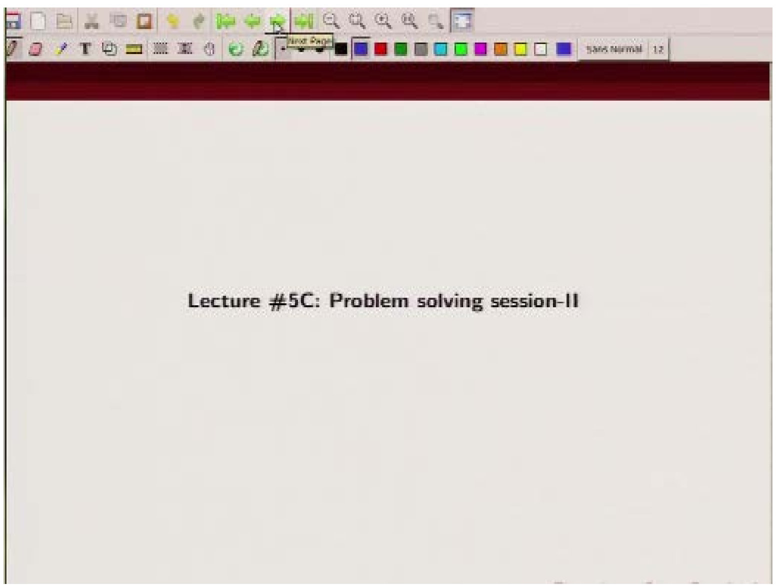
Indian Institute of Technology Kanpur
National Programme on Technology Enhanced Learning (NPTEL)
Course Title
Error Control Coding: An Introduction to Convolutional Codes

Lecture -5C
Problem Solving Session-II

by
Prof. Adrish Banerjee
Dept. Electrical Engineering, IIT Kanpur

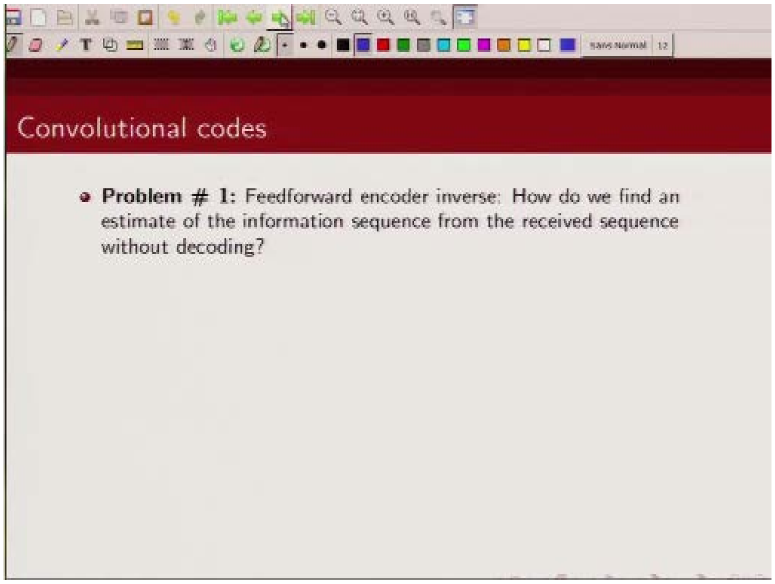
Welcome to the course on error control coding, an introduction to convolutional code.

(Refer Slide Time: 00:19)



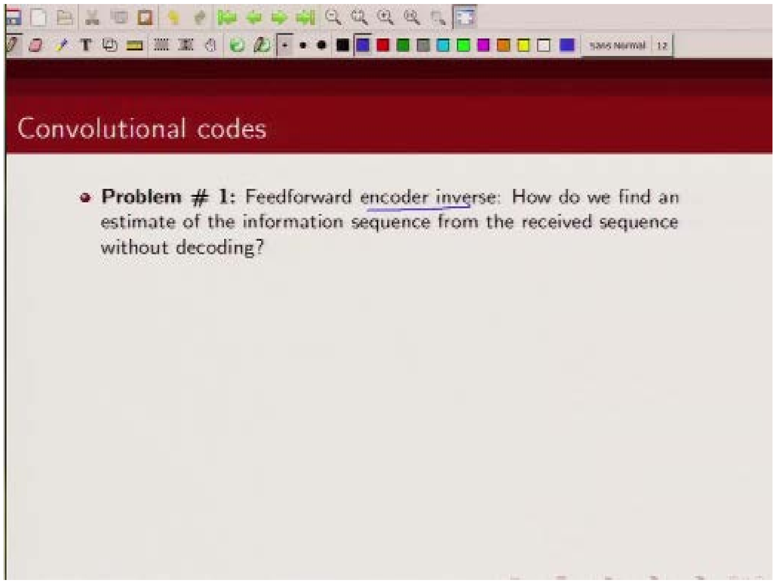
So today we are going to continue with some more problems related to convolutional code. So let us solve some more problems and then we will move to our other topic.

(Refer Slide Time: 00:32)



So first question is on feedforward encoder inverse.

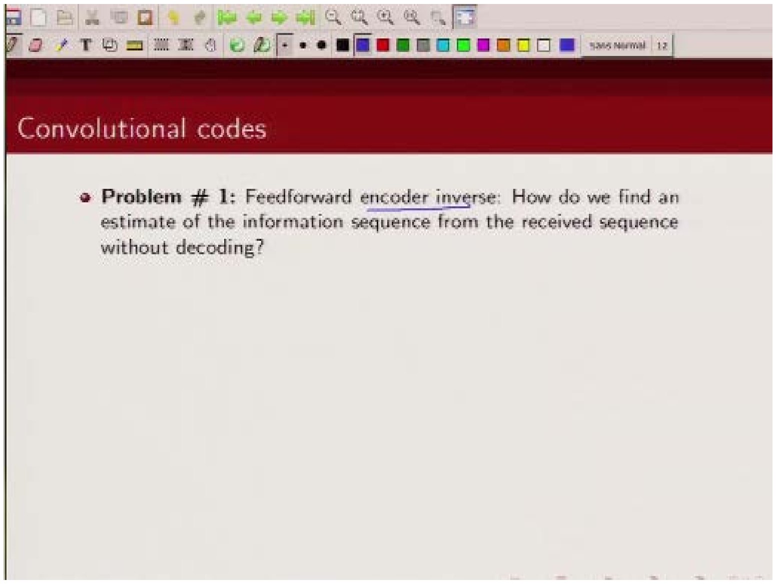
(Refer Slide Time: 00:39)



So what is encoder inverse, we will talk in a minute. So many times we are interested in estimating the information sequence directly from the received sequence without decoding it. So for example if we are encoding a sequence using systematic encoder then you can directly from the received bits you can get back your information bits.

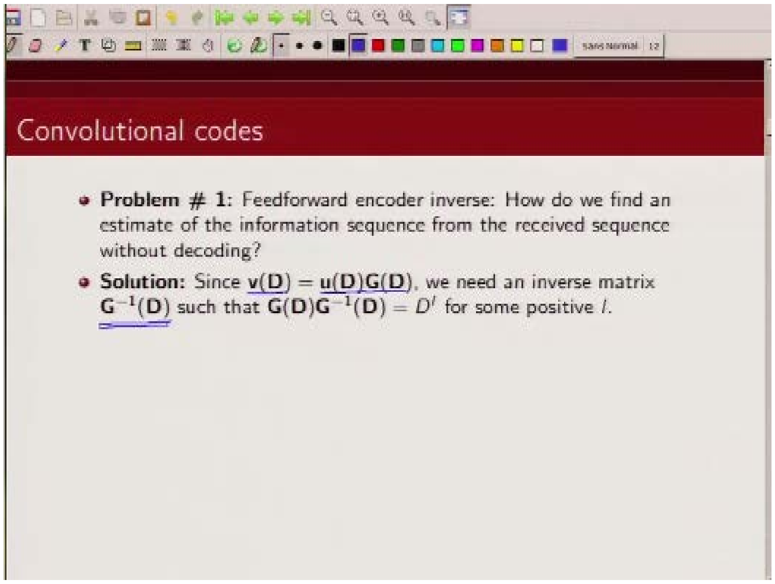
However, if we are using a nonsystematic encoder then you cannot directly get the information bits. So we are talking about an encoder inverse which will allow us to recover back the information bits directly without decoding.

(Refer Slide Time: 01:24)



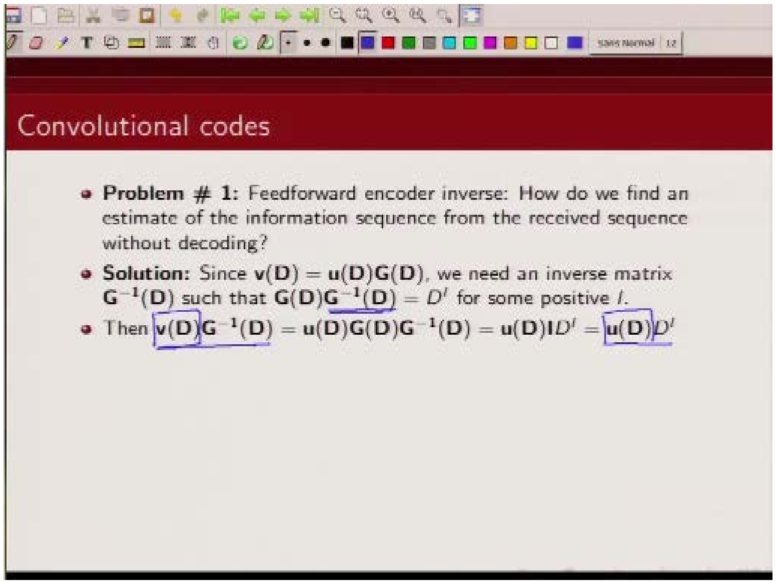
So in this problem we will look into what is an encoder inverse and under what condition the encoder inverse exist.

(Refer Slide Time: 01:37)



So as we know that our coded bits can be written as our information bits times is generator matrix encoding matrix, and the problem that we are looking at is finding out the encoder inverse, and we will talk about whether a feedforward inverse for this encoding matrix exist or not and under what condition it exist.

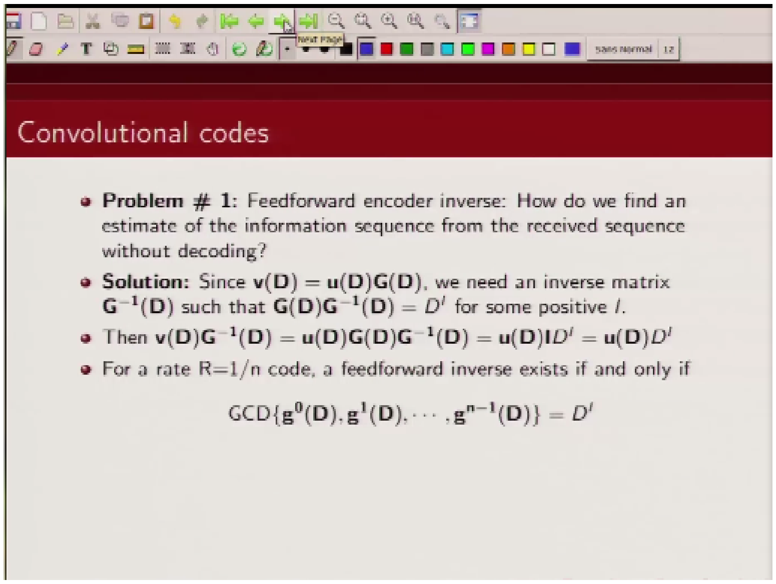
(Refer Slide Time: 02:16)



So if there exists a feedforward inverse then if we from the received sequence if we just multiply by the encoder inverse we can get back our original information bits without decoding after some delay. So this Dl is some delay Dl. So what we are saying is we are interested in finding this encoder inverse does this encoder inverse exist, a feedforward encoder inverse does it exist such that $G(D)G^{-1}$ is some delay element.

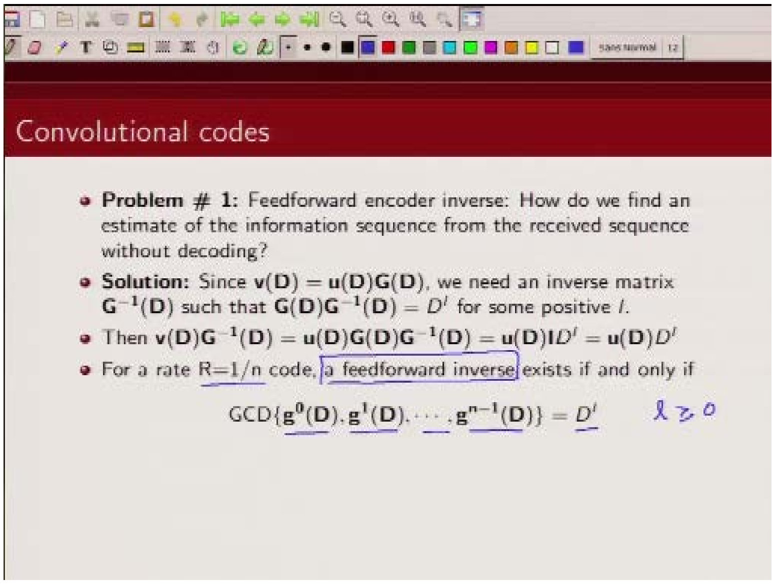
And what is the use of this, so if you have your information sequence v(D) if it passes through this encoder inverse circuit, we can directly get back our information sequence. And in many cases for example, if the channel conditions are good you may directly want to first guess or check whether the information bits are directly estimate information bits. So you may want to pass it through this encoder inverse circuit.

(Refer Slide Time: 03:23)



So I am now stating without proof the condition under which these encoder inverse exist.

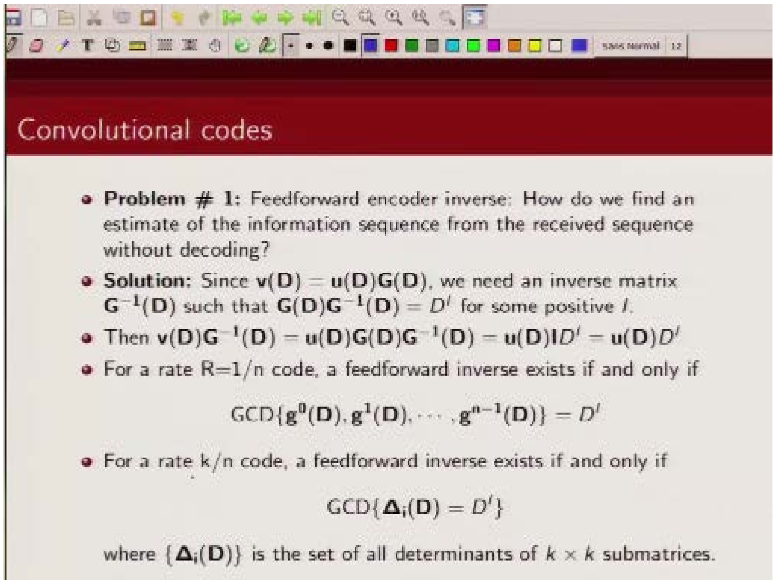
(Refer Slide Time: 03:34)



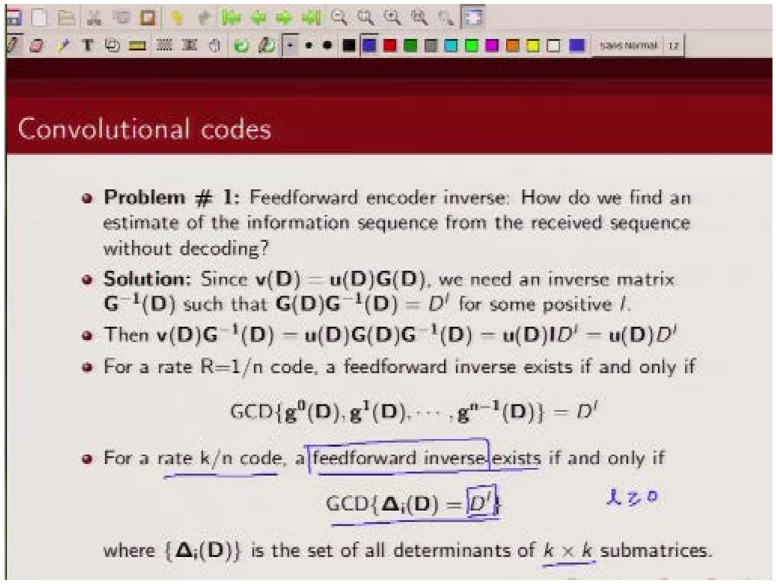
A feedforward encoder inverse exist, so for a rate 1/n code a feedforward inverse will exist if the greatest common deviser between these n generator sequences of this rate 1/ n code. If the greatest common deviser among these generator is some delay element this l is something which is greater than equal to zero.

So they do not have many term common in them, just some D times, basically some delay element. So we do not want these generator sequences to have any term common between then. If they have any term common between them, then a feedforward inverse would not exist. Then there will be a feedback inverse.

(Refer Slide Time: 04:33)

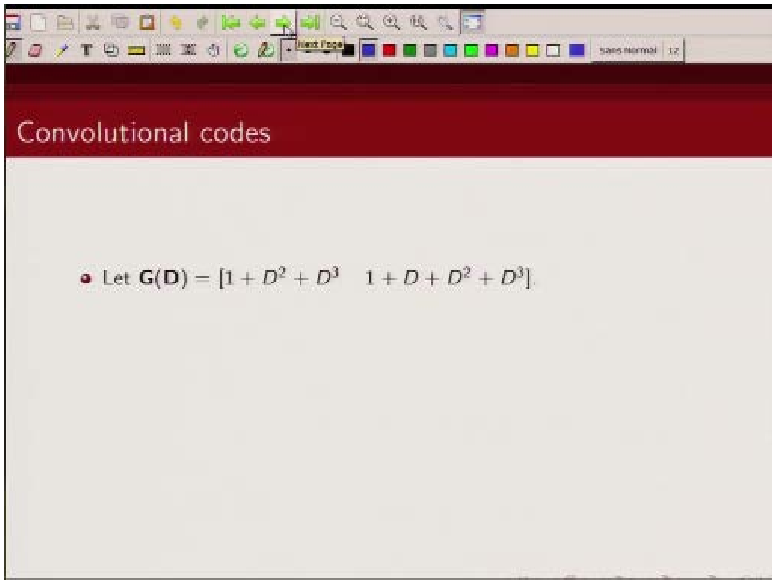


(Refer Slide Time: 04:36)



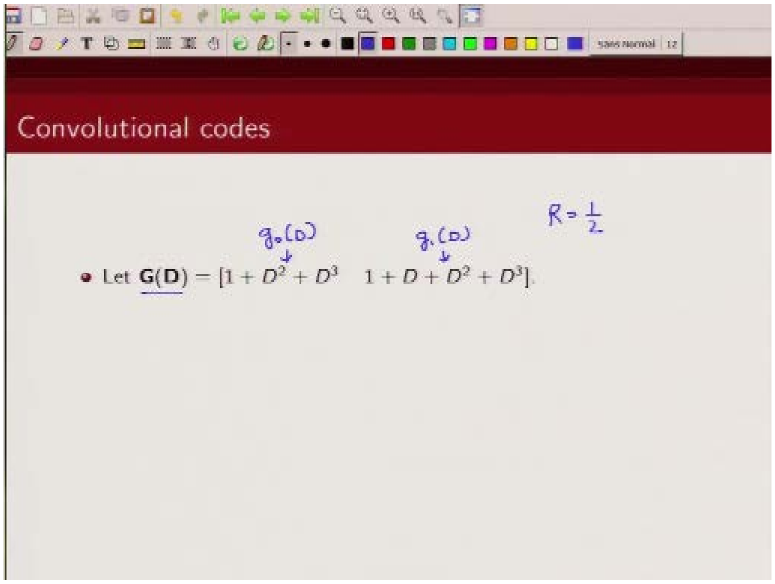
Similarly for a rate k/n code a feedforward inverse will exist if and only if the greatest common deviser. If you look at set of all determinants of k x k submatrices of this generator matrix, then the GCD of these set of determinants should be again some D^l where l is a positive number. So we do not want the determinants of this k x k all possible k x k submatrices to have any common term among them. If this condition is satisfied a feedforward inverse exist.

(Refer Slide Time: 05:27)



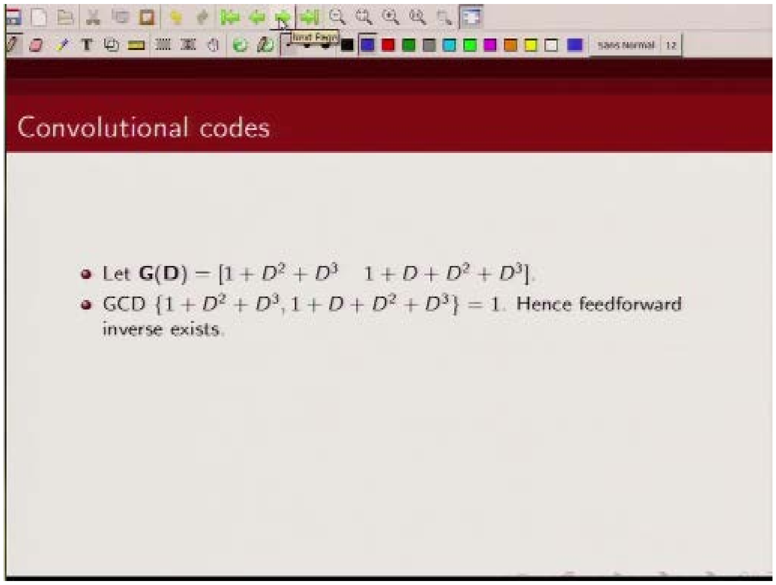
So let us take an example where feedforward inverse exist.

(Refer Slide Time: 05:35)



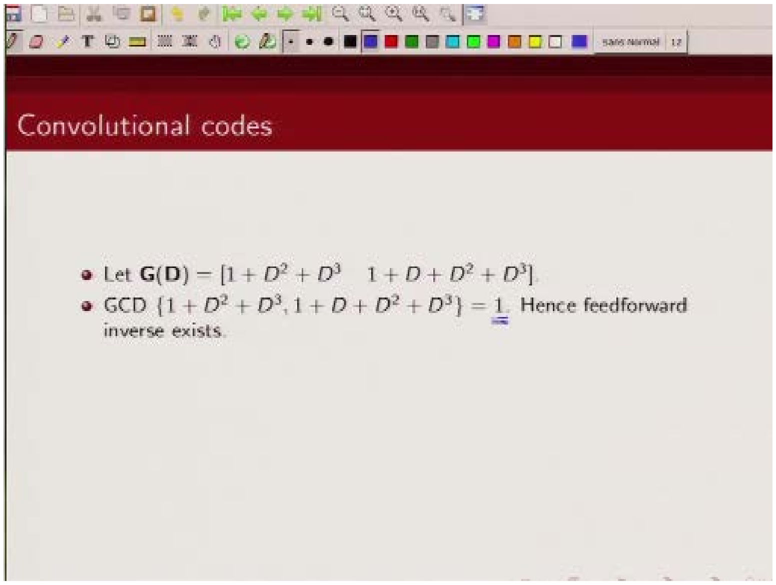
So we are considering a feedforward rate ½ is the rate ½ encoder. So g_0(D) is this one and g_1(D) is this.

(Refer Slide Time: 05:54)



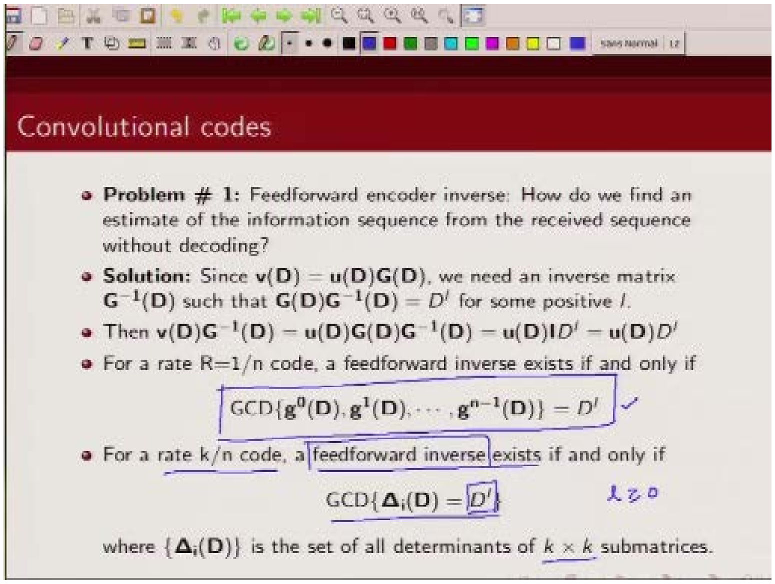
And what is the common deviser between them?

(Refer Slide Time: 05:58)



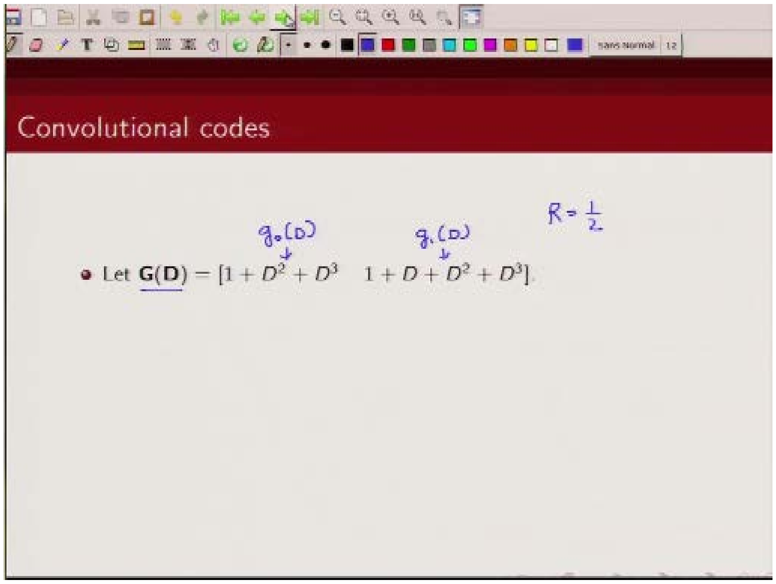
We can check basically they do not have any common terms so the greatest common deviser is 1.

(Refer Slide Time: 06:06)



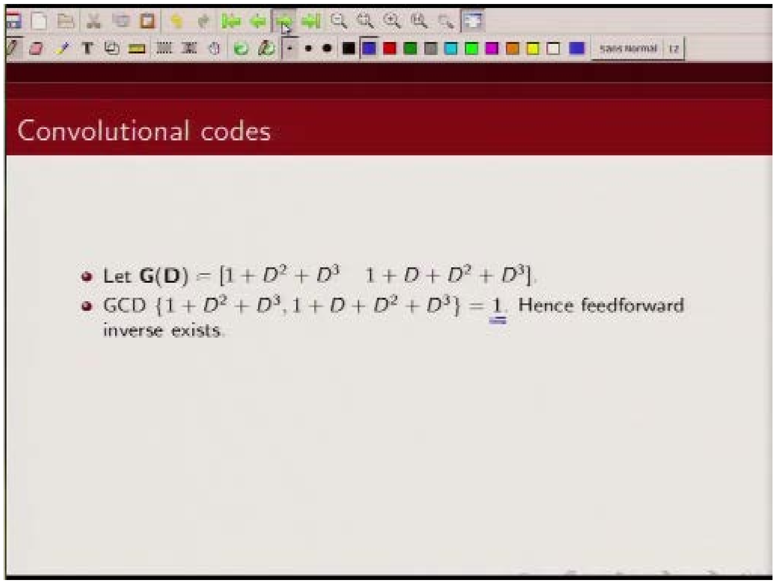
So if we go back and look at our condition for encoder inverse to exist this condition is satisfied.

(Refer Slide Time: 06:19)



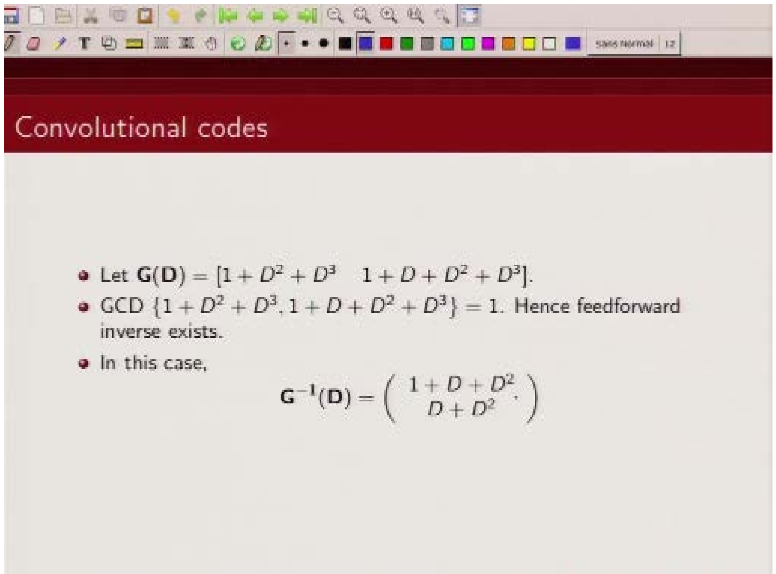
So for this particular code with G(D) given by this.

(Refer Slide Time: 06:24)



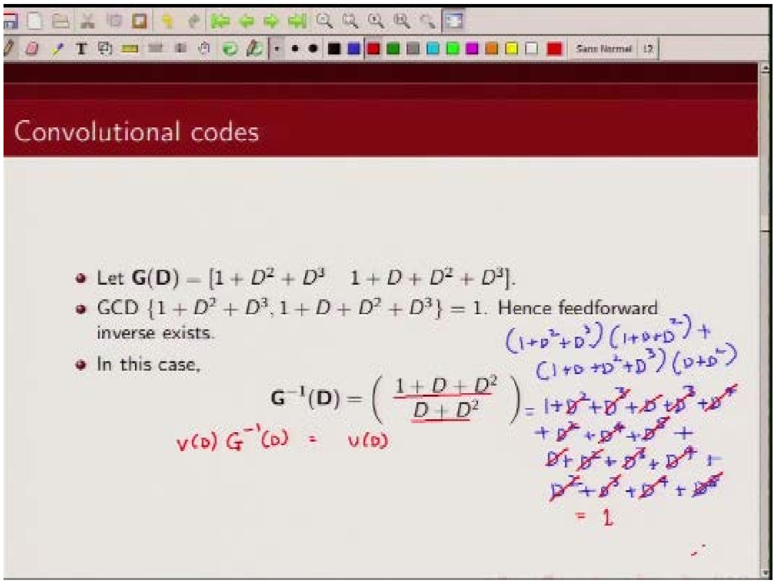
Will have a feedforward encoder inverse.

(Refer Slide Time: 06:26)



And in this particular case the feedforward inverse is given by this okay. So you can check G(D), G(D) inverse will be 1.

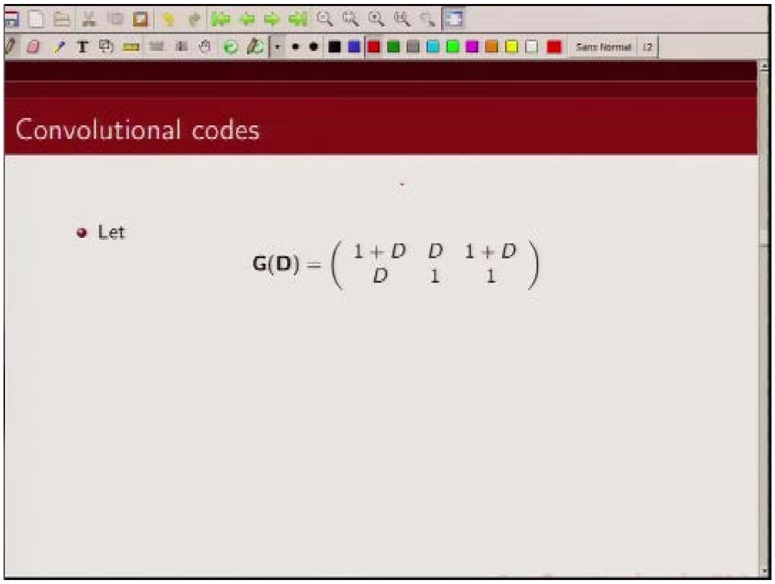
(Refer Slide Time: 06:43)



So we can just do a simple check $(1+D^2+D^3)$ x $(1+D+D^2)$ + $(1+D+D^2+D^3)$ x $(D+D^2)$ this is, so this is 1+ D^2+ D^3+ D times D^3 + D^4 +D^2 times D^4+ D^5 then multiply this with this you get +D times D^2+ D^3+D^4+D^2 +D^3+D^4 +D^5 okay and let us see so D^5, D^5 cancels out D^4, D^4 cancels out then this D^4, D^4 cancels out D^3, D^3 cancels out D^2, D^2 cancels out D, D cancels out D^3, D^3 cancels out D^2, D^2 cancels out so what we are left is basically 1 okay, so and you can see this is a feedforward.

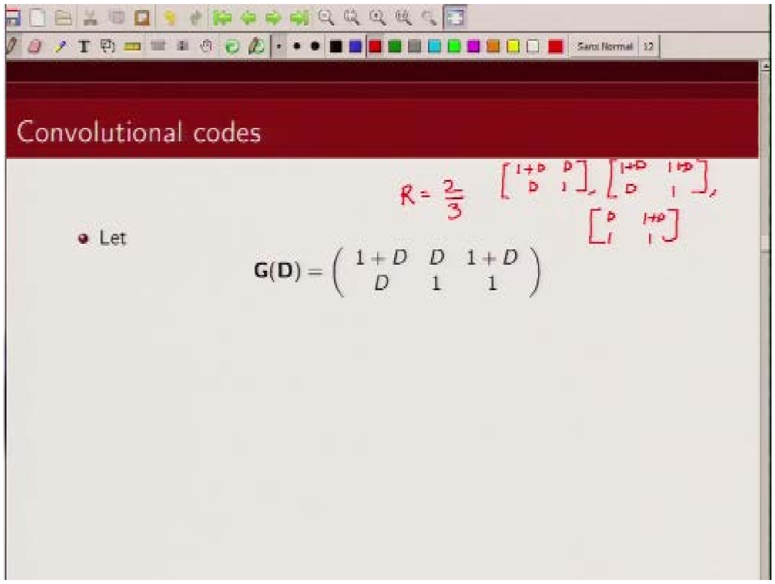
Inverse so if you have your v(D) and if you pass it though this, this thing what you will get is your get back your information sequence okay, get back your information sequence.

(Refer Slide Time: 08:34)



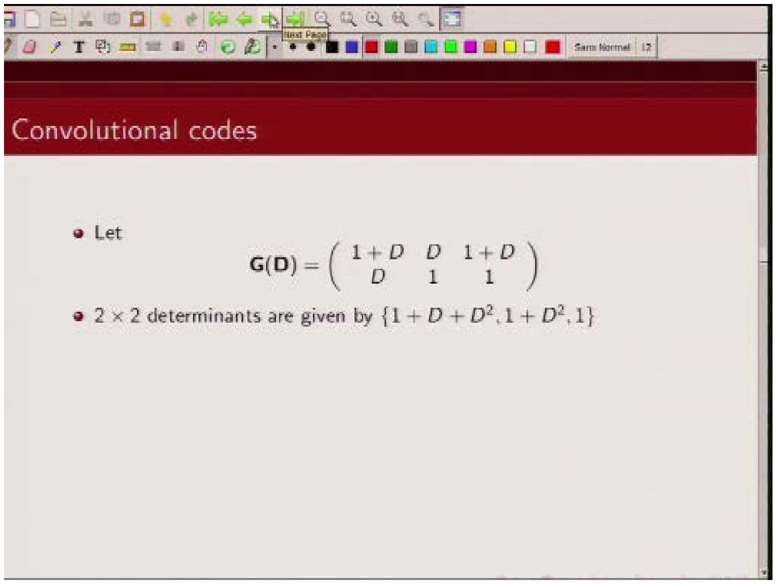
Now let us look at example for rate

(Refer Slide Time: 08:38)



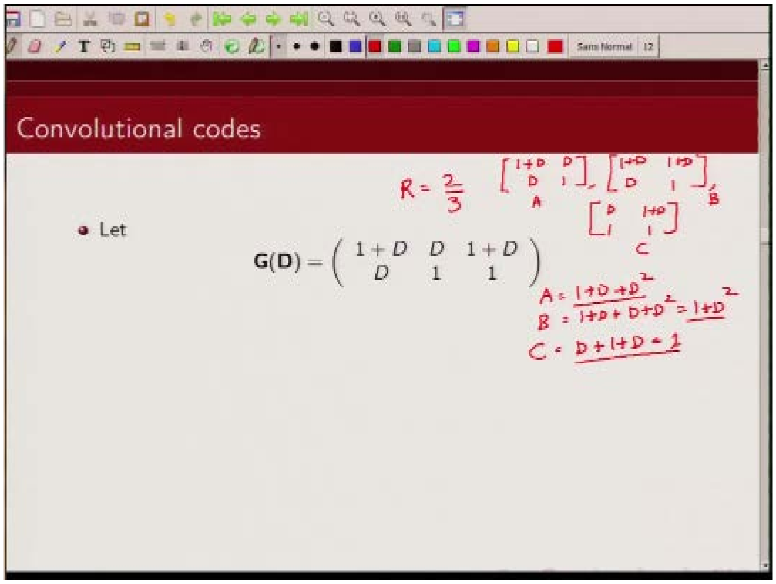
R= 2/3, so in this case we first have to find the determinant of all to 2 x 2 submatrices so what are those 2 x 2 submatrices, one of them is this 1+D D D 1 next one is 1+ D D 1+D 1 and the third one is D 1 1+D 1, so these are the three 2 x 2 submatrices.

(Refer Slide Time: 09:13)



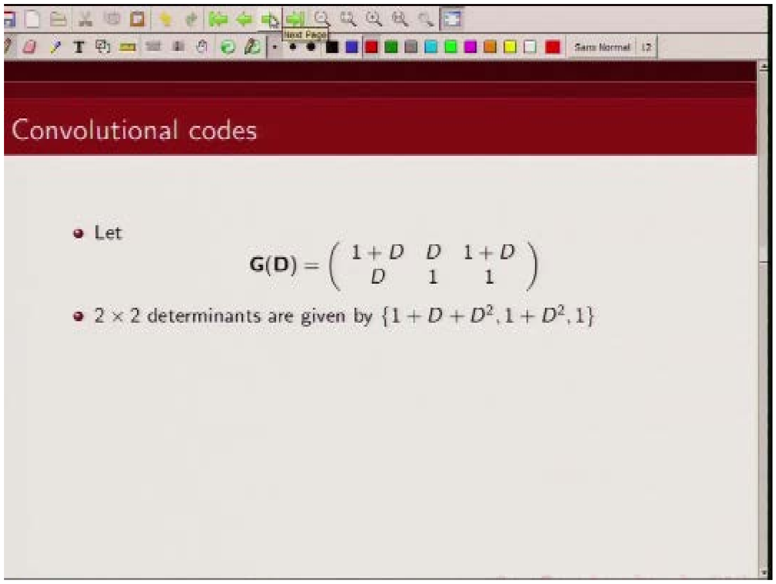
And we can

(Refer Slide Time: 09:17)



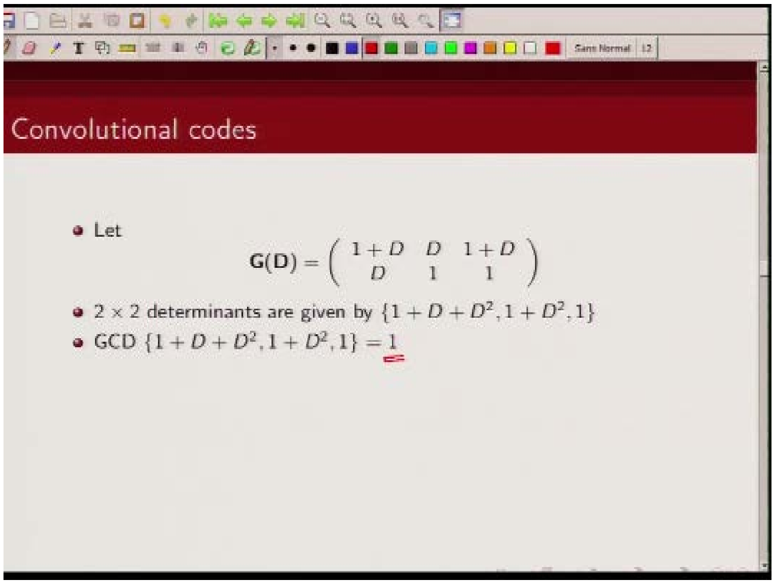
Find out the determinant in this case, in this case is just call it A, B and C, in case of A the determinant is $1+D+D^2$, in case of B the determinant is $1+D+D+D^2$, so that is$1+D^2$ and C is D + 1+D so that is 1, so these are the determinants of these 2 x 2 submatrices.

(Refer Slide Time: 09:57)



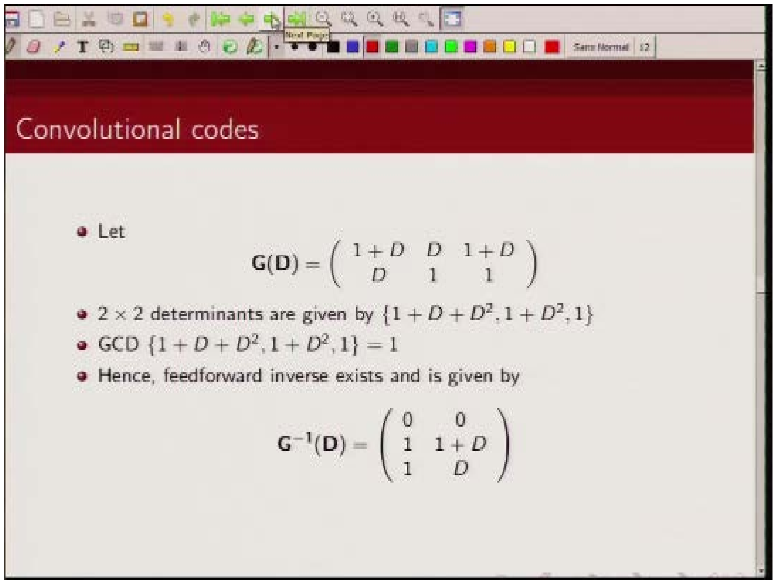
And that is what I have listed here.

(Refer Slide Time: 10:00)

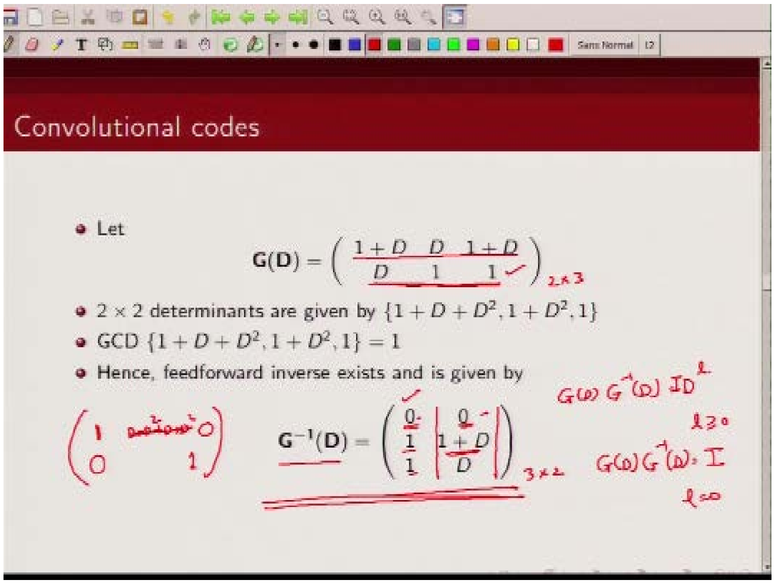


$1+D+D^2$, $1+D+D^2$ and 1. Now we need to check what is the greatest common deviser among them and in this case the greatest common deviser is again 1, so they do not have these determinants of these 2 x 2 submatrices, do not have any term common on them, so in this case also

(Refer Slide Time: 10:31)



(Refer Slide Time: 10:33)

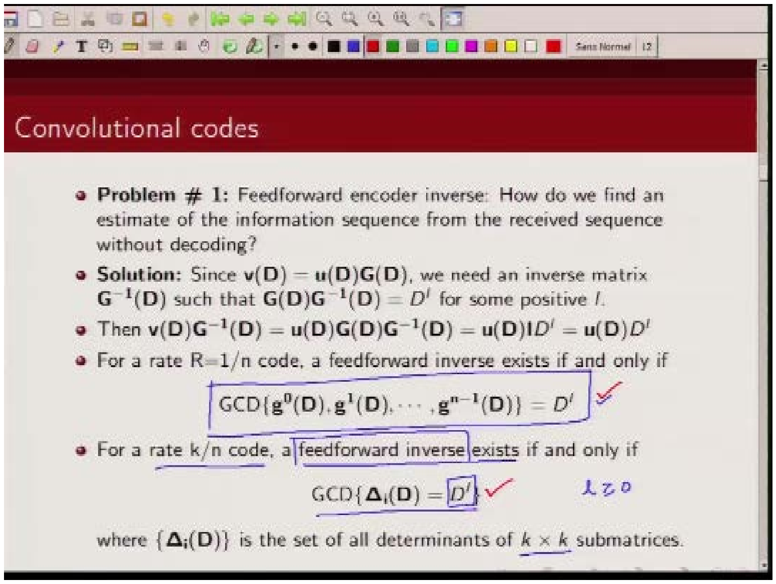


A feedforward inverse exist and this is given by this okay, and again we can check that G(D) inverse is basically will be some delay elements wireless greater then equal to zero with something like this, we can verify this quickly. Let us see this, this will be 1 +D times 0 and then this will be D 1+D, this is 2 x 3 and this will be 3x2 matrix so what we will get is a 2x2 matrix and so this will be some I times 2x2 matrix so let us just workout, so this will be 1+D times 0 that is 0, and then you have D times 1 and this is 1 + D so that is 1, first term will be 1 and then this will be multiply this by this.

So that is 1 + D into 0 that is 0 D into 1 + D so that would be $D+D^2$ and then 1+D into D so that is again $D+ D^2$ so this will be 0. Next multiply this row by this column so what we get D times 0 1 times 1, 1 times1 so that is 1 + 1 is 0, and if you multiply this by this the second row by second column what you get is 0 times D 1 times 1+ D and 1 time D so that is 1+ D +D so that is 1.

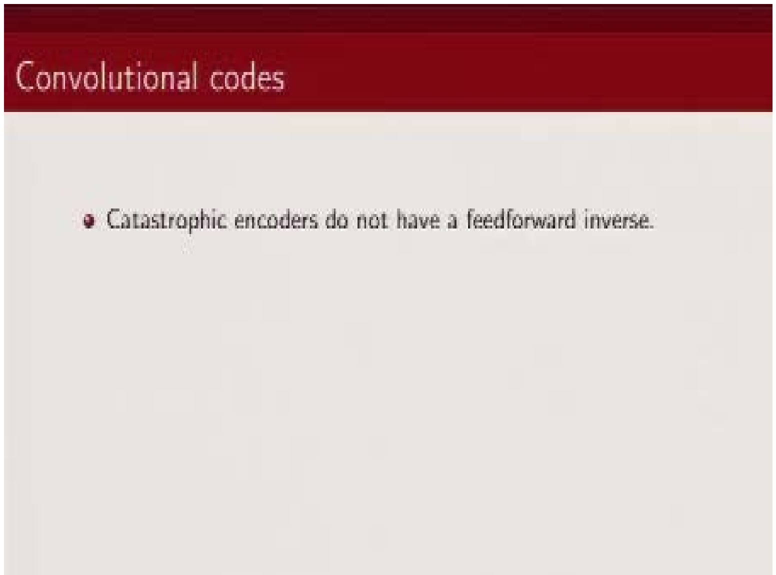
So again what we are getting for this case is G(D), G^{-1}(D) is identity matrix so l is 0 essentially here okay, so this is a inverse for this generator matrix and we can see that this, all the terms are feedforward terms is 1 1 /D and 1 1, so this a feedforward inverse for this conversional code with this generator matrix okay. So now in to recap basically so the condition under which the feedforward

(Refer Slide Time: 13:34)



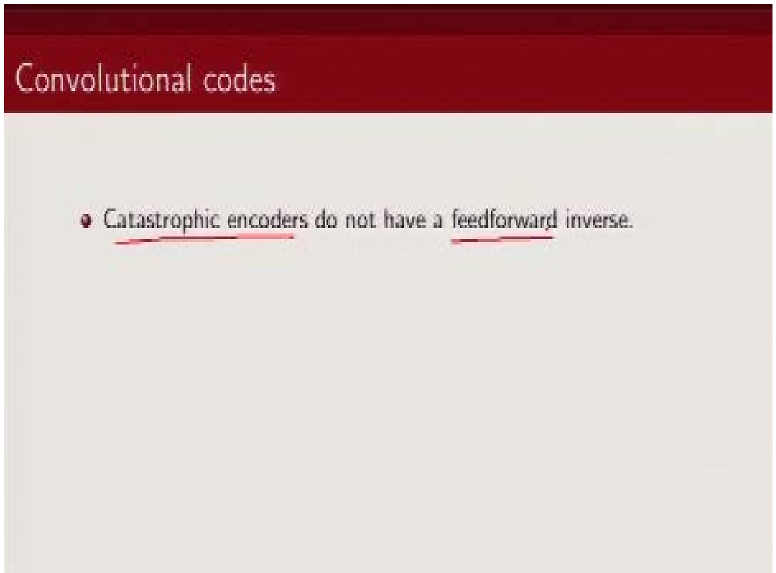
Inverse for convolutional code whose generator matrix is given by G(D) is given by this condition for rate 1/ n code and for a k x n code is given by this condition.

(Refer Slide time: 13:50)

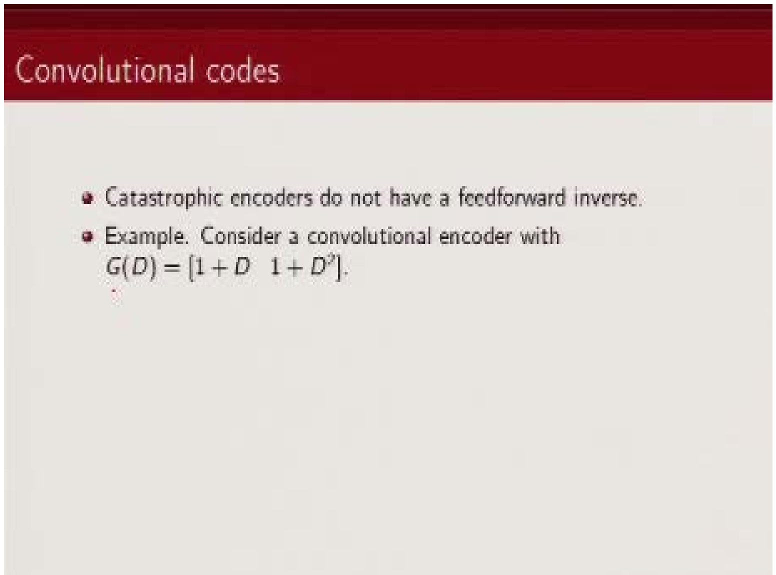


Now catastrophic encoders do not have a feedforward inverse, so for a catastrophic encoder we will just show you that their inverse has feedback terms.

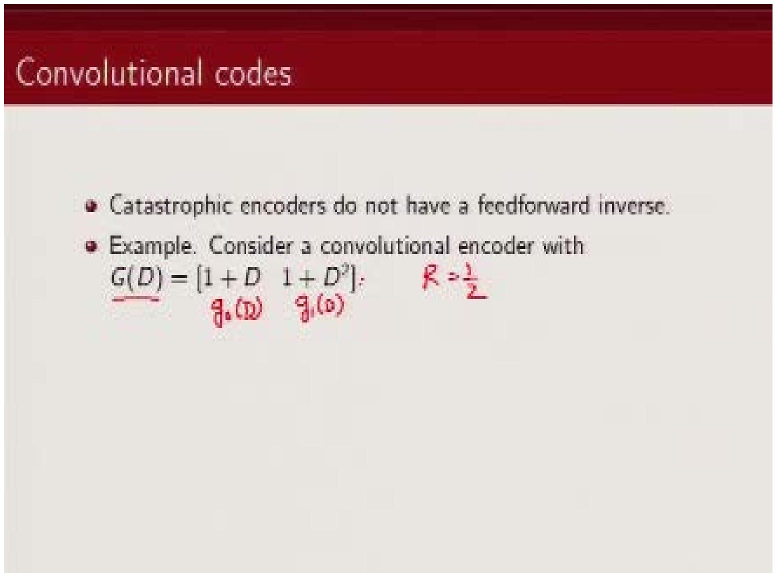
(Refer Slide time: 13:59)



(Refer Slide time: 14:08)

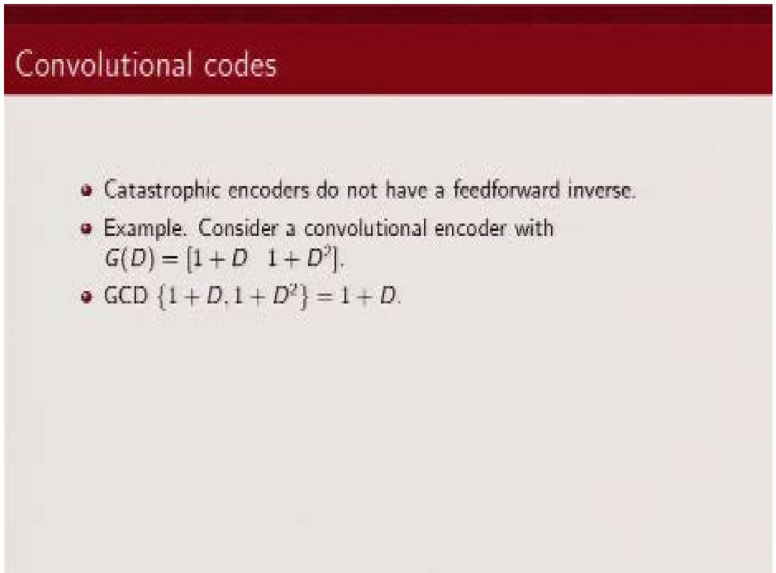


(Refer Slide time: 14:11)

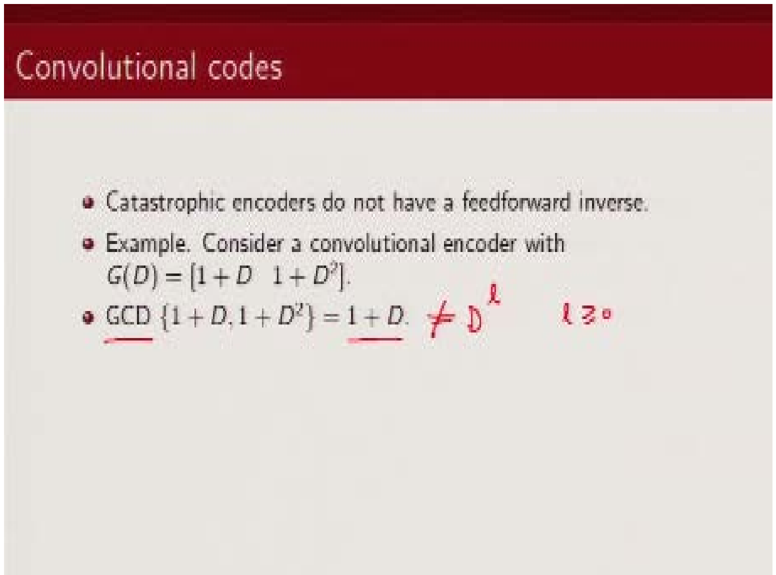


So let us look at one example, so let us consider a convolutional code whose generator matrix is given by this, so this is a rate R = ½ convolutional code and it has four states because the maximum degree of D is two, so the generator sequence is $g_0(D)$ is given by 1+D and $g_1(D)$ is given by $1+D^2$.

(Refer Slide time: 14:42)

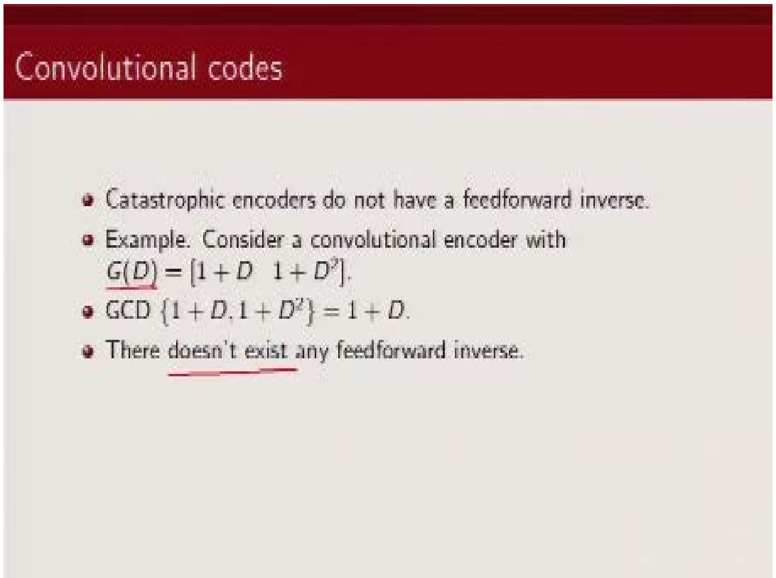


(Refer Slide time: 14:45)



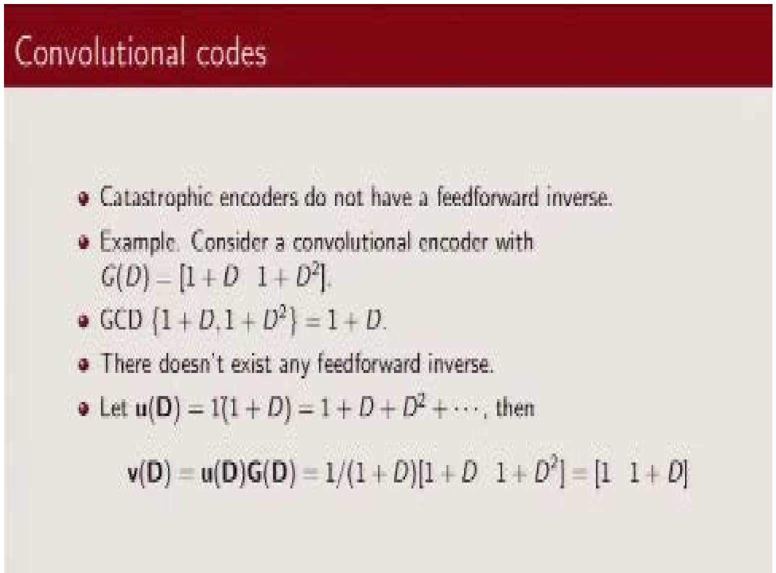
Now first thing that we check is, what is the greatest common divisor among g0 and g1, as it turn out in this case the greatest common divisor is 1+D so that is not same as D^l for some $l \geq 0$, so that means for this generator matrix we do not have any feedforward encoder inverse.

(Refer Slide time: 15:14)

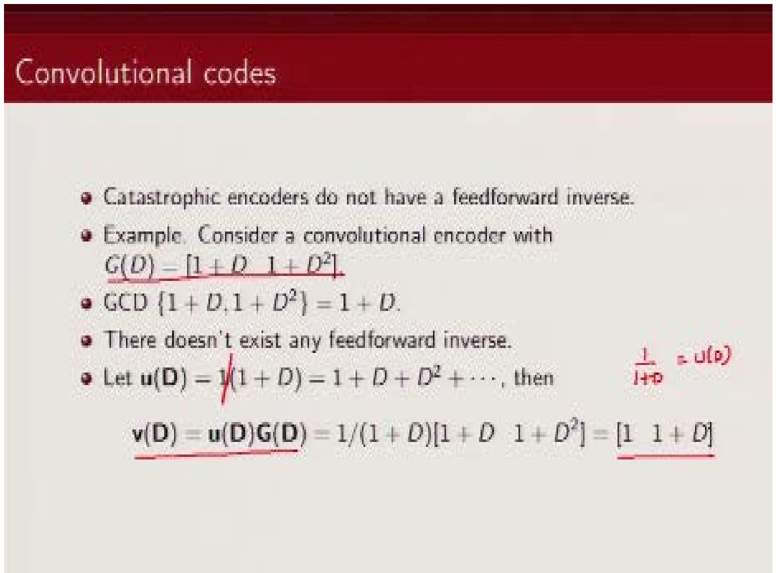


So there does not exist any feedforward inverse for this particular convolutional code with generator matrix given by this.

(Refer Slide time: 15:25)

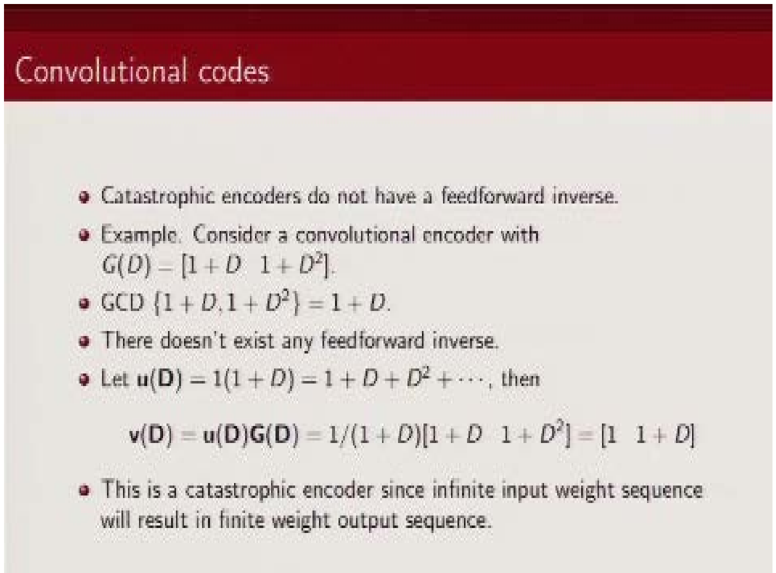


(Refer Slide time: 15:30)

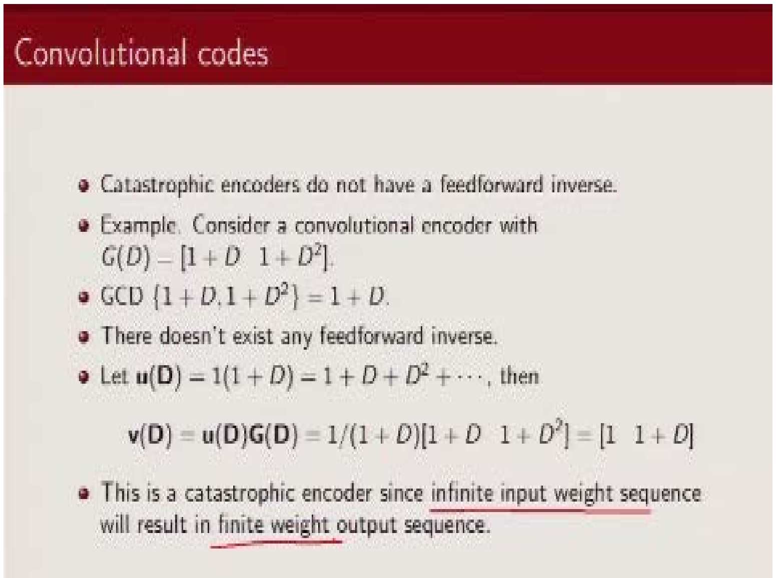


So let us take an example of u(D) given by 1/ 1+D this is a typo, this is 1 1/ D, now 1 1/D can be written as 1+D+D2+D3 so this is an all one sequence so our input is all one sequence which can be written as in this D notation it can be written like this, this may u(D), now if I give this input to my convolutional code whose G(D) is given by this what is my output? My output is u(D) times G(D) so this will be given by this. So note I just, my input has infinite weight but the output only has weight three and this is precisely an example of a catastrophic encoder.

(Refer Slide time: 16:24)

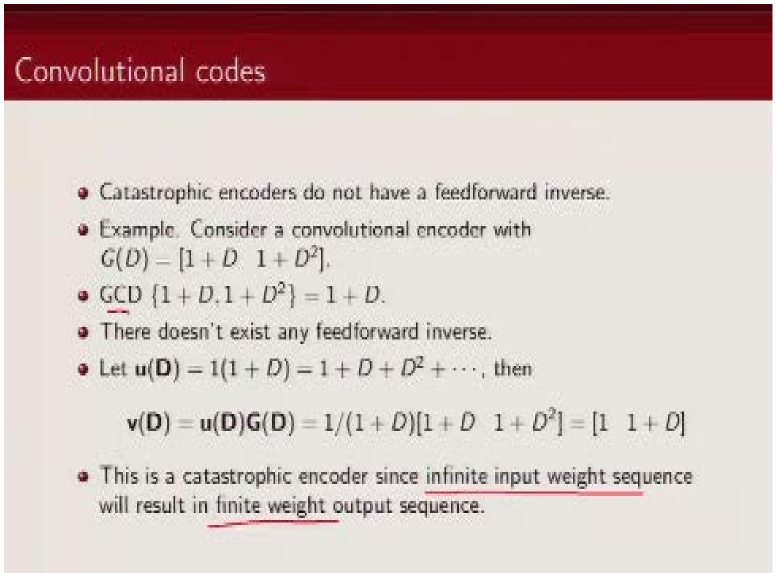


(Refer Slide time: 16:28)

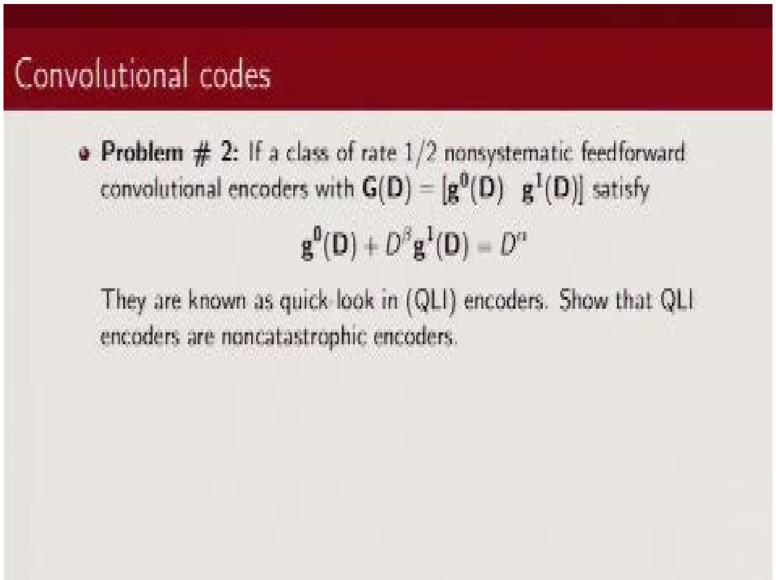


So you can see my input has infinite weight but my output has finite weight, so catastrophic encoder would not have a feedforward inverse as this condition is violated.

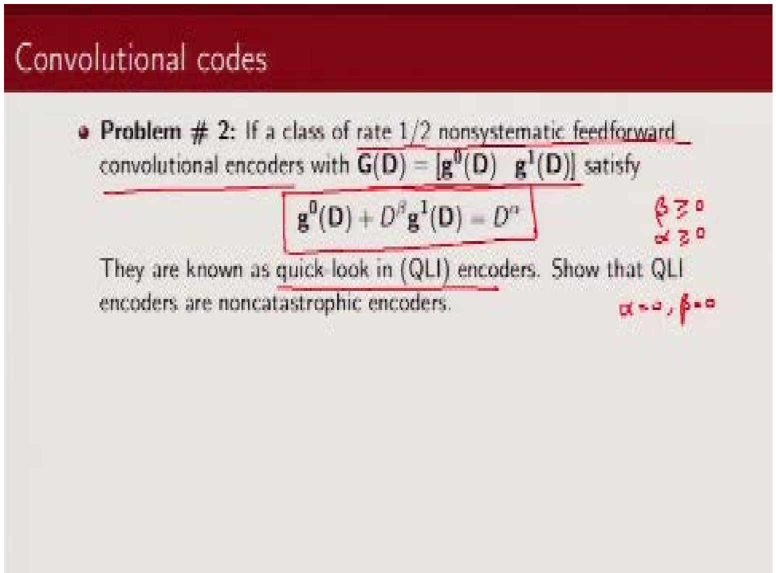
(Refer Slide time: 16:39)



(Refer Slide time: 16:44)



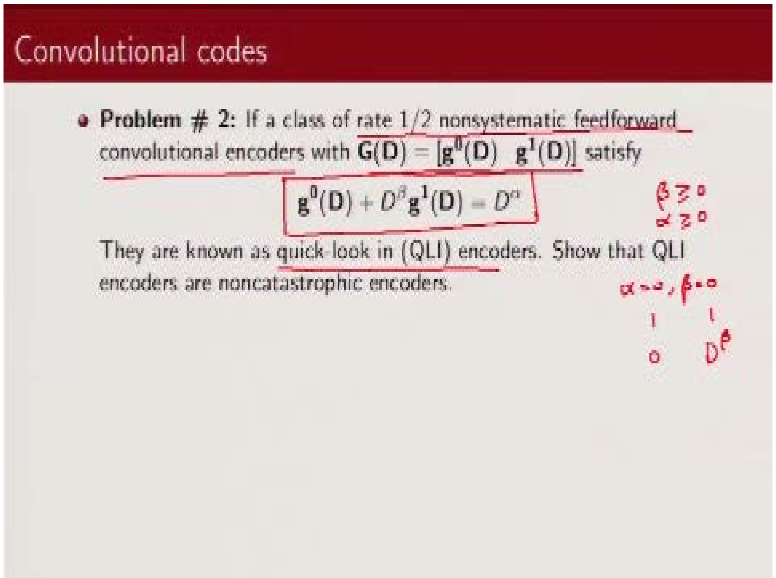
(Refer Slide time: 16:55)



Next we look at a class of rate 1 ½ nonsystematic encoders so we are looking at a rate 1 ½ nonsystematic feedback feedforward encoders whose generator matrix is given by this, and these generators g^0D and g1D satisfy this property. So what is this property, it says g^0D plus some delay thus ß ≥ 0 so some delay of g^1D is given by D^{α} where α is also something greater than 0, okay. Now let us take a simplified case am I let us say α = 0, ß = 0 so what does it says, it says $g^0D + g^1D$ is 1.

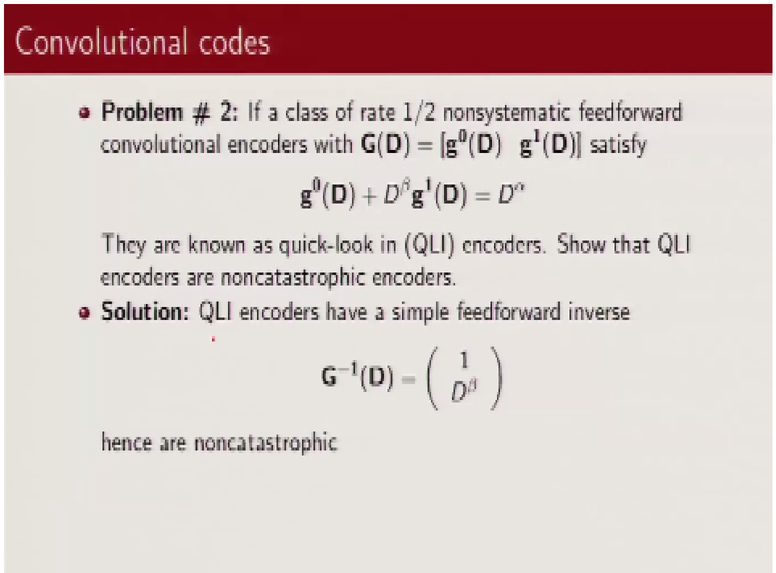
So then I can essentially, looking at these generators I can essentially find out that this encoder has a very simple encoder inverse so which is just one and one, if α is this so these are known as quick look in encoders, why they are called quick look in encoder because quickly looking at this encoder you can actually easily find the encoder inverse and essentially encoder inverse just consist of two tabs, so in some sense for a systematic encoder the inverse is a form 1 and 0.

(Refer Slide time: 18:40)

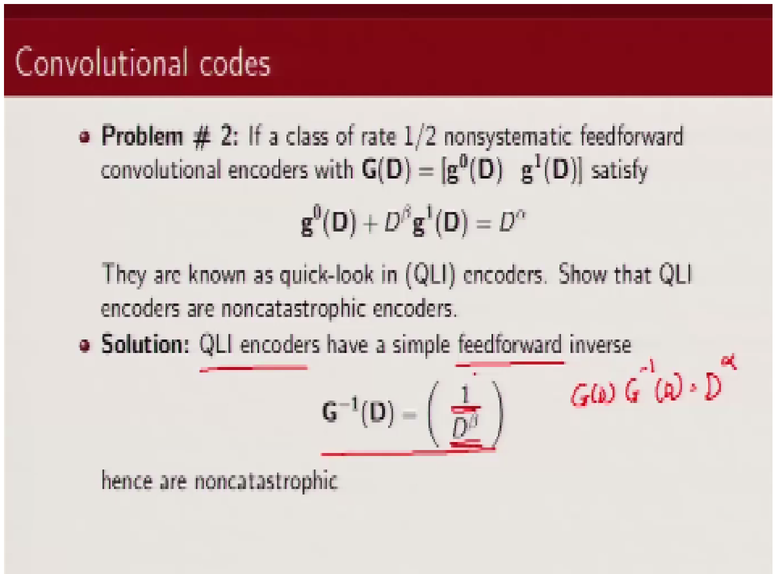


For a rate ½ code and here at they are of the form 1 and D^{β} so they are in some sense closest to systematic code if you want, like to call them.

(Refer Slide time: 18:53)

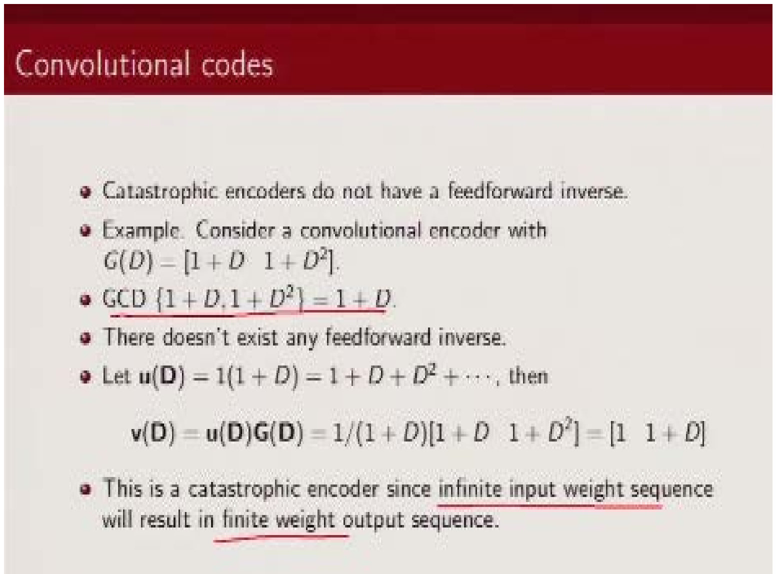


(Refer Slide time: 18:57)



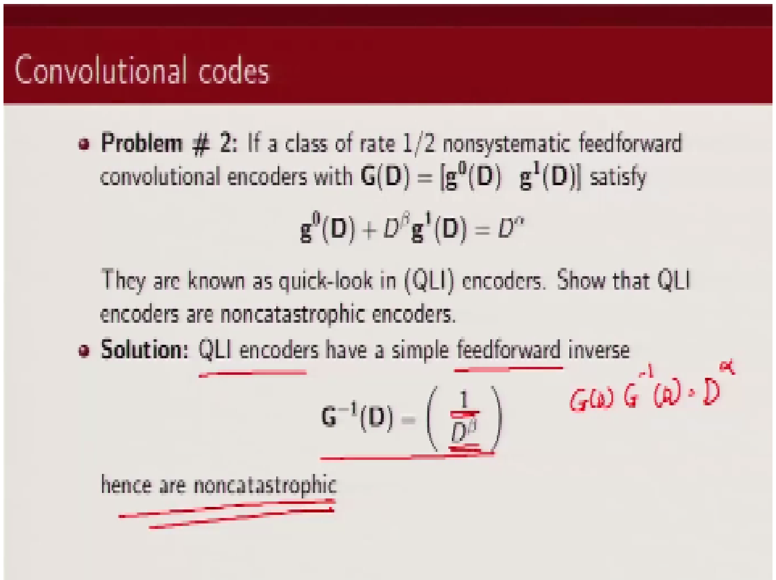
So this quick looking encoders have a very simple encoder inverse, a feedforward encoder inverse and that is given by this and you can verify that G(D), $G(D)^{-1}$ is your D^{α} , okay. Now note that the encoder inverse of quick looking encoder has just two tabs 1 and this D^{β} and it has a feedforward inverse so this cannot be a catastrophic encoder.

(Refer Slide time: 19:39)



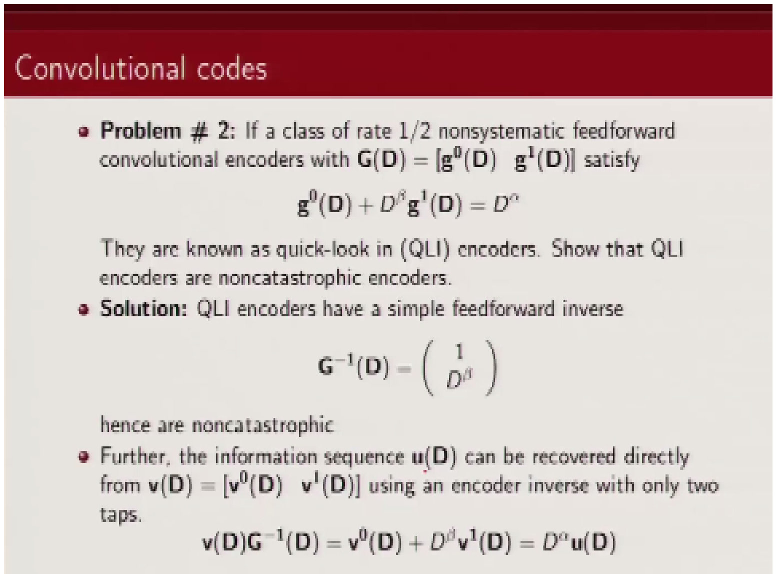
We just showed in the previous slide that a catastrophic encoder does not have a feedforward inverse.

(Refer Slide time: 19:47)



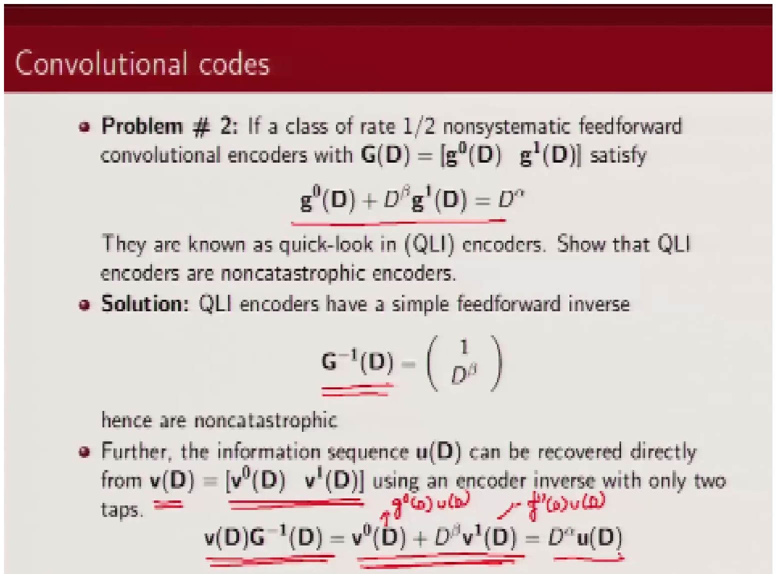
And since this has a feedforward inverse this cannot be a catastrophic encoder so because they have a feedforward inverse they are not catastrophic.

(Refer Slide time: 19:58)



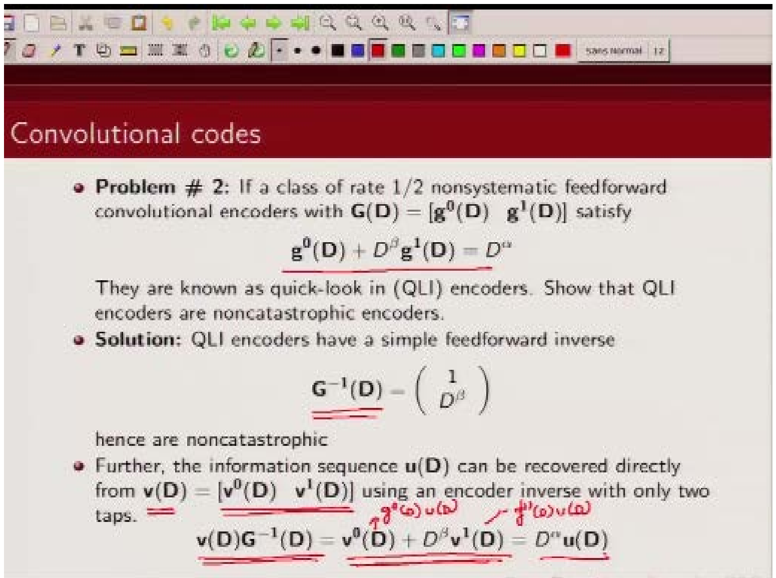
And as I said you can very easily recover back your information sequence.

(Refer Slide time: 20:06)



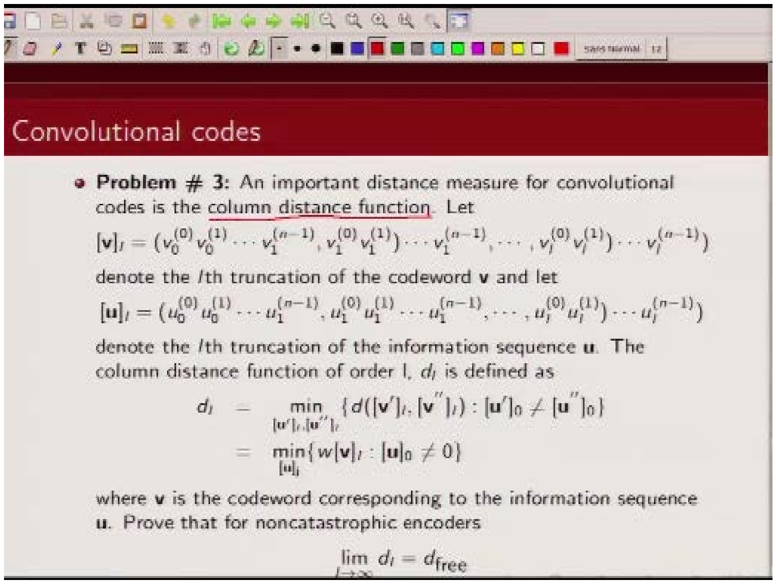
By making your coded sequence pass through this encoder inverse, so if your output sequence is given by v(D) which is this then once v(D) passes through this encoder inverse may encoder inverse what we get is $v^0(D) + D^{ß}v^1(D)$, now we know that quick looking code have this property that $g^0(D) + ßg^1(D)$ is D^{α} and $v^0(D)$ is this is equal to $g^{0(D)\ u(D)}$, similarly this one is $g^1(D)$ times u(D). So from this condition and from here this will come out to be $D^{\alpha}u(D)$. So among this class of nonsystematic encoders quick looking encoders have a very simple encoder inverse circuit.

(Refer Slide Time: 21:25)



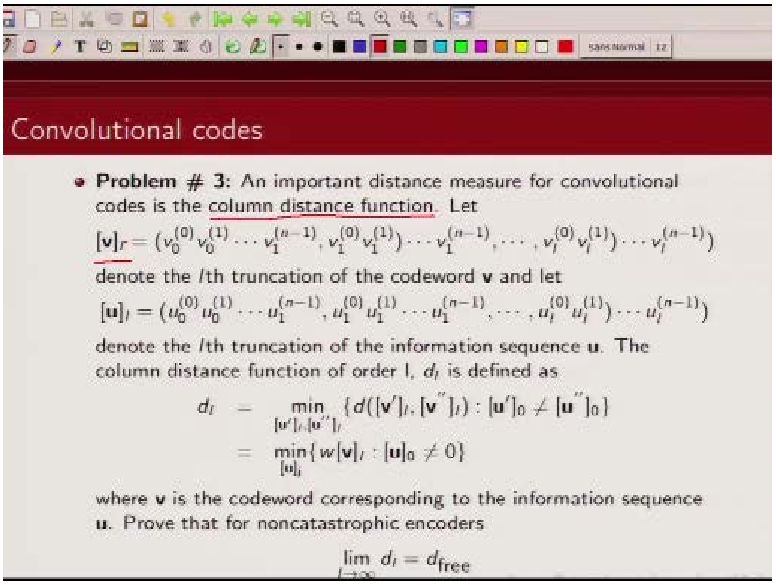
And one can easily find out what the information bits are from the coded bit without decoding.

(Refer Slide Time: 21:34)



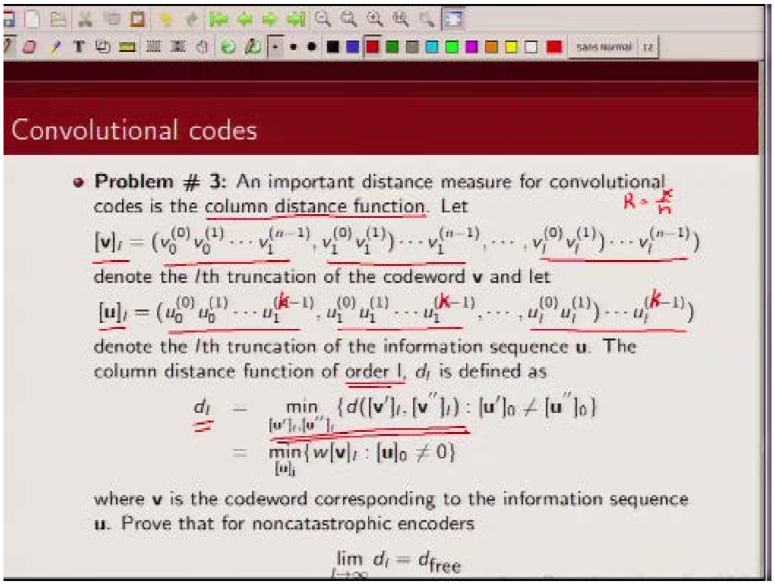
The next problem that we are going to talk about is about a distance measure for convolutional code. So we will first define what we mean by column distance function. As we know a convolutional encoder can continuously encode an information sequence. So we can have an infinite length input sequence and correspondingly an output sequence.

(Refer Slide Time: 22:07)



Now we define the column distance function for a convolutional code as follows. So before that I am describing output code sequence v which is truncated to up to length l. So this notation that you see $[v]_l$ it shows essentially our code word up to time l and what is our code word up to time l?

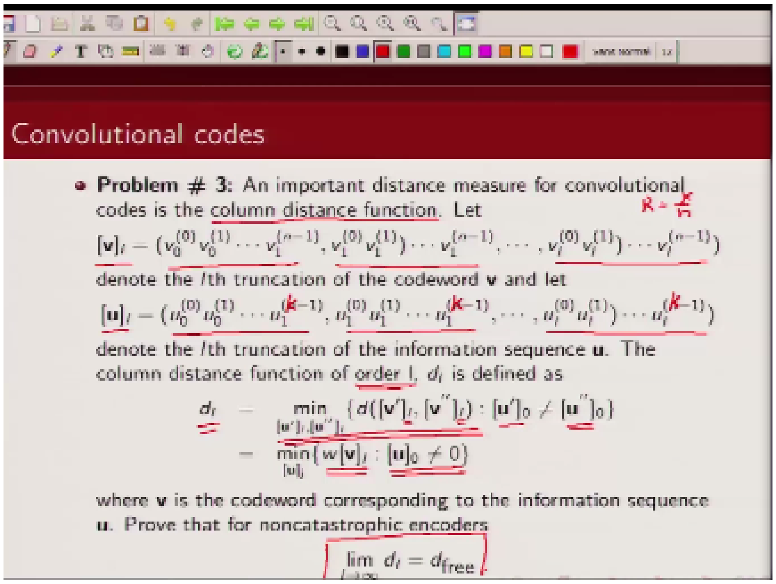
(Refer Slide Time: 22:38)



So this will be $v_0^{(1),}$ $v_1^{(n-1)}$ then v_1 v_n because it is a rate, this is a rate let us say this is a rate 1/n code then this is for first time instance you have n bits, second time instance you have n bits, and then for l time instance you will have n bits. So this is your truncated code word up to time l. Similarly I can define my truncated information sequence okay, so just a minute, this is a typo, this should be k-1, this should be k-1, k-1 and of course if k is 1 there will be just 1, input so you have for first time instance you have k inputs, second time instance you have k input as similarly for l time instance you have k input.

Now note that both your information and coded sequence is truncated up to length l. Now we define column distance function of order l as follows. It is the minimum distance between two truncated code words of length l.

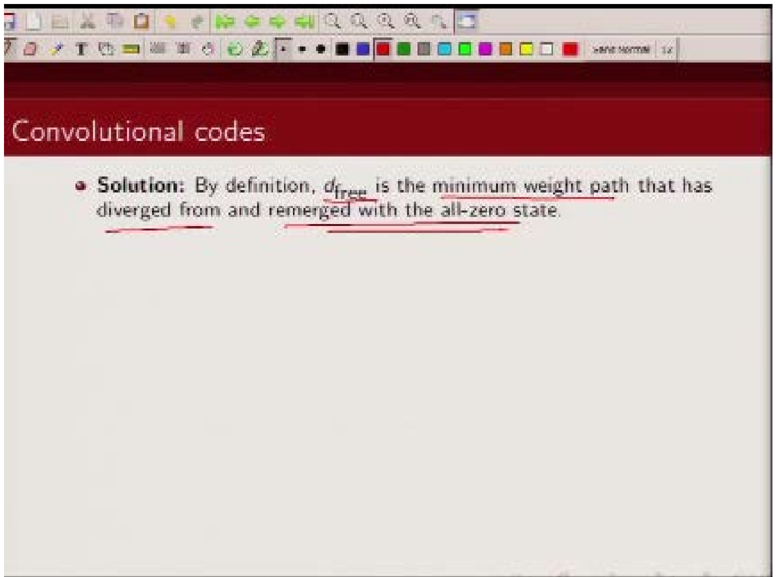
(Refer Slide Time: 24:05)



Such that so you can see it is a minimum distance between two code words v′ and v″ both of length l such that u′and u″ they are not same, so it is, it is essentially hamming distance between two truncated code sequences. Now we know that hamming distance between two sequences minimum distance can be written as minimum weight of a non zero code word. So the same thing we can write as minimum weight of a l th truncated code word belonging to a non zero information sequence.

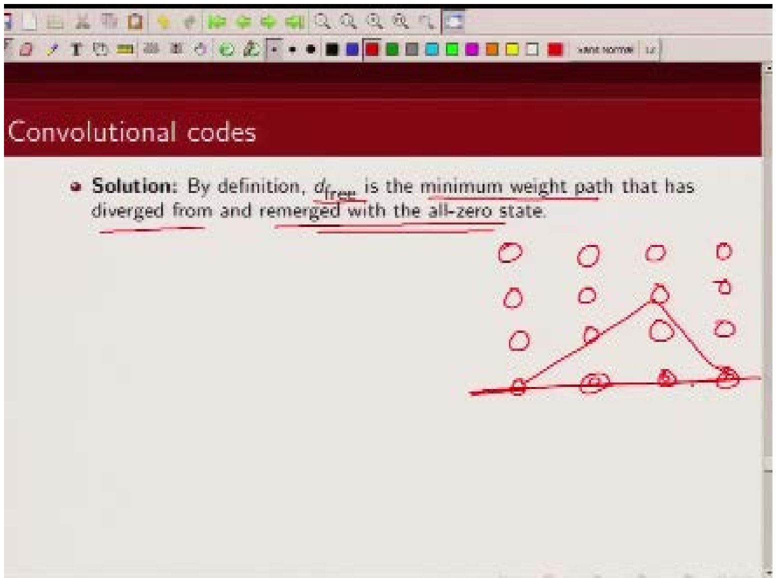
So we can define our column distance function of order l as minimum weight of l th truncated code sequence belonging to a non zero information sequence. Now the thing that you have been asked to prove here is show that as l goes to infinity this column distance function tends towards free distance of the convolutional code. In fact after three or four constrain length you will see the distance reaches d free and then its remains there.

(Refer Slide Time: 25:41)



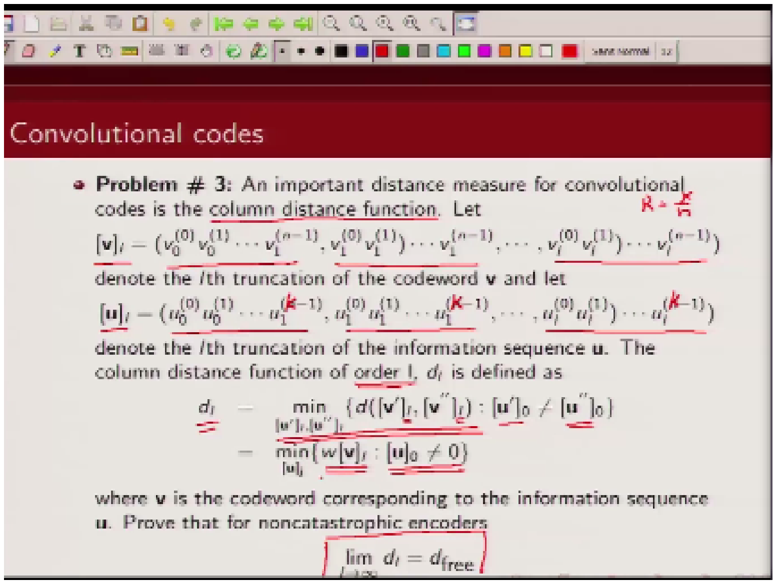
Column distance function, so by definition what is a free distance of convolutional code it is the minimum weight path that has diverged from an all zero state and merged back into all zero state. How do we find minimum weight code word, minimum length code word so it is, so if you have convolutional code without loss of generating let us say we are transmitting all zero code word, then minimum weight code word will be the length of the minimum weight along all non zero.

(Refer Slide Time: 26:24)



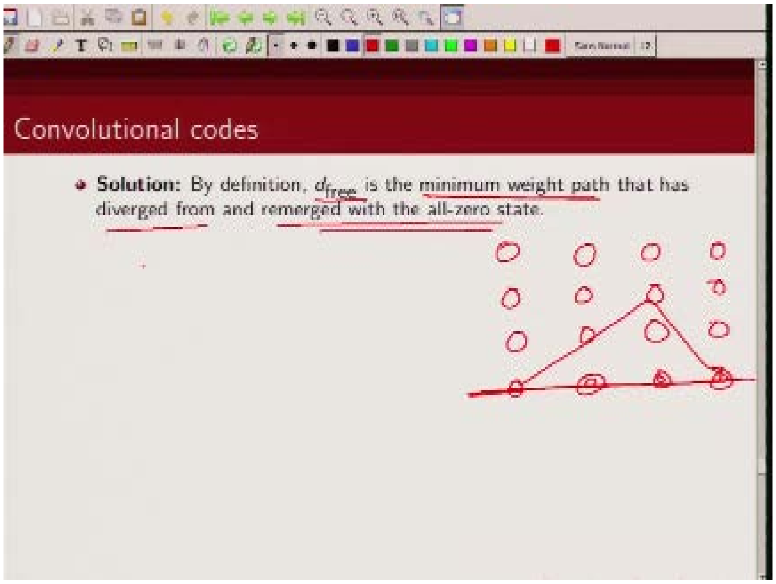
A part that goes through non zero state, so let us say you have some convolutional encoder, some four set convolutional encoder and this is let us say your all zero state, all zero state. So, so all zero state, and let us say you have some diversion from this and then you coming back so this is your all zero state and what is your

(Refer Slide Time: 26:59)



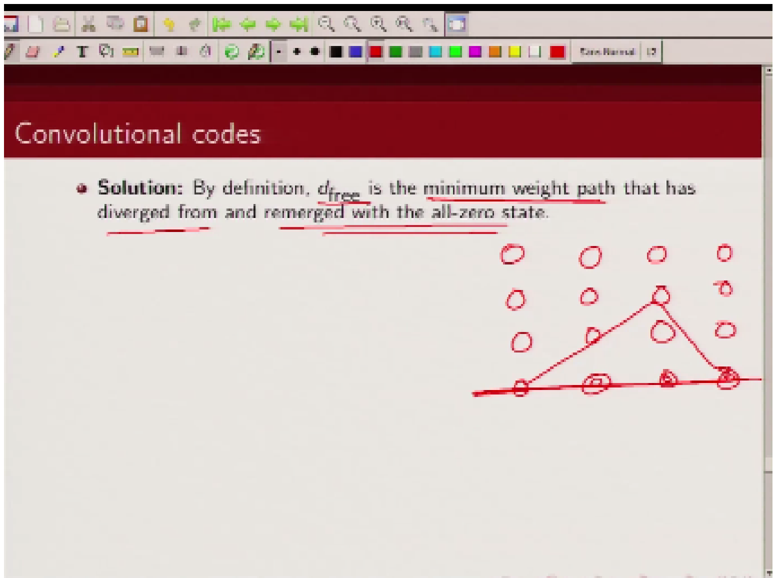
Column distance function, it is a minimum weight of your code word belonging to non zero information sequence.

(Refer Slide Time 27:12:8)



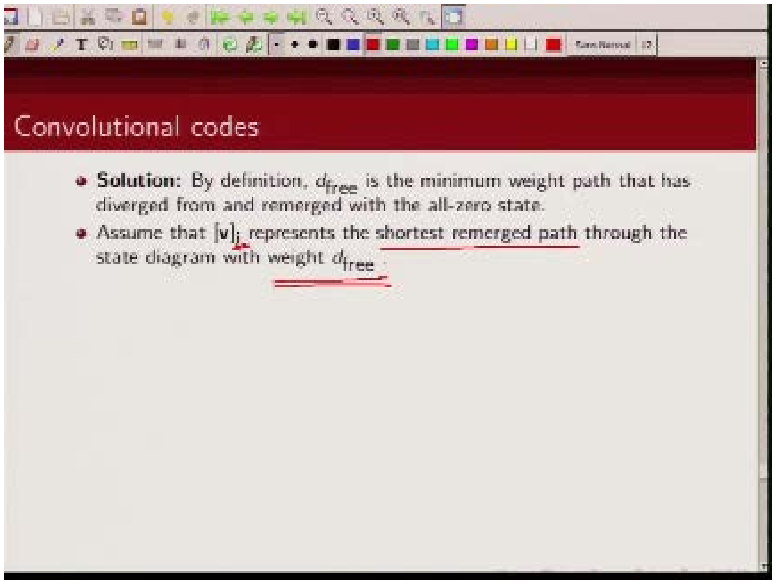
And what are the paths through the trellis diagram, these are all our valid code words so we need to find a path through the trellis which has minimum weight, so and that would be our free distance so it is the free distances minimum weight path that has divert from all-zero state and merge back right.

(Refer Slide Time 27:38:2)



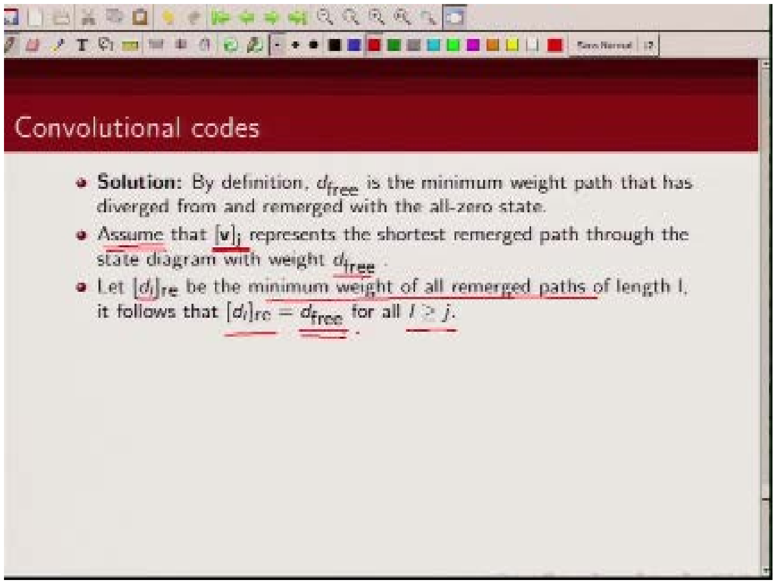
So that to get a non zero weight you essentially diverged from all, all-zero state and then merge back to all zero state.

(Refer Slide Time 27:47:2)



Now let us assume that at time T=J so [V]j represent the shortest re-emerged path through this trellis diagram or straight diagram which has weight of d free so if T=J is the smallest times which represent the shortest re-emerged path through this trellis diagram.

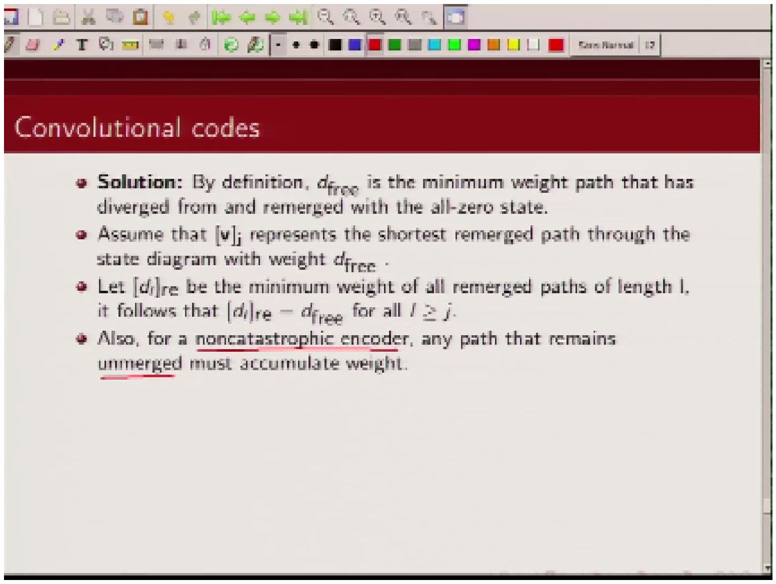
(Refer Slide Time 27:16:9)



And what does it mean, it means if we denote by DL the minimum weight of all re-emerged path what is re-emerged path, so these paths which are diverging from all zero state and then merging back into all zero state, these are our re-emerged path. Now what we are saying is for T=J that is the smallest re-emerged path which has weight equal to dfree, so if you have any time any J which is > than any time just > than this J.

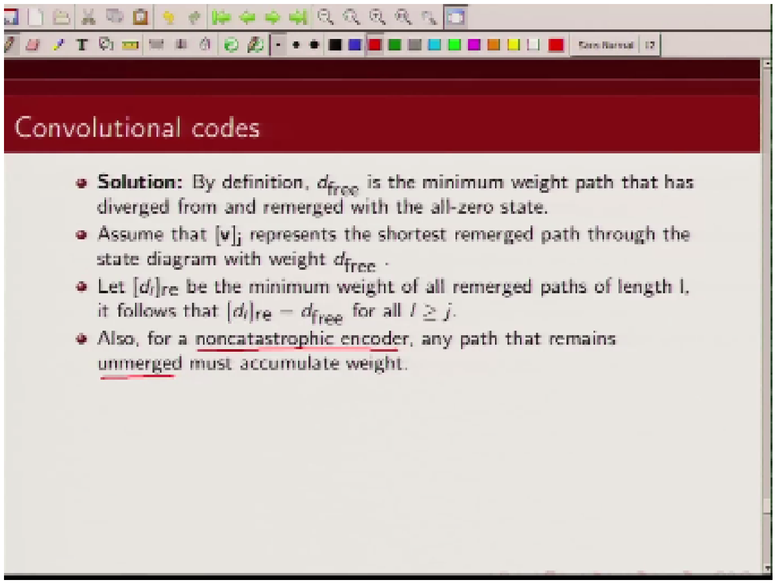
Then your column, this is column function will be equal to the free distance. Why this is so because we have said that for time = J that is the shortest re-emerged path through this trellis diagram which has free distance which has rate = free distance, so if we take any time larger than that then of course we will have a re-emmerge path having minimum distance.

(Refer Slide Time 29:40:3)



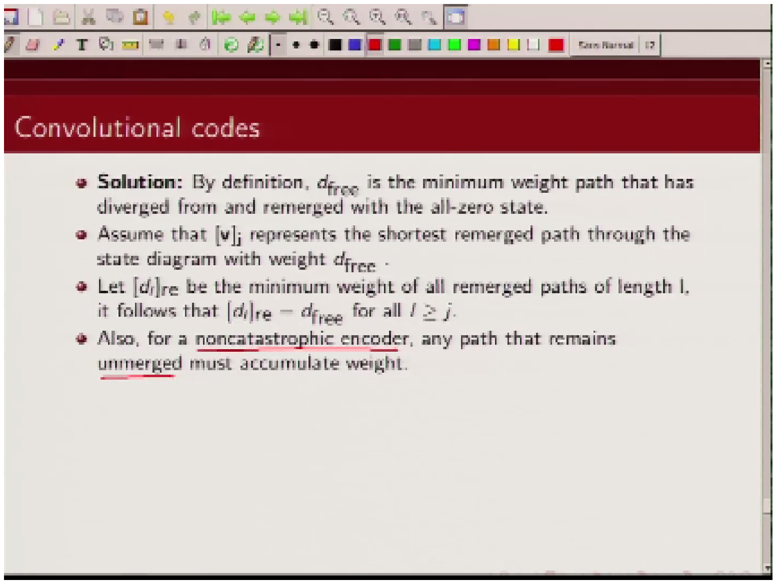
Please d free, now if there are any non merge path what are non merge path, so these paths which are diverged from all zero state but have not yet merged to all zero state so those are unmerged path. Now for a non-catastrophic encoder any path that is not merge must accumulate to it, so only for the catastrophic encoder we have situation where input weight is higher and output.

(Refer Slide Time 30:16:3)



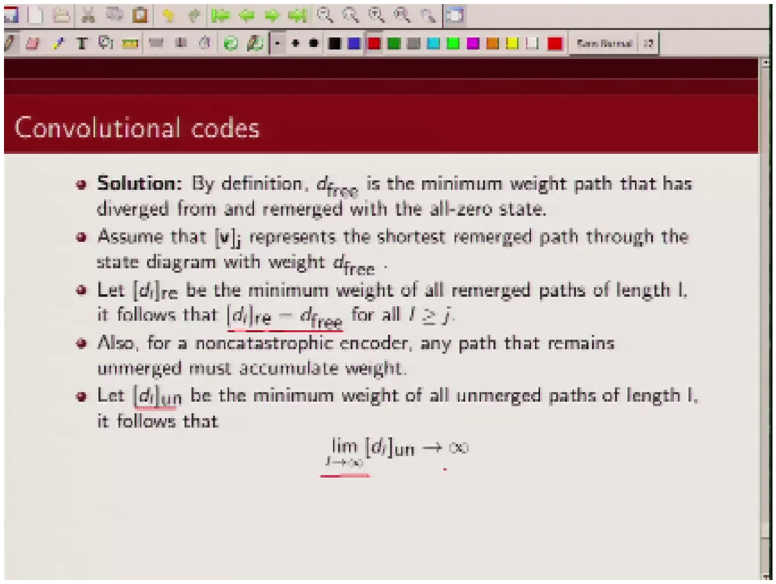
Weight is smaller but if it is a non-catastrophic encoder it will accumulate weight.

(Refer Slide Time 30:23:1)



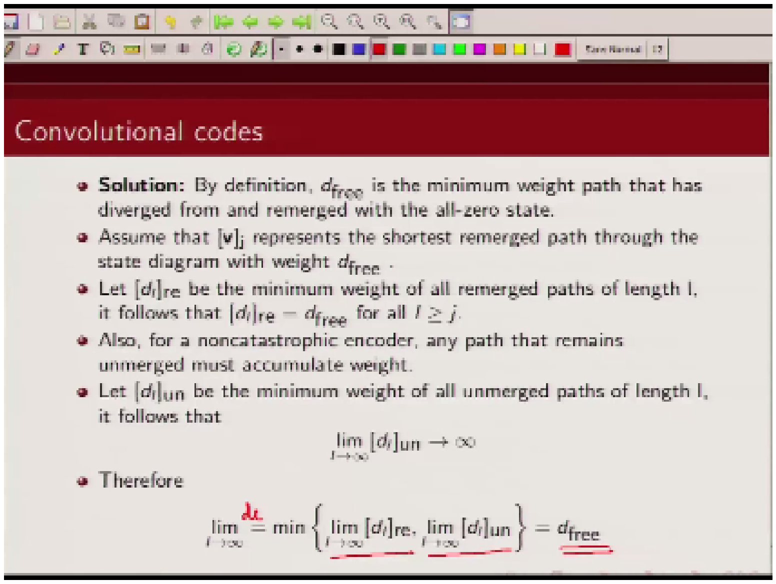
So if we have one non systematic encoder any path which has not yet merged with all zero state will try to accumulate more and more weight.

(Refer Slide Time 30:35:1)



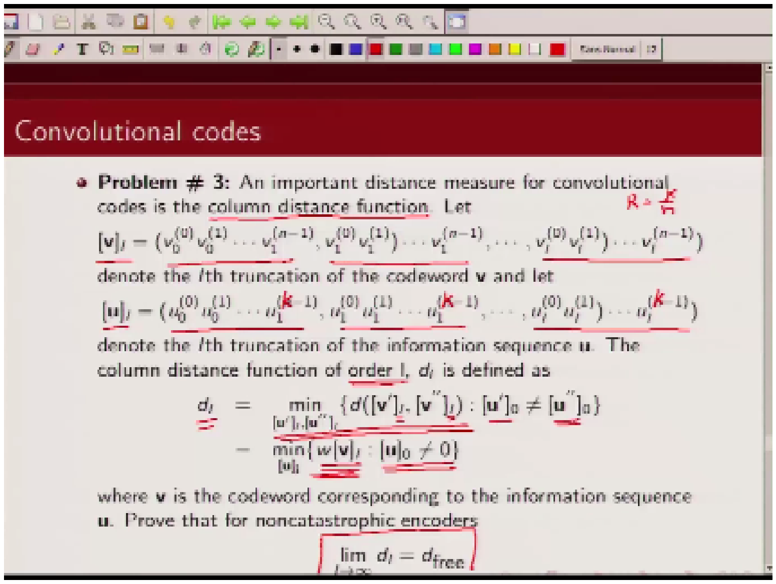
So what is going to happen, so if we look at distance for column distance for unmerged path then as L tends to infinity this distance will also grow, this also go to infinity because it is a non catastrophic encoder. So what we have shown is so for L greater than J, for all re-emerged path this column distance function is dfree and for unmerged path this is going to be infinity as L goes infinity.

(Refer Slide Time 31:17:7)



Hence we can say that limit D(L) is minimum of the column distance for re-emerged path or unmerged path. This is infinity, this is d free, so we know that as L tends to infinity the columns distance, column distance function will be.

(Refer Slide Time 31:47:0)



dfree so this proves that columns distance function will go to dfree as L goes to infinity, thank you.

Acknowledgement

Ministry of Human Resource & Development

ss

Prof. Satyaki Roy

Co-ordinator, NPTEL IIT Kanpur

NPTEL Team

Sanjay Pal

Ashish Singh

Badal Pradhan

Tapobrata Das

Ram Chandra

Dilip Tripathi

Manoj Shrivastava

Padam Shukla

Sanjay Mishra

Shubham Rawat

Shikha Gupta

K. K. Mishra

Aradhana Singh

Sweta

Ashutosh Gairola

Dilip Katiyar

Sharwan

Hari Ram

Bhadra Rao

Puneet Kumar Bajpai

Lalty Dutta

Ajay Kanaujia

Shivendra Kumar Tiwari

an IIT Kanpur Production