; it is perfectly ok for N to be five.

So, anyway, these are the 5 cyclotomic cosets and the convention is that, when you have a cyclotomic coset, the smallest element, the smallest representative, or the smallest element in each of the cyclotomic cosets is called the coset leaders. So, each row is a cyclotomic coset and the smallest element is the first; that is how it is conventionally written. So, these are the various coset leaders, and that is your expansion here. Then there is the obvious statement, that if there are k cosets, with n_i elements in the i th coset, then the sum of the n_i will give you N .

(Refer Slide Time: 47:29)

the coset leader.

In general, if there are cosets with n_i elements in the i -th coset, then $\sum_{i=1}^k n_i = N$.

Defn. A subset $S \subseteq \{0, 1, 2, \dots, N\}$

So, for example, here, this coset has of size n_1 equals 1, n_2 equals 4, n_3 equals 4, n_4 equals 2, n_5 equals 4 and the sum of all of these is 15. That is just introducing some notation. So, just to quickly recap.

(Refer Slide Time: 48:10)

Defn. A subset $S \subseteq \{0, 1, 2, \dots, N-1\}$ is said to be a closed set of frequencies if

$$\lambda \in S \Rightarrow 2\lambda \pmod{N} \in S$$

What we have done so far is, say, look, our interest is in cyclic codes; we defined what it means to talk about the cyclic codes. We said that now we are going to apply our transform

theory to the code words; that is, I am going to have a code word; I am going to treat that as my time sequence, and then I am going to compute its transform. And then in the transform domain, we are going to actually divide the frequencies into equivalence classes, which are called cyclotomic cosets. Now, the reason for doing that, excuse me, or the motivation is, because of the conjugacy constraint; remember, we had the conjugacy constraint that, if the code word belongs to the ground field, then the transform actually, the transform values satisfy the conjugacy constraint, and that brings in λ^q , given λ . So, it turns out that; that is the reason why we are interested in this cyclotomic cosets; but we will see a little bit more of this pretty soon. Now, we come to, now we are slowly winding up a way, to actually saying, look, I have given you a definition of cyclic codes, but in essence, I can give you a simpler definition of cyclic codes, in the transform domain.

So, that is why we had it. So, a subset S of the integers from 0 to N minus 1 is said to be a closed set of frequencies, provided, if λ belongs to S , then λ^q must also belong to S . So, that means that, these are, you typically think of these as conjugates, λ and λ^q . So, if λ belongs to S , then λ^q must also belong to S . Now, you can actually verify that, this forces the closed sets to be the union of cyclotomic cosets. Why is that, because let us go back to our example here. Supposing, I was trying to construct a closed set, and I happen to pick 3, and I said, I want 3 to be my closed set, but because the set is closed, I am forced to include 6, 12 and nine. So, when I include 3, I am actually bringing along the entire cyclotomic coset. Supposing, I next choose 2; I am forced to bring 1, 2, 4 and 8, because 2 times 2 is 4, times 2 is 8, times 2 mod 15 is 1. So, I am forced to bring along this entire cyclotomic coset. For this reason, closed sets are the union of cyclotomic cosets. Now, we come to our key concept, that of the null spectrum.

(Refer Slide Time: 51:03)

NULL SPECTRUM

Defn Let C be a linear, cyclic code of block length N over \mathbb{F}_2 .

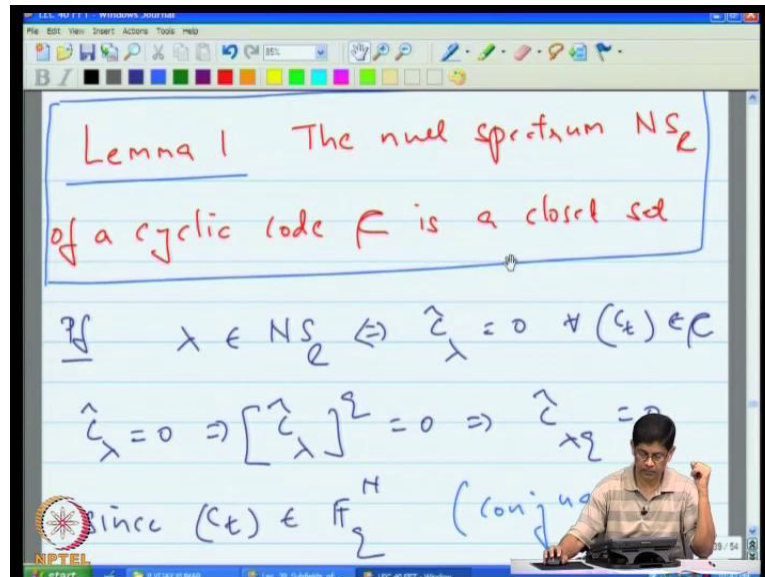
Then the collection of frequencies

$$NS_C = \left\{ \lambda \mid \begin{array}{l} 0 \leq \lambda \leq N-1 \\ \hat{c}_\lambda = 0 \text{ for all } (c_k) \in C \end{array} \right\}$$

is called the null spectrum.

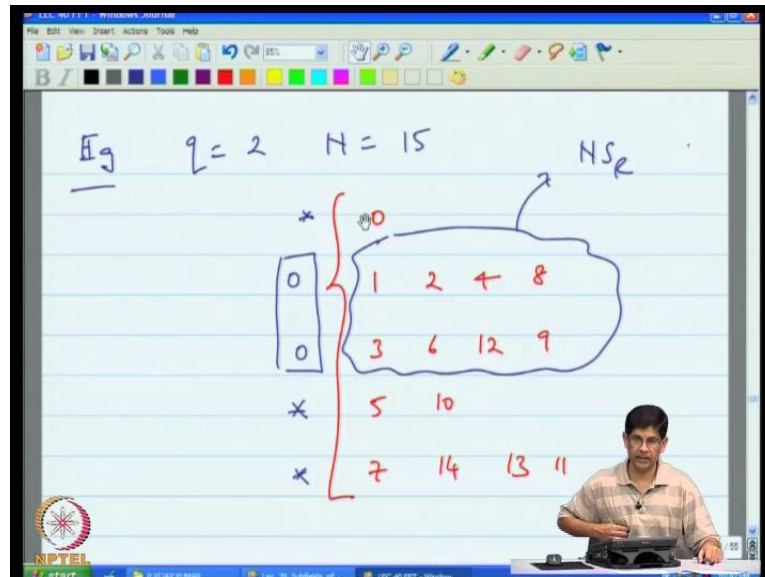
So, given... So, there, the null spectrum is something that is associated with the cyclic code and it turns out to characterize the cyclic code. So, let C be a linear cyclic code of block length N , over the finite field of q elements; then the collection of frequencies... So, this NS here, stands for the null spectrum; then the null spectrum of the code is the collection of all those frequencies λ , having the following property. Of course, since the code words are of length N or period N , if you think of them as periodic sequences, the λ is lying between 0 and N minus 1 . And now, the requirement is that, you take all those frequencies having the property that, every code word in the code has a transform value of 0 , at this frequency. So, that is why it is called the null spectrum.

(Refer Slide Time: 52:27)



The null spectrum is the collection all those frequencies, where the transform value of every code word in a code is 0. So, that is called the null spectrum of a code. It turns out now that, the null spectrum of a cyclic code is a closed set. In other words, you can actually construct cyclic codes very easily, because all that you need to do it turns out, is, let us go back to our cosets, here. So, supposing, for instance, I was designing a cyclic code here, and it turns out that, you can actually design cyclic codes in a very simple way. Maybe, what will I do, is let me just copy the page and insert it, where I can use it; here we go. So, I will just edit this little bit; some of this is unnecessary on this page; here we go.

(Refer Slide Time: 53:38)



So, if I wanted to design a cyclic code here, as it turns out, then all that I need to do is, specify a certain null spectrum. You know that the null spectrum is a closed set; that is, the set of all frequencies, where all the code words have transform value 0; that is a closed set, we will show; and so I can design a cyclic code, by just looking at the, each of these cosets and saying 0, and not zero.

So, for instance, I can design a cyclic code in here, by just saying, I am going to choose this 0, 0 and then I am going to put stars here; meaning that, I am not going to impose the 0 value here; and then it turns out that, the presence of these two 0s implies that, this set over here, is then the null spectrum of the code. And, it turns out that, designing a cyclic code is as easy as that; it is just picking a certain union of cyclotomic cosets and declaring that to be the null spectrum. And then you, the result is a cyclic code and we will show that. Of course, the big question is, **is** how good is your cyclic code. So, there you have to think a little bit more, and there is a certain result, that simplifies the design of cyclic codes; a very simple result, and we will come to that. But the point

I want to make is that the transform domain approach gives you this complete simplicity; I mean, it cannot get any simpler than this; that you are saying that, all that a cyclic code is, is just a collection of vectors, whose transform values are zeroes, at a closed set; that is, at

every frequency in the union of a certain collection of cyclotomic cosets. So, that also means, for example, that if you are constructing cyclic codes which are binary, because q is 2, of length 15, then the total number of such cyclic codes is precisely 2 times 2 times 2 times 2 times 2, which is 32, because for every cyclotomic cosets,

You can either choose to include it into the null spectrum or not. So, that gives you a binary choice, and there are 32 possible choices with that, and each one of them gives you a distinct cyclic code. And that completely describes all cyclic codes, whose block length is 15 and whose alphabet is binary. Now, which are the good ones amongst those, well, we will come to that. We have just about a minute left and the first theorem, the first lemma result is that the null spectrum of a cyclic code is a closed set; and I have already mentioned it. We will begin our next lecture by actually proving this. I think this might be a good place to stop. Thank you.