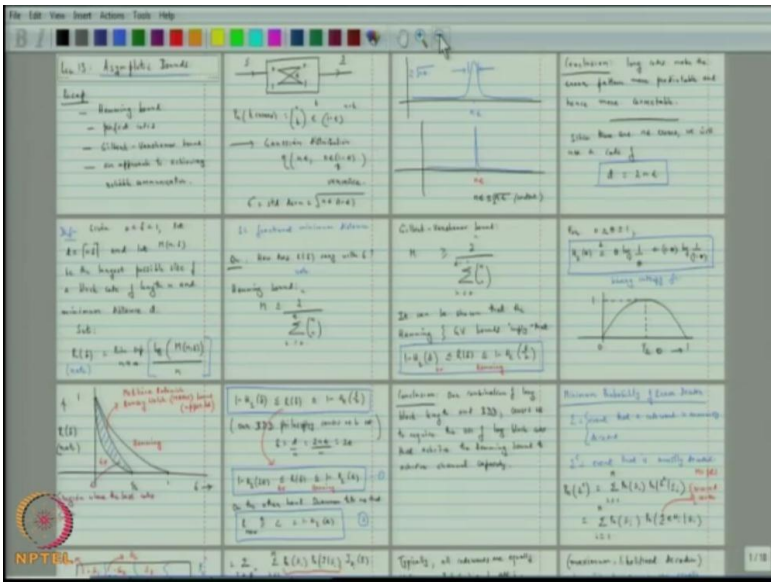
Prof. Dr. P. Vijay Kumar
Error Correcting Codes
Electrical Communication Engineering
Indian Institute of Science, Bangalore

Lecture No. # 14
Standard Array Decoding

(Refer Slide Time: 00:21)



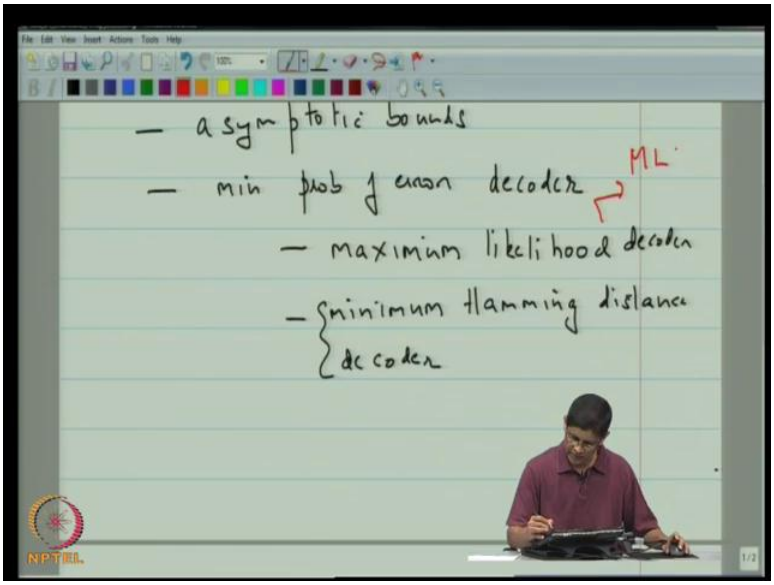
So, last time we will talking about asymptotic bounds. We should that for very large block lines there is a certain regions, where the best quotes at to be form. And I want well on that, since it is that is lecture is valuable to you. And after finishing the section on asymptotic bounds, we went on to talk about decoding techniques. Then we said that principle goal in decoding is to actually try to minimize the probability that you decoded the code word error, which is very natural.

And then we saw that the minimum probability of error decoder. First of all, reduces under the assumption that all the code words are equally likely, which is often the case to maximum likelihood decoding.

That is you choose that code word, which is such that why given excess the most? y being receive vector, x being the code word. And in turn, we reduce that to the minimum distance decoder. That is you simply decode to the nearest code word in nearest in the sense of hamming distance. And I think this is something that we discuss right in the beginning, and in a sense you

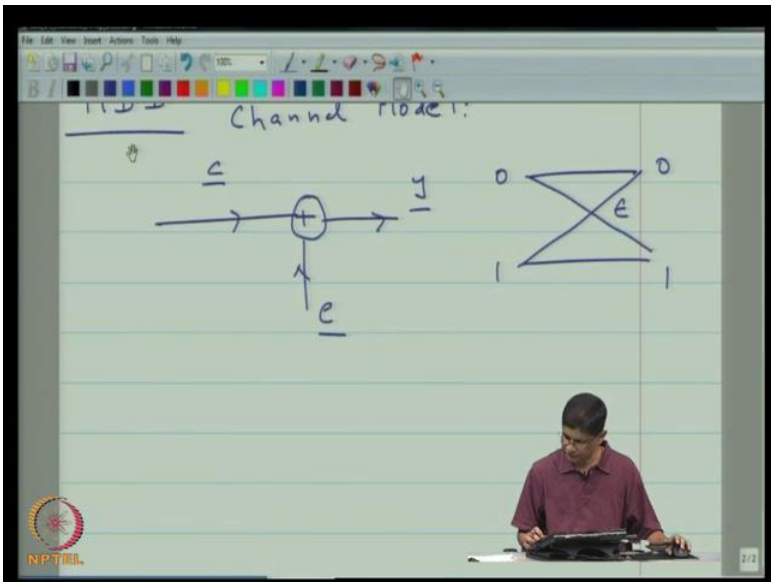
would have said that all obvious and that is true. It was obvious even then; that we needed to actually go through the motions are actually proving that. So, them that previous let us go back to that review. Let us go back to are present lecture and I am going to call this standard array decoding, so this will be are lecture fourteen.

(Refer Slide Time: 02:03)



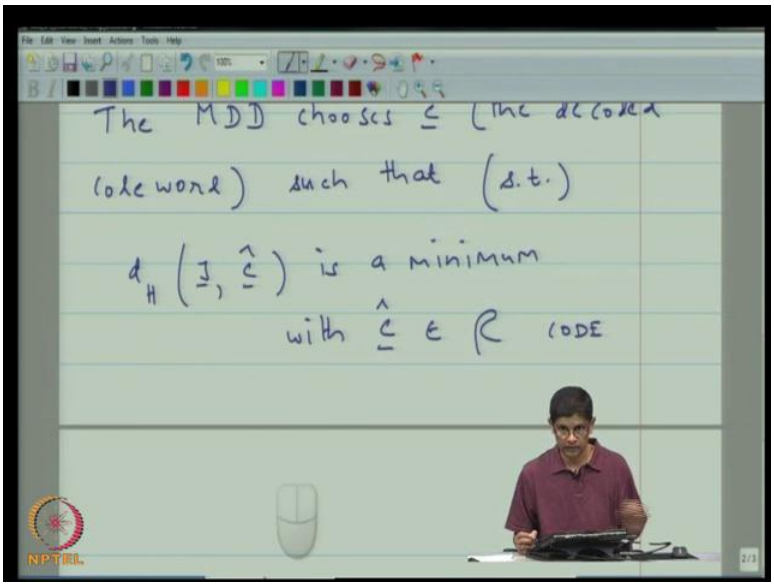
So, recap from the last lecture, we looked at asymptotic bounds, and then we talked about the minimum probability of error decoder. And this letters in turn to the maximum likelihood decoder, and again in turn this letters again in turn to the minimum hamming distance decoder. And as deviations, let me a deviate and it write this as MLD; that is very commonly employed. And will refer to this as the MDD are the Minimum Distance Decoder.

(Refer Slide Time: 04:13)



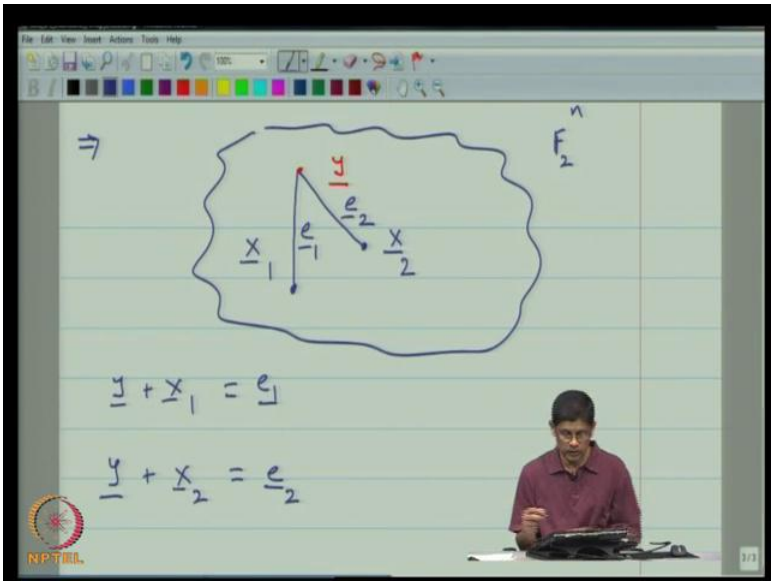
So, as the starting point is in fact the minimum distance decoder. Let us adopt the following channel model that is that new transmitter's code word c. They additive noise, that is additive it and then what we actually receive is y. Now, we are still operating our banner symmetric channel, where the cross over probability epsilon. And in make one change living call this vector e. And basically, the way to make a connection between the banner symmetric channel model here, and this additive channel model here is j simply to say that this error vector as value equal to one precisely, when this is an error to over the channel in transmission over the channel.

(Refer Slide Time: 05:40)



The minimum distance decoder chooses c hat and the c hat we will mean the decoded codeword. Such that and have I told you that I will deviate this by s dot t dot. Such that, the hamming distance between y and c hat is a minimum under the restriction; that c hat belongs to the code. This script c here denotes the code. Now in turn, this is equivalent to sign and I will illustrate this with the picture.

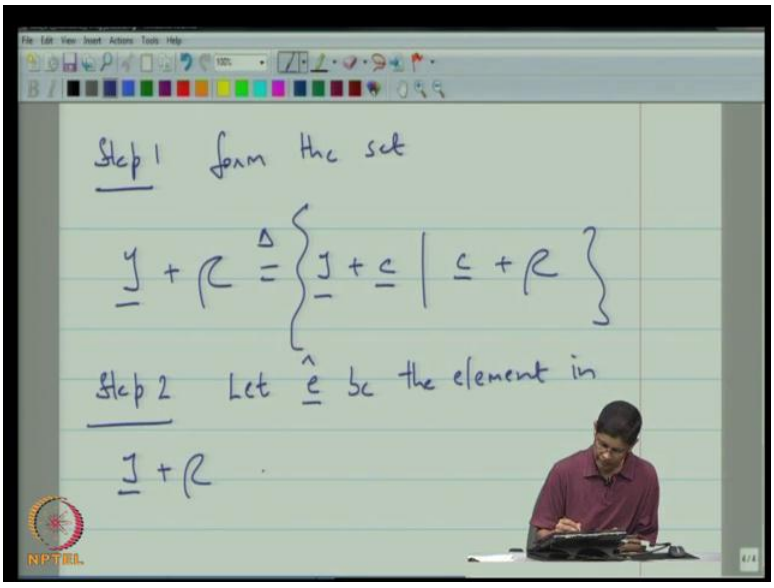
(Refer Slide Time: 07:00)



Let say, you have the set of all and topples here. And somewhere in here, we have the receive vector. And the MAB competing code words, this is the code word x1, and this is the code word x2. And what you are trying to do is your trying to decode to that code word which is closes in hamming distance. So, what we do is, we compute one way of finding the closes code word is to actually compute y plus x1 and let say that, this vector is even. So, we could compute y plus x1, we could also compute y plus x2 and let say we call this e1 and this is e2. The reason that we are writing e1 and e2 is, because these I mean if x1 was the transmitted code word, then e1 would actually be the corresponding error vector. So, that sense when we add the excess to the y is what are we, what we are actually getting is the list of the potential error patterns.

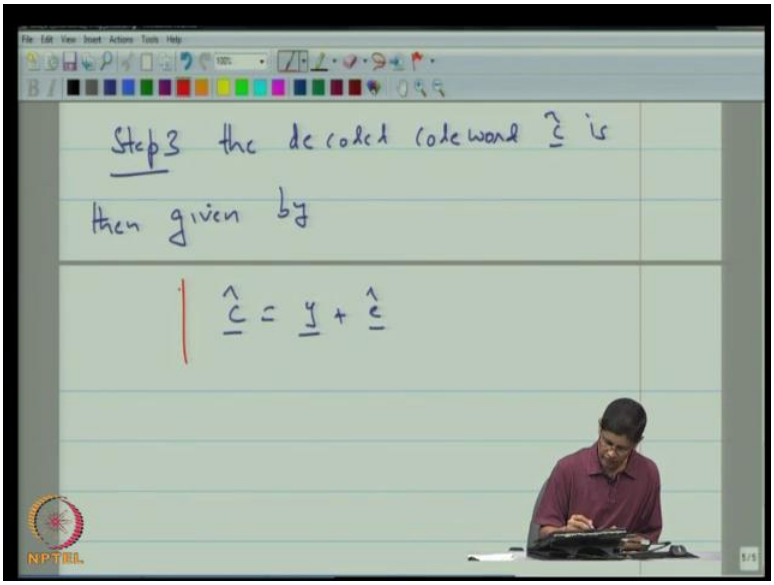
And now, what we want to see is which the closes code word is? But that is completely equivalent to sign, what is the error pattern? Which has the least hamming weight? So, we now made a shuttle shift and instead of looking for the closes code word, we are saying the let us look for that error pattern, whose hamming weight is the least. And if we want to actually pictorial leave you this, you can think of this as a though as line joining these two. And this line represents e1 and this over line where this represent e 2 and what we really interested in doing as, fining that code word whose corresponding error vector as least hamming weight with that is background. Now I think you like agree that, what I am going to write down now is a completely equivalent decoding algorithm.

(Refer Slide Time: 09:20)



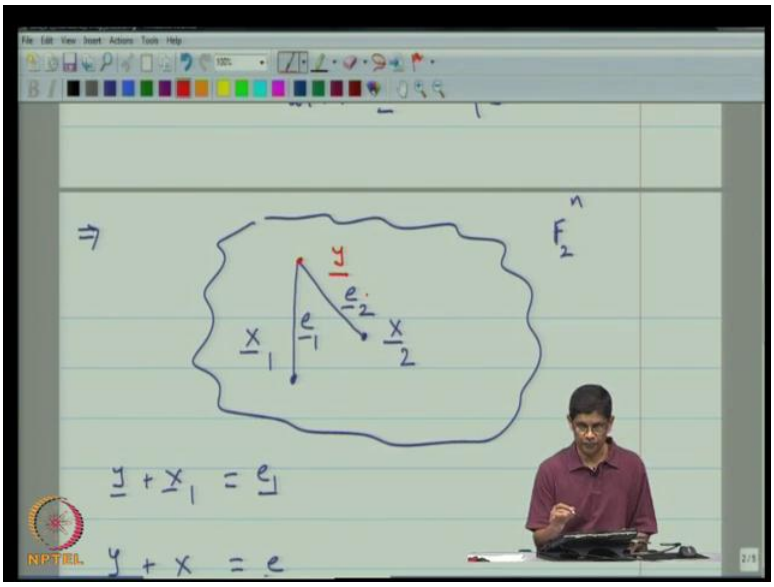
Thus the decoding algorithm, this is the minimum distance decoding algorithm can equivalently be phrased as follows: step 1, form the set y plus c which is defined to be y plus c where c belongs to the code. So, this is the actually a set and which is of the ten by adding to receive vector y, all possible code words. We form the set step 2, let now remember I told you that you think of each of these elements as possible error pattern, potential error patterns. Let e hat be the element in y plus c having least hamming weight. Step now, of course there is always these questions in your mind well, what is there are two vector whose hamming weight of the least? Then what we do is, in you can arbitrary toss of a coin. So, does it in really matter that one. So, with that in mind let us pertained that, this is only one.

(Refer Slide Time: 11:45)



Then, we gone to next step 3, the decoder code word code word c hat is then given by, c hat is equal to y plus e hat. And if you think about it in some sense, what we really or doing as you saying that, you know what trying to estimate code word is really the same as trying to guess, what the error pattern was. And in a sense it may be easier to actually find the error pattern, find out what the most likely error pattern is? And then added to receive vector to get back the code word. You might be a little bit possible, how come you are adding should into by subtracting. You should be subtracting, but the point is that when you doing arithmetic modulo two, we do not eliminate the difference for the adding or subtracting. And for that reason, we are did back.

(Refer Slide Time: 13:00)



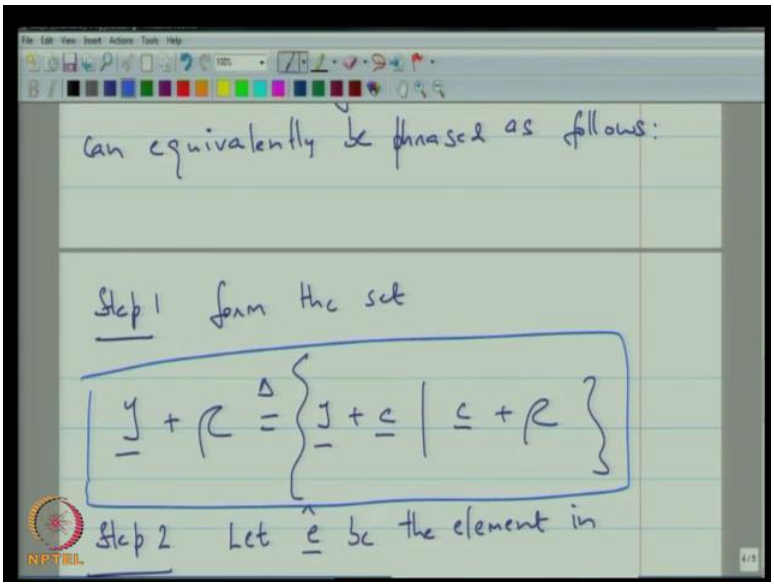
So, again just recap the idea is that, it is very clear from this picture here. Here is your received vector y and then you try to decide the closes codeword. By the way, I am interchangeably using x is and c is to represent code word, how fully the causes no confusing. So, it trying to find that code word which is closes to it, which is equivalent to sign your trying to find that distance vector which has least hamming weight. And each of this distance, difference vector e1, e2 represent potential error pattern.

So, we are trying to find the error pattern with least weight. And then you are going add it back to y to get estimated code word. And that is a slight shift in perspective, but convenient one. So, now you have this. Now this one thinks where I like to actually bring to your notice here. That is this set there are declared here and put down at set to be here. We actually encounter this, we count this way back in an earlier lecture, when where actually talked about co sets of a sub group. That time we wrote the co sets in a multiplicative form whereas here, where actually writing it in added to forms. So, may be a let see since we have the technology, where can I take you back to the lecture. So, that we take you quick look up at that.

Here we are an example. When we are talking about groups, co sets, sub-groups, rings and fields, then we have an example where we had groups, set of integer's mod six and H was the sub group consisting of 0, 2, 4. And then we said that the co sets of the group partition, the group, the sub-

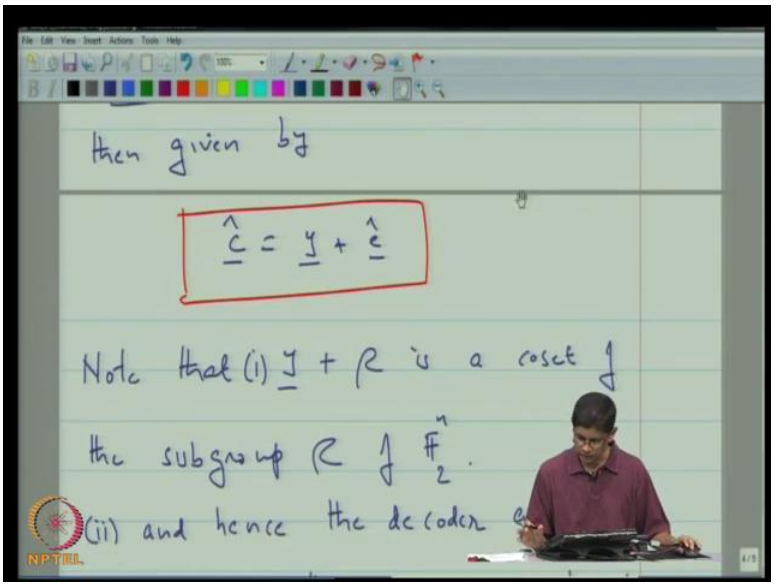
group so here the sub group is H which 0, 2, 4 all the even elements. And we have an even and the odd in the partition it. So, there we are introducing this notation H plus b.

(Refer Slide Time: 15:31)



So, there exactly the same situation, there you have a sub- group which is the code, and then you have adding an element to it. So, hopefully that put things back in perspective, so let us get back to our lecture. So here, the role played by H is now played by the code, so it the same sense. So, we will just make remark here.

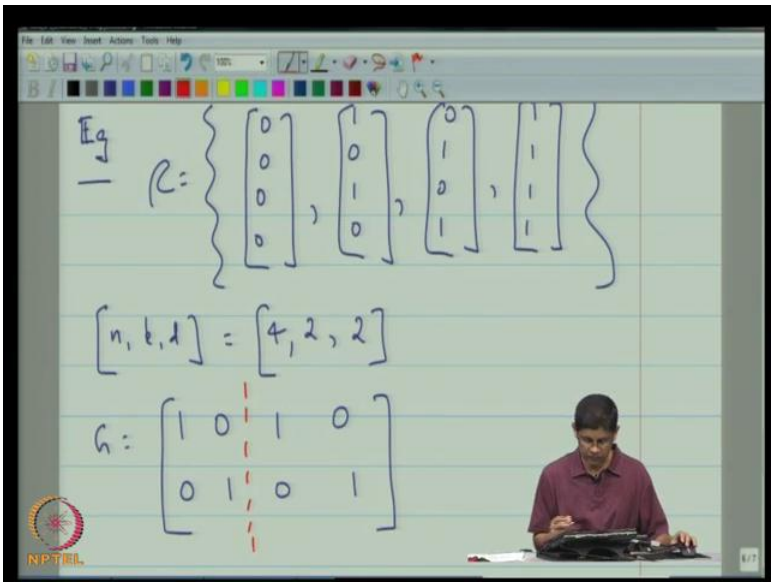
(Refer Slide Time: 16:00)



So, note that y plus c is a co set of the sub-group c of F2 to the n. If you look up at the set of all n topples, now that forms of group and additional modulo two component wise. And then the code c is itself a sub-group. In fact, all linear codes are always sub-groups even through that and therefore we looking at actually co set.

I think at this point it may be good to given example, but this in other when other point all make. Note that one, so let us call this one. Two and hence the decoder action is only a function of the co set of c to which y belongs and not to y itself. The reason being that what you do is, you form this set. And once you formed it, you form find the least the vector having least weight which in this case happens to be had and then you are add that back to y. So, this is the reaction that is taken by the decoder. All that you are doing here, adding e hat pack, and this e hats only the function to with of the co set to which y belongs. And therefore, decoder action is only a function of the co set. With that in mind, let us take a look up at an example.

(Refer Slide Time: 18:23)



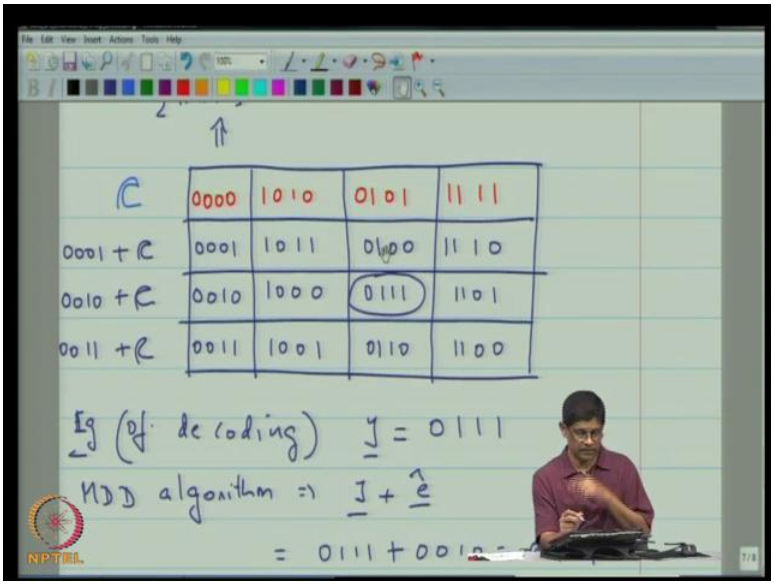
So in our example, let see we the following code. This is the code and in this case, the parameters are the obvious. The parameters n, k, d is given by 4; the dimensional is clearly 2, it is easiest to check the linear. Because you add any two code word in you get a codeword, and the minimum distance is the same as the hamming weight.

So, from that you can easily see that the minimum distance is actually 2. So, this is the 4, 2, 2 codes. Now let us also, since will need this little later, write down a generator matrix to this code. So, a suitable generator matrix would be remembered for to construct a generator matrix. What we need to do is, we need to set on a basis for the code, when you regard the code as a vector's space. So, it is easy to see that you can take any two non-zero vectors in this particular case. Now I am also interesting trying to keep the generation matrix and systematic forms. So that in mind chooses these two vectors of here. Some we can choose these two, because they will lead as will see to a systematic generated matrix.

So I am going to put down 1 0 1 0 0 1 0 1 and now, I have the generating matrix in systematic forms where I have the identity matrix some left, which is I 2, and have the matrix p on the right. And we already outline in algorithm to find the paretic check matrix given, the generated matrix in systematic forms, and actually in terms of that the paretic check matrix. So, in a case what you do is, you would take p transfers which should be 1 0 0 1 and then I… And so find that in this

particular case, reality, parity and generating (( )) are both actually the same. And that happens for a class of codes mono self dual codes, but does not need to bother us. Thus the co incident which is if know consequence here.

(Refer Slide Time: 22:00)



Now I am going to set up a table and that table these going to have four rows; one, two, three, four. So, that is one and I am going to have four columns and what I am going to do is, in the top row I am going to put down the code words. So, that is 0 0 0 0, 1 0 1 0, 0 1 0 1, 1 1 1 1, so these are the code words. So this then represents c the code itself. Now, I want to what we are looking up to accomplish is to implement minimum distance decoding in a very simple way.

This is going to help us do that and already point to that, the minimum distance decoder really cares only about the co set to which y belongs. So, with that reason what we are going to do is, we going to write down the co sets of the code. Now below this, I want to reserve now, the number fourth that is the sixteen. So, if you actually look at the set up of all of four topples from an abstract point of view, then there are sixteen of them and so they will be a code, and then the three of the co set of the code. You can partition this entire space into the code and co sets of c, the three other co sets of the c; three other co sets, c itself can be regarded of the co set.

There are other three co sets of c, so now you want to put down the co sets. But (( )) about that, I want to actually put them down in a certain way and the reason for that will become the parent in

a little (( )). Now to identify the new co set, a simply need to start by taking some element which is not in here. So, let me take I notice here let 0 0 0 1 does not belong to this. So, I am going to bring that in 0001 and I am now, I am going to form the co set zero 0001 plus c. And I am going to obtain the other element simply by adding this vector to the corresponding vectors. So, that will give you 1011, 0100, and 1110.

And I am going to continue in this fashion, I am going to look and I choose a vector small hamming weight here. That is the reason for that I am going to continue to do there. Now I notice that I have exacted zero, one hamming weight, one vector here and other one here. So, let me try in other one which has not been taking enough; 0010.

Then this becomes 1 0 0 0, 0 1 1 1, 1 1 0 1. Then this co set is therefore 0 0 1 0 plus c. The last co set will be are used upon the hamming weight one vector, I am looking now hamming weight two. That would like to start with this of vectors are flow hamming weight. So, I choose 0 0 1 1 plus c. I get 0 0 1 1, 1 0 0 1, 0 1 1 0, and 1 1 0 0. Again the table is set up in such a way that you add the header, the column header to the row header; these are the row headers. Now these vectors of least weight are actually called co set leaders. The vectors of least hamming weight in each column are called a co set leader. How would you use this for decoding? Selects try on example here, an example of decoding.

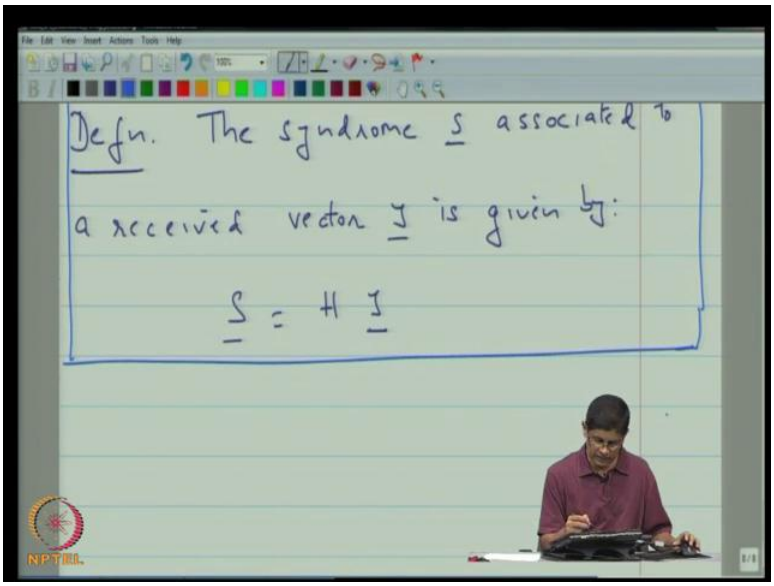
So, let say that why now that code words are these vectors are top right here. You have four code words one, two, three, four, so let say that for instance this code word was transmitting. Let say that the third symbol got corrupted so meaning that y what was it to see it actually 0111. An error corrupting in the third symbol so now how does decoding proceed? The decoded has a look, I am interested in finding my algorithm is at least our formalizations of the minimum distance decoding algorithm is to take y and then had e hat, where e hat is the least is the vector of least hamming weight within the co set which y belongs. So, I look up I look up for this, so I notice that here this is the co set to which y belongs, so y belongs over here.

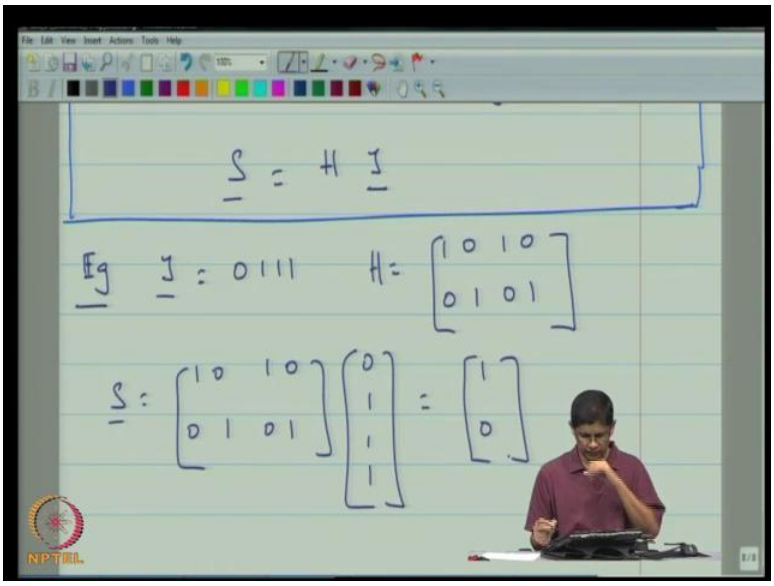
The co set leaders 0 0 1 0. so what I do? I take 0 1 1 1 plus 0 0 1 0 to recorded 0 1 0 1 and now correctly decoder. Now this table next decoding very easy to visualize, because all did you do is you pick your receive vector out of this table. And then the vector to which actually decode is the vector that is the top of the column. So for example, here was receiving that this is the co set

leader. You added and then actually you got that code word, which is the head of the column. That is having decoding proceeds here and that will always be the case. Now this is the further concept that way going to bring in. Now you notice that, this vector e hat here. That we had to receive vector was a function only of the co set which y belongs.

So, if I could somehow identify from y, the co set which belongs on I can directly figure out, what this e hat? Because in some sense, if u look up down this table, the set of possible e hats occur in this table so just a matter of determining. So, decoding or minimum distance decoding is just the matter of finding out, which co set y belongs to and adding the corresponding co set leader. Again just to emphasis the way this table is set up such that every entry here is the some of the row header and the column header. That is I have a set it up; that is important. Now we will make when observation which is important.

(Refer Slide Time: 30:56)

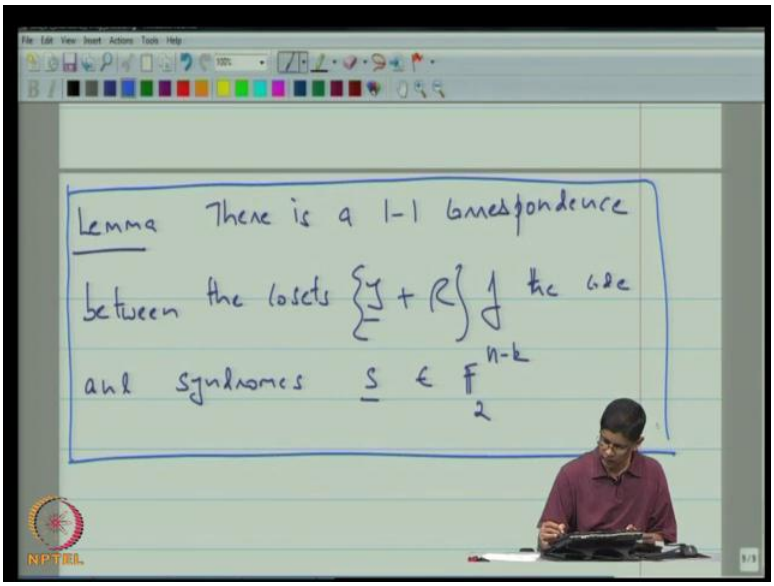


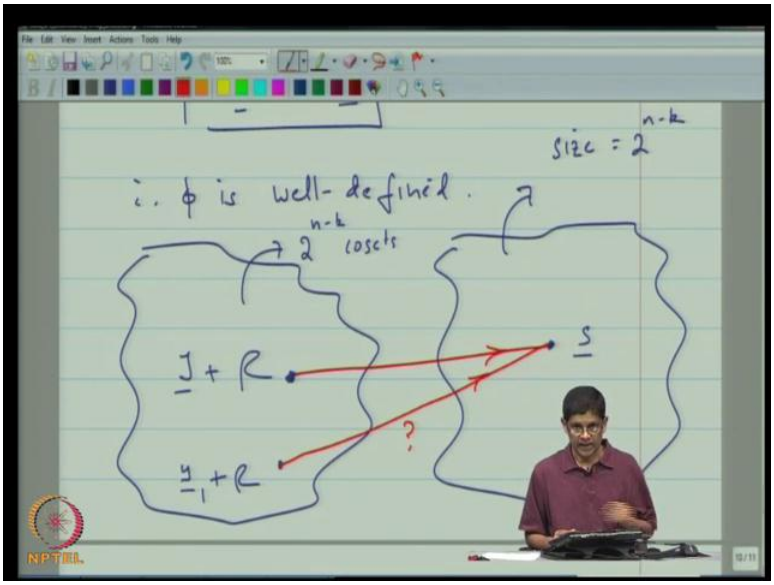


So definition, the syndrome s associated to a received vector y is given by, S is equal to H times y. Now remember that this H is the parity check matrix of the code. For example, if y… Now let us go back to what we had earlier here so we had y is equal to 0111 here. So, let us go down here and say that y is 0 1 1 1, then and we also know that the corresponding matrix H was given by 1010, 0101. So 1 0 1 0, 0 1 0 1 which is have to simply complete the sum. So, s is equal to… and you can easily see this is nothing but the sum of the last (( )) so that is 1 and 0. So, this is the syndrome associated to the particular receive vector.

Now the word syndrome is very similar in meaning to symptoms so syndrome is like symptoms. So, just like symptom tells you something about disease, syndromes tells you something about the error pattern. As it terms out that for the `linear codes, the syndromes tells you all that you need the actually find out the most likely error pattern, therefore the most likely code word.

(Refer Slide Time: 33:25)





So, the key to like lies in this lemma, there is there is a one to one correspondence between the co sets y plus c of the code and syndromes s which belongs to F 2 to the n minus k and all explain that. So back over here, we notice that in order to compute the syndrome you multiply H by y and we also know that this matrix H is an n minus k by n matrix. So, that is why this resultant vector will always have n minus k components and that is the reason for putting n minus k there. So proof, first of all what is the map?

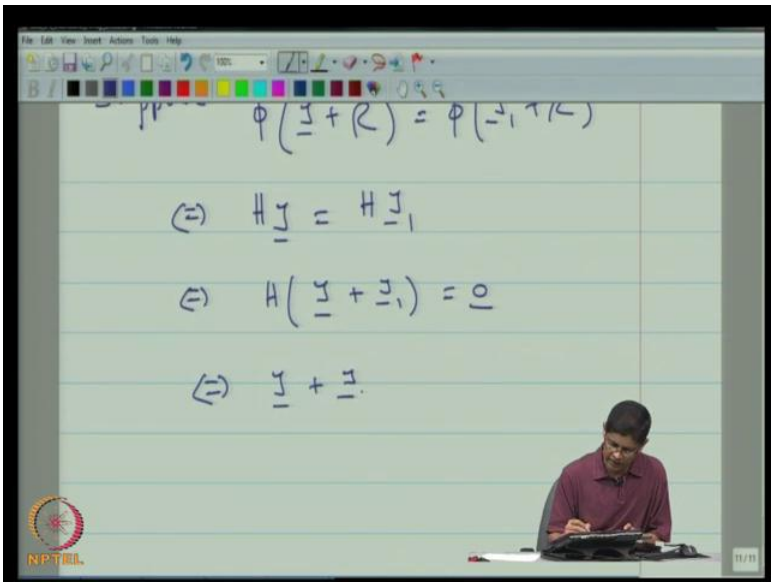
We introduce the map which takes the co set y plus c and maps it to H y, which is the syndrome. In mathematics, whenever you encounter situations like this, the first think you have to check is that the map is in fact well defined. So, what is the mean to say? It is well defined, because we need talk about the co set. You can you are operating on phi by multiply H by y. But this co sets could also be represented by different vector other than y. So, it make sure that in order which vector you pick from the co set, you will actually get the same syndrome. So, suppose the co set to ask here, is phi well defined?

So far that what you need to check is suppose y prime also belongs to y plus c. The same co set which means that y prime is equal to y plus c, where c is some code word and script c. Now you want to make sure that each times y or each times y prime does not make any difference. That is the same thing, therefore the H times y prime is H times y plus H times c. But H time c is 0 for every code word. So, therefore the same … and this is what we want to show.

Therefore it is well defined. Now we want to actually establish more, we want to establish that in fact that they one to one corresponding. That is the pictures something like this that on this side, we have co sets y plus c. And on this side, you have syndromes. What we doing are your mapping each co set for particular syndrome.

And you want to make sure that the map is one to one and one two. Now quick check will tell you the number of possible syndromes is two to the n minus k. So, this set is of size equal to 2 to the n minus k. Since the set of all in topples is partitioned in to co sets by code on the code has sized to the k. There are 2 to the n minus k co sets as well so you are actually looking at a mapping that is taking place between two sets, and the size is the same. So, only you shows now are this one to one? Or it possible that for some other y 1 plus c, which the different co set, is? Is it possible that from other co set will also be not on to the same syndrome, can this happens? So, let us go to the question mark. As it terms out, it cannot.

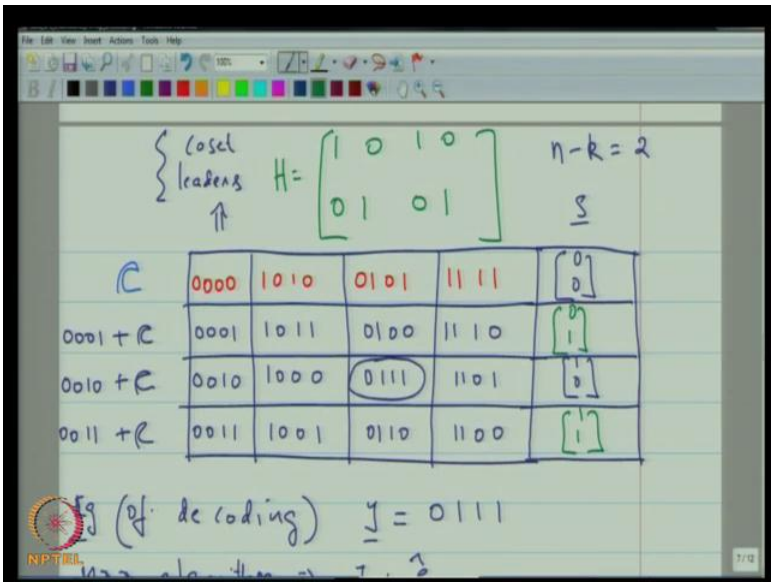
(Refer Slide Time: 38:58)



Suppose phi of y plus c is equal to c is equal to phi of y 1 plus c. That is equivalent to sign that H of y is equal to H times y 1, which is equivalent to sign that H of y plus y 1 is 0. But that is equivalent to sign that y plus y 1 belongs to the code, which is equivalent to sign y 1 belongs to y plus c or, I mean take this little bit slowly perhaps this that I am clear. This is equivalent to sign that y plus y 1 is equal to c, where c belongs to the code which is equivalent to sign that y 1 is y plus c in c. But that again, y 1 is y plus c, then that can only happen if y 1 and y are in the same co set. Implies that y 1, that y one and y define the same co set and we are done.

Because what we shown is that the only way y plus c and y one plus c can map on to the same syndrome is in fact defining the same co set? It is not possible to have a picture like this. Because the problem in the picture here is that here you regard your actually looking at two different co sets. But we are saying that the only way two co sets can map on to the same syndromes is in fact they are the same so we cannot be (( )). So, they are the one to one correspondence and the correspondences are given by this. So, let us go ahead, see this in example.

(Refer Slide Time: 41:26)



So, what will do is for this particular code, we will go ahead and extended will add an additional column here. And will reserve this for this syndrome which means, what will actually compute H times y for any y in this co set. It does not make it difference.

So here for example, H times zero is 0 0. The number, this code has parameters n, k, d is 4, 2, 2 so that n minus k in this case is equal to 2 so the syndromes are two topples. So, now I want to compute the syndrome here is 0 0. Here we already computed earlier and it was 1 0. We computed the syndrome here found that is 1 0 and in fact, so path set (( )) Although it is going to crowd this screen a little bit, if I put down the matrix H forward here. But we use a different color, which is lightly darker.

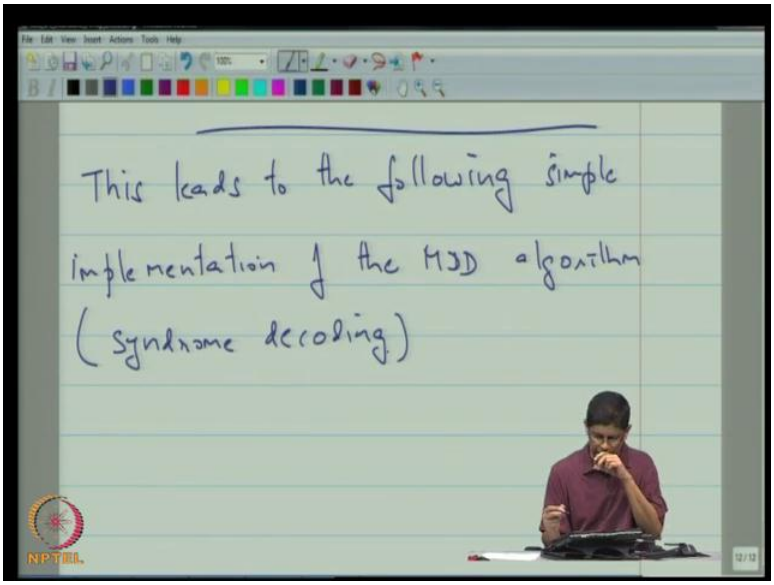
So, if I want to compute the syndrome of this I am going to multiply this vector by this, and going to multiply this matrix into this vector which means I pick up the last column. Therefore, this is 0 1 and similarly here you can actually see it is 1 1. So, you see that are four possible syndromes and there are four co sets and each co set is associated to unique syndrome.

This is the map phi so how it is that help in decoding? Now it is very simple, because if you have this table, then the right most column of the table gives you syndrome for each of the co sets. So, what you do is you simply give in a receive vector, your decoder does the following. First of all, I compute the syndromes by just computing H times y. Then from the syndromes dues table we

up to determine the co set leader, which is called e hat. Then it adds e hat to y to recover the estimated code word and in fact, in this particular example here we could have seen the decoding. We could have seen the decoding has being carried out like this so here is 0 1 1 1.

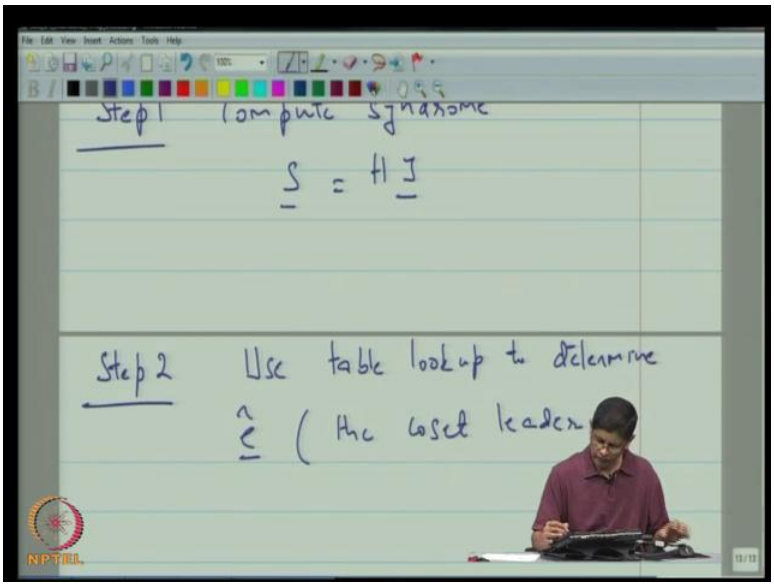
So in this, I am going to view this syndrome decoding which means that given 0 1 1 1 or first as to compute the syndrome which is 1 0. Then we use the syndrome to actually look up the corresponding co set leader which is 0 0 1 0, then we add that back to the received vector to get the decoder code word. So well did you see an example of this, but just so that we are clear.

(Refer Slide Time: 45:21)



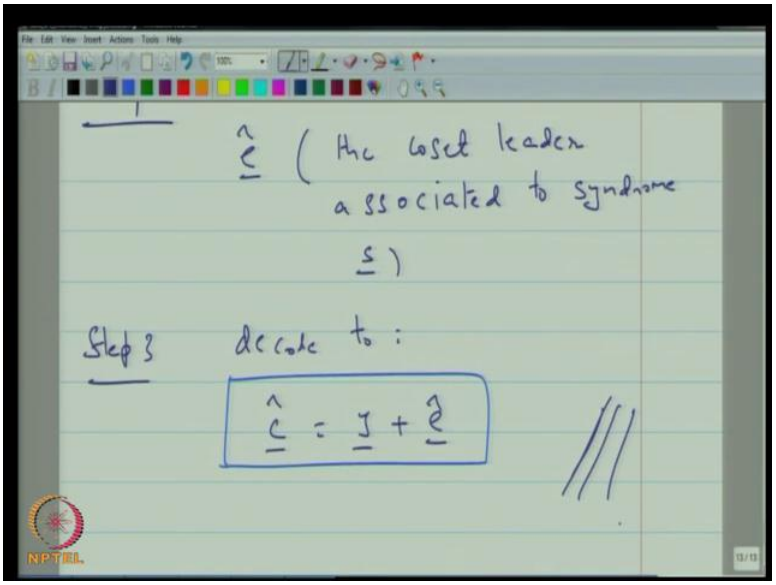
Let me write on the steps. This leads to the following simple implementation of the minimum distance decoding algorithm and if you like, you can call this syndrome decoding if you like or standard array decoding.

(Refer Slide Time: 46:20)



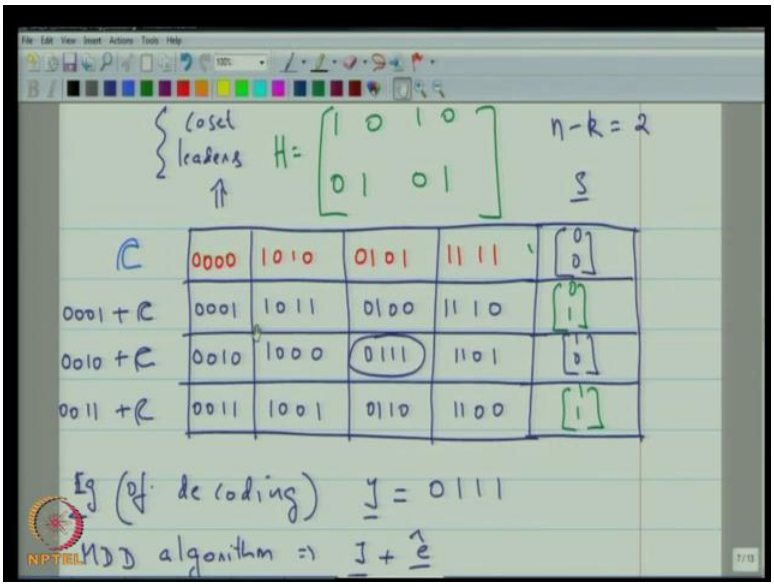
So step 1, compute syndrome s which is given by H times y. Step 2, use table look, use table look up to determine e hat which is the co set leader; the co set leader associated to syndrome s.

(Refer Slide Time: 47:27)



Step 3, decode to the vector c hat which is given by y plus e hat and you have done. So, in actuality, what that means is so let us go back a couple of frames.

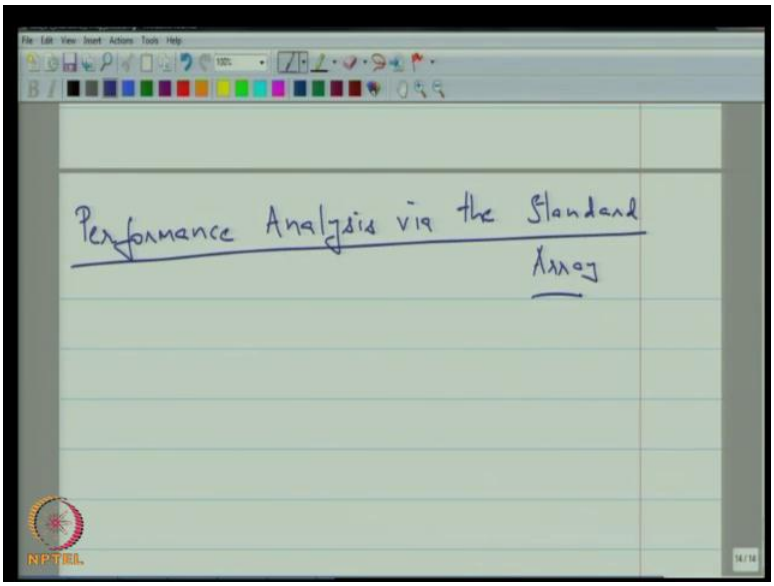
(Refer Slide Time: 48:05)



What that means in actuality is that to actually carry out syndrome decoding, you only need, you do not need this entire standard array. You actually need only the co set leader column and the syndrome column.

Everything else you can actually do away with, because you can compute the syndrome and then jump the table look up dually from the syndrome to the corresponding co set leader. I let to the receive vector and you get it to the decoder code word. So, rest of those, it middle three columns in this particular case when not really required. However there is in other use for this standard array and will come to that so that is next topic.

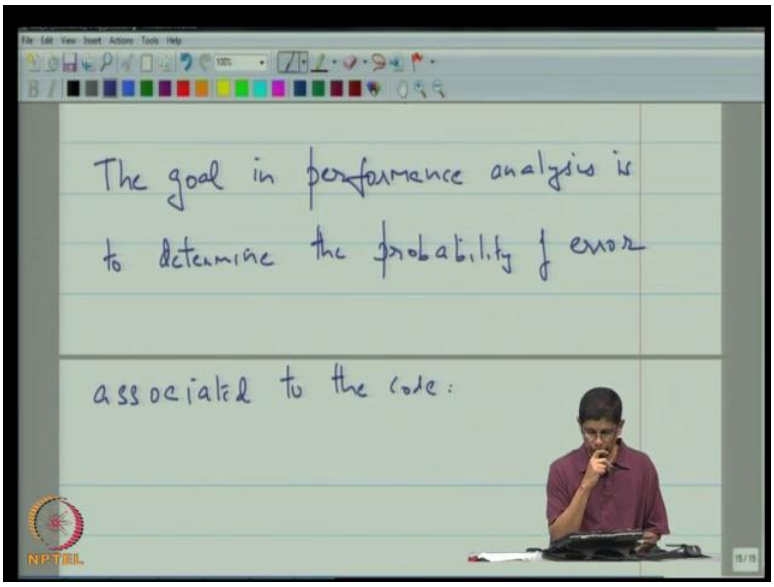
(Refer Slide Time: 49:18)



So, the next topic is performance analysis via the standard array. In this, so for this what I am going to do is I am going back to this standard array and I am going to reproduced it. But we going too viewed lightly differently so first let me selected, now want to copy it and I want to set it down. Here we have are standard array and let me get read of some unwanted entries here. So, these are syndrome. And let us also get read of this line here so this is standard array again. However this time, we going to interpret entries the little bit differently. Now are earlier interpretation was that we view this is the table in which the receive vector is one of this entries.

And then to actually decode, what we do is we look up at the row to which they receive vector belongs. And then we look up at that first entry in that row and that you estimated error pattern, and we are in the back to the row. That was how you viewed it. But now you are interested in performance analysis. So, what is performance analysis mean, performance analysis means that we trying to determine what the probability of error of which is involved decoding in this code? Now this decoding algorithm is the best in terms of minimize the probability of error. So, whatever the probability of error is it is a consequence of the nature of the code itself. We are trying to say is this where is the probability of the error of this code when it is decoded to minimize the code probability of code word error. Let us put that down.

(Refer Slide Time: 52:18)



The goal in performance analysis is to determine the probability of error associated to the code. Now this naturally sound little bit wake to you and actually deliberately left this wake. Because you could say what exactly do you mean the probability of error? Do you mean probability of code word error? Do you mean probability of message bit error? And there are so many different message bits, the probability of error which particular message bit. So, the answer is there actually this standard array can we used to determine which ever one of this quantity, all of these quantities that you like. And that is going to be the next in the next lecture way going to take this are.

So, given that we have just about two minutes left, let me just summarize. What we looked at in this lecture is a convenient implementation of the minimum distance decoder and it is call this standard array decoder, all and it needs to something which of called syndrome decoding.

The basic idea is that the minimum distance decoder only cares about the co set to which the received vector belongs. So, what you do is soon as the received vector comes in, you determinate co set by computing the syndrome. And ones you are the co set, you jump to the co set leader by using table look up. And once you have this co set leader, you add the back to the receive vector and you get most likely code word. So, you are minimizing there by the

probability of code word error. Now, what was I going to say? I was going to say that for modulate codling sizes; this is in fact very practical algorithm.

In fact, I myself have involved in a project where this was carried out, when we consult it for resultant company and so for reasonable blocks, are you can actually do this. Therefore, implement maximum likely decoding. It is only when the code blocking become very, very large, then this is the no longer practical. So, I think that this is more or less good place to stop and the next lecture will take about performance analysis of linear block code using this standard array. So, the array remains same, expect the perspective will change. Thank you.