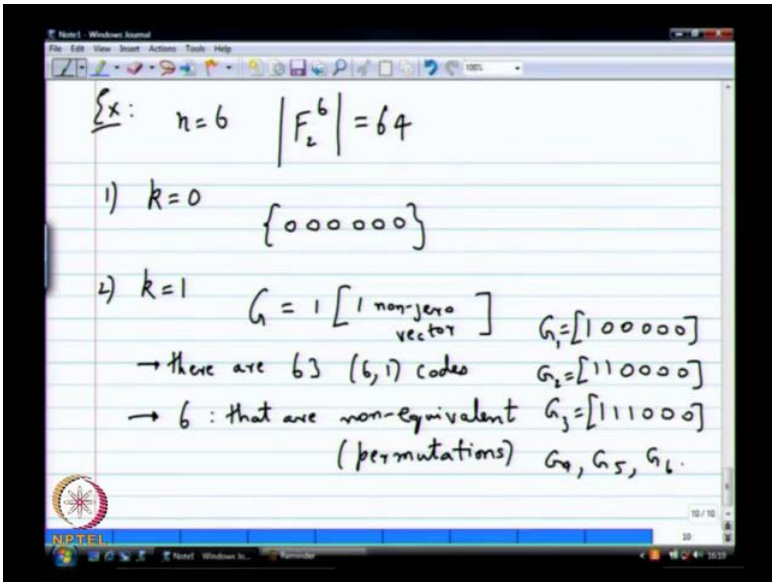
Coding Theory
Prof. Dr. Andrew Thangaraj
Department of Electronics and Communication Engineering
Indian Institute of Technology, Madras

Lecture - 4
Minimum Distance of Codes

So, we are going to continue just a quick reminder about what these codes are you know I mean I have been emphasizing the vector space view the linear space view. So, you might want to think of think of codes of the spaces that is the good thing to view a book that is a good idea to keep in your head, but you should also remember that these are discrete objects I mean this only a finite number.

So, often times when you think of planes you think of plane in 3 D, there are infinite number of planes. So, many of them and same thing also spaces in order, but these are just finite objects there is also more component oriel or discrete way of looking at it, which is also equally important. So, just emphasize that I am going to ask few questions very simple questions as to make sure you also view them as discrete objects and not just as a spaces.

(Refer Slide Time: 01:06)



So, here is the example, so I am going to put this in a example but this will be a bunch of questions about what these codes are this is a first exercise n equals 6. Remember, F 2 6,

so it is a vector space all right it is a six dimension vector, but how many how many vectors does it have exactly 64, so that itself is finite.

So, if you think of codes inside this finite vector space you only finite number of thermal only a finite number of there will be only finite surfaces also we will see how it works. So, suppose I ask a question suppose a 6 k equals 0 and then I ask you how many 6, 0 codes are how would answer that. So, it is very easy answer, there is only one code with k equals 0 what is that one code with k equals 0. There is only one code and that is some this, so this is only a single code with k equals 0, so that seems like a simple enough answer to give, but what about k equals 1 k equals 0 also not a bad idea, just to emphasis that it is a pure component oriel 0.

Definitely, it is a fine 6, 0 code there is an object it has only 0 definitely not a very nice it has no significant what about k equals 1 there are 2 code words in each code, but my question is little bit different I am asking how many 6, 1 codes are there. That is the correct answer a 63 is the correct answer there are 63 different codes with k equals 1 how they how what why does that answer make sense. You have thought of generator matrix always go to the generator matrix how many different generator matrices can I have for a k equals one code.

So, what is this generator matrix, now it is simply k equals 1 and n equals 6 you just have to pick one vector 1 on 0 vector how many non zero vectors are there in F 2, 6, 63 for its a choice you will get a well a technically a different 6, 1 code. So, is there are 63, 26, 1 codes, now suppose if I say I want only codes that are not equivalent under permutation not equivalent under permutation is what exactly I want. So, how many different k equals one codes will they have which are distinct, even I include permutation, remember when I have a G, I can do column swaps and really change the code.

Suppose, I say I do not want that to happen what should happen you cannot have the same number of 1 any 2 non zero vectors, they pick here with the same number of 1 will be will be equivalent under the permutation I can. Let me do the permutation get it, so I have to have different number of 1's how many different vectors are there with different number of 1's amongst them. If we have 6, exactly 6 you can have 1, 2 or 3 or 4 or 5 or 6, only 6 of them are distinct from a from a permutation equivalent point.

So, that are nonequivalent, so of course under permutations, this I think I mean the way the reason why I am doing this simple questions is to give you an idea of how you tell me it is 6 or 5. So, what is this 6, first of all not 6, 6 is not the number of benches number of different codes there are 6 different codes. So, what are the 6 different codes, there will be 6 different generator matrices, the first generator matrix will be what 1 followed by five 0's the second one will be 1 followed by five 0's.

The second one will be 1 1 followed by four 0's, you can also take a variety of different things here, how many different can I take here, I mean instead of G 1 how many others are possible 6 are the G 1's are possible. How many other G 2's are possible 6 is to that would be 15, 15 different are possible all of a G 3 1 1 1 0 0 0 and then G 4 likewise straight.

So, you have a G 4 G 5 and a G 6, so these are the six codes which are not equivalent under permutation. So, these are different questions that you can ask in a binary vector space which does not may be does not have an equivalent in the real numbers. There is no reason why you should ask such questions in the real numbers, but in the binary vector space there are nice questions like this that you can ask these are essentially common oral.

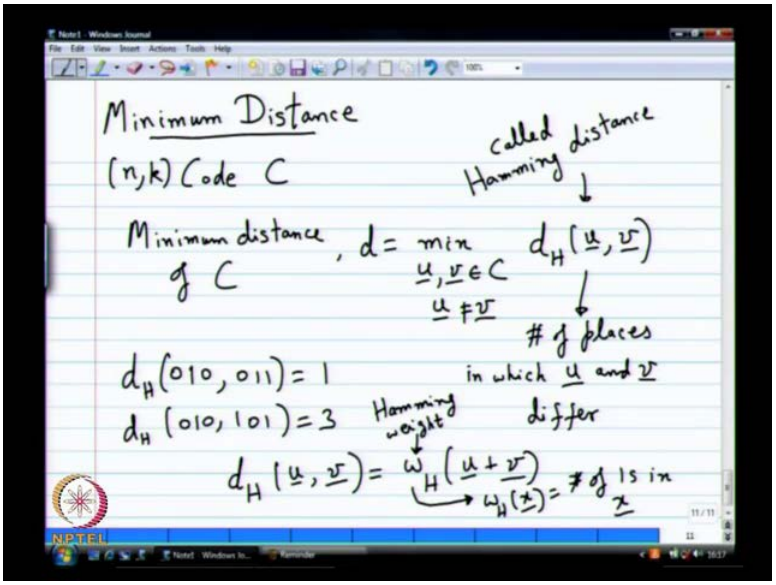
So, a question like this you can imagine becomes much more complicated if I say k equals 2, now k equals 1 becomes easy enough everything is simply if I say k equals 2 what will happen. So, you have to pick two linearly independent non zero vectors, so the first answer is actually not very difficult. So, in fact there is a formula that is well known for n k, so if you have k equals 2 what will be the formula the first vector you can pick as any non zero vector. The next one should be linearly independent of that what the only linearly independent dependent vector is.

So, you pick any one from the 62, so 63 into 62 is that enough or am I making some mistake. Here, should I divide by 2 to reduce the row choice, you have to divide by 2 to get a better number, so that is a formula like that in general. So, it even eventually becomes more and more complicated if I ask a question about nonequivalent codes for 6, 2, it will become little bit more complicated.

So, they those are combination oriented questions about these codes which are more difficult to answer I am briefly mentioning it. We will never get back to this in this

codes, we just writing it down so that you are aware that there are other ways of viewing codes. Then just as sub phases they are also equally interesting they have very challenging problems in that.

(Refer Slide Time: 08:36)



So, let us move on the next important and probably the most important attribute of code is what is called minimum distance. Of course, n and k are important, nobody can say n and k are not important, but n and k are in a way easily compute over n. Definitely is very easily comfortable k is also easily comfortable collision elimination. It is not a very complicated operation of utmost it can be n part of complexity, it is not very difficult, and so you can compute it.

So, minimum distance is a property of the code which is not easily comfortable, you can show it is known as empty complete extra. So, it is a very difficult thing to compute it, but never the less its one of the most important characteristics of the code you can imagine. The most important characteristics are not easy comfortable, so that is what all problems in engineering are. So, you have to get used to it, but let us see what the definition of this is suppose I have a code C, so let us say we take n k codes c does not matter n and k are not so important for the definition of the minimum distance.

Usually, denoted d is let me write it down differently, I also introduce a motion of a hamming distance, soon I mean as in this definition it will come. Let us do this minimum over u comma v in c u not equal to v was known as the hamming distance between u and

v what is d H of u comma v. So, d H u, v is number of places in which in which u and v differ. So, this is the definition, so this is the minimum distance, so just looking at it seems like why I said it is hard to compute.

So, what I am saying is after all I am doing a very simple competition why is it hard to compute which part of it is hard and which part of it would be easy given u and v finding d H of u comma v is not that hard. It is a complexity of order n, so you have to do n different operations, you can compute it, but what is the difficult part over all possible u and v. So, how many possible u and v do you have, u and v are in c, so let us be very careful of it, how difficult this problem is.

So, 2 per k, choose 2 that is the possible which is going to be exponential in k exponential in k. So, it is not going to be problem on it and you can do it very easily, so that is the problem essentially. So, u and v in c are going to be 2 power k, choose to which is a finally, large number, so the number of different comparisons you have to do is very large.

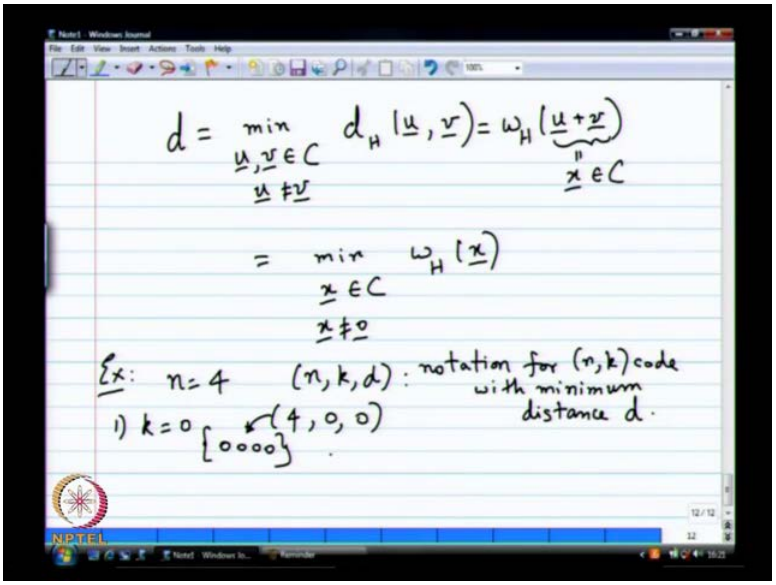
So, this it turns how the 2 power k choose to can be reduced to 2 power k itself I will show you how the distance it is a very simple, I am sorry. So, I am going to derive that next it is it is a very simple step, we will do that first and then we will proceed further. So, first of all this distance if you have not seen it before can be little bit confusing, so the hamming distance is called hamming distance after hamming which can be considered as a like the father of code, it is a very simple.

So, let us see if you quick examples of this and then we will proceed further, so suppose I said the H of say 0 1 0 0 1 1, it should be 1 1, so and then the H of 0 1 0 1 0 1 should be 1 3. So, these are kind of examples, so one relationship which is very useful for having distance of the following the H of u comma v is the same as what is called the hamming weight of u plus v. So, what is w H, now to define what w H is this is called the hamming weight, the hamming weight is slightly simpler.

Then, simpler then the distance suppose you have w H of x this equals number of 1’s and x, so this is the weight of a vector it is called hamming weight often times hamming is dropped the same thing. Hamming distance also distance and weight are basically hamming distance and hamming weight is the number of 1’s in a vector and this relationship is very easy to prove number of places in which you and we differ.

If you do an x sort of u comma v what will happen where ever there is a 1 u and v differed in that place if it is 0. Then they will differ you simply count the number of 1's in the x of which is the weight you get the answer. Now, this relationship can be used to decrease the competition needed in the minimum distance competition a little bit not too much you go from 2 power k. Choose 2 to simply 2 power k, remember two power k, choose 2 is what I think, 2 power 2 k, this is slightly small smaller. So, it is not a it is not a to be a statement it is a reduction it is not a very significant reduction, so like is said this problem is very hard.

(Refer Slide Time: 15:07)



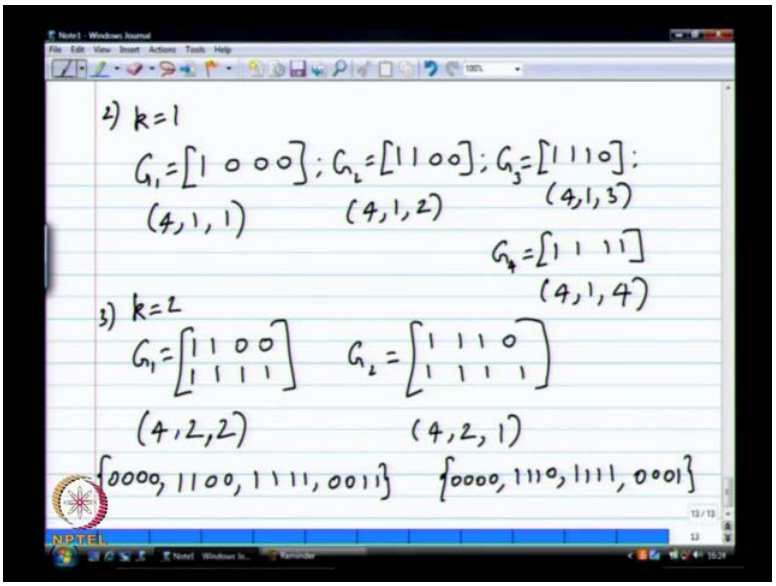
So, you cannot really make a so let us see how to do that it is very easy what is the definition of minimum distance minimum over u comma v in c u not equal to v, the hamming distance between u comma v. Just now, we saw the hamming distance between u comma v is the same as the hamming weight of u plus v, but remember u is in the code v is in the code. So, what about v plus v it is also in the code, so this is this is equal to some x which is in the code.

So, instead of going through all the u, v's and then adding them up and ending back in the code itself, I can just look at a code words of the code and simply do a weight. So, this becomes the same as minimum over x in c the only thing I have to worry about is u not equal to e v, u not equal to v translates as x not equal to 0 of weight of x. Now, the

minimum is over only two power k minus one possibility, but still its exponential in k lot of competition has to be done.

It is not simple little observation that u plus v anyway belongs to the code, so you will be anyway doing the same competition over and over again, unnecessarily you can avoid this. So, let us see some examples quickly compute with a minimum distance, so I will restrict myself let say n equals 4, this is simple cases let us start k equals 0, k equals 0 is nothing much to do here. So, it is 4, 0, so I should tell you what I am writing here, so codes that usually denoted n k d, so will denote codes as n k d. So, notation for what n k code with minimum distance d so that is the notation, so for k equals 0 we saw just one code which was just 0 0 0 0 and that is a 4 0 0 code, so it is of not very interesting.

(Refer Slide Time: 18:00)



So, what about k equals 1, so we have several different possibilities there will be four possibilities which are nonequivalent. I will list them out, so let us say G 1 equals 1 0 0 0 G 2 equals 1 1 0 0, G 3 equals 1 1 1 0 and G 4 equals 1 1 1 1. So, those are the four non trivial non equivalent possibilities k equals 1 and what would be the parameters for these codes 4, 1, 1 how did I get that one? Remember each of these codes have just two code words one of them is all 0, I do not have to consider that in a minimum distance competition the other code is what is in that row itself.

So, clearly it is 1, so what about this 1 4 1 2 4 1 3 and this k is 4 1 4, so among these four which is the best code, I mean I never told that having a large minimum distance is good,

but it is like you assumed it and seems like good as this one. So, may be for 1, 4 is best code of this, so let us take a few with k equals 2, I do not want to take too many of these codes, let us take let us say G 1 equals 1 1 0 0 1 1 1 1. Simply take two just for comparison 1 1 1 0, so there are 2 codes k equals 2 there are of there are of these several others I am not writing down for about the parameters is here 4, 2 is easy.

What about the third parameter 2, so how did I get that there are four code words in this code four code words are 0 0 0 0 1 1 0 0 1 1 1 1 what is the fourth code word 0 0 1 1 clearly by looking at it you know that is the minimum distance what about G 2 4 2 1. The code if you find out is 0 0 0 0 1 1 1 0 1 1 1 1, the last code word is 0 0 0 1, now I am going through out open challenge in the class we are recording, but any way this is interesting challenge. So, what is the best possible minimum distance that you can come up with for k equals 2 any better, so for k equals 1, we are able to quickly see that there was one generator metric which gave you the absolute best code.

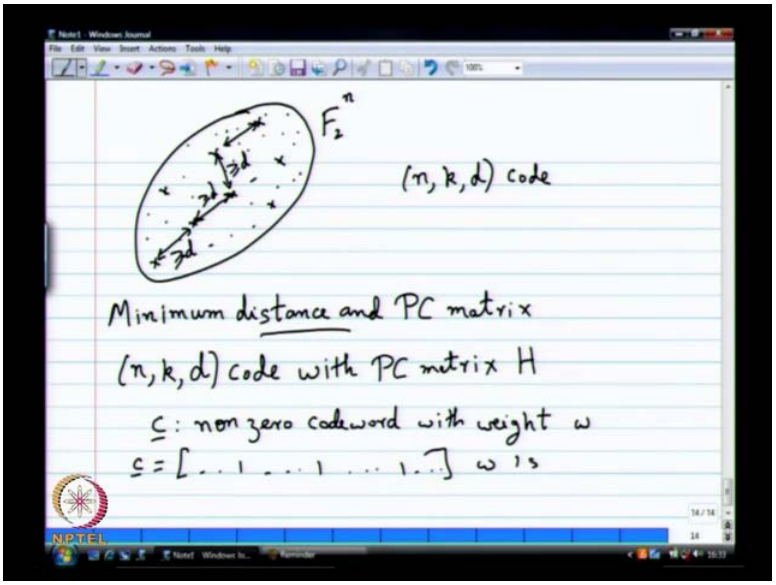
Nothing else you can do what about 4 2 can you quickly prove in your head eliminate all the cases, so that 3 is not possible it is not a difficult proof for 4 2 you can never have a 4 2 code which has minimum distance 3. So, can you can look at it for 4 2 3 will not be possible, it needs some careful arguing, it is not too difficult in this exhaust all the possibilities we have to have two G 2 rows in G.

Each row should have weight at least 3, otherwise it do not have height, so both have 3 right and then we exhaust them what has to happen, two of them will go of 0. They have to overlap at least in two places, so that is a pure component oriel argument, so this is where a component ores and coding will come in. So, when you do minimum distance arguments, it will become little bit more component orient, so it is more painful writing down sub spaces and all is very easy when you think in terms of this even for 4, 2 is so difficult.

You can imagine for a general n k, how hard it would be to think of minimum distance, so it is bit more complicated, we will see more about these things later on. Anyway, hopefully we have convenience that answering the question of what is the best minimum distance for an n and k is not very easy. So, it is a bit more difficult you have to do some calculations think about it. So, it is a bit non obvious I can ask you more difficult questions 4, 2 is easy enough.

If you go to like 1,000 and 500 n equals 1,000 and k equals 500, what is the best possible minimum distance, it is very hard question to answer, In fact many of those questions are not answered till now if you find an answer you can become very famous, so think about it. So, this is the difficulties already you have seen minimum distance is a little bit more painful and what we thought, so we will see more and more examples it will it will become clear as we go along any questions at this point. So, that is all the examples I wanted to do with minimum distance, but I will draw a picture of code and minimum distance etc just to give you a feel for how it looks.

(Refer Slide Time: 24:15)



Suppose, you imagine that this big ellipse here holds all the this is nice geometric view which is very useful so holds all the vector in F 2 n. Suppose, this is the set of all vectors in F 2 n and suppose you are looking at an n k d code, how many code words will there, be 2 power k code words. So, maybe I put a star at each code word, so there will be 2 power k code words, maybe I should put one more star, make it this is valid 2 power k. So, you have these code words how do you capture a d you can think of distances having distance, so little bit not same as you see in the distance. Anyway, you can imagine that that this hamming distance what am I saying, when I say minimum distance is d the distance between any two stars is at least d.

So, that is the picture that you should have in mind, so you have this set of all vectors of few of the vectors of a code word and any two code words are at least a distance be apart

in hamming distance. So, once again let me summarize n k are easy for most codes, you describe them n n k will be easy or in facts for some codes descriptions k might be difficult. Anyway, does not matter, so n k are usually easy given n k if you ask question like what is the best possibilities that they can get those questions are tough.

It is naturally going to be because they are not algebra, they are not linear algebra questions they are component oriental questions involving discrete objects, and these are naturally more difficult, in fact many of these problems are can be complete. So, you can for hope for a simple answer if you imagine that difficult answers, so the next never the less what I should tell you is there are constructions available which give a d and an n try to construct a generator matrix or a paretic matrix for n k d code. So, there are constructions available for a given d, so it seems like quite challenging job it is a very challenging situation.

One relationship which useful in controlling minimum distance is the following property of the quality check matrix, it turns out the paretic check matrix in the minimum distance are related. So, it does not seem very obvious now, but when I write it down, it will be clearly to you and that relationship is very vital in the design of codes with a sudden minimum distance. If I want to design a code with a certain guaranteed minimum distance this relationship that I am going to write down between the minimum distance and the paretic check matrix is vital.

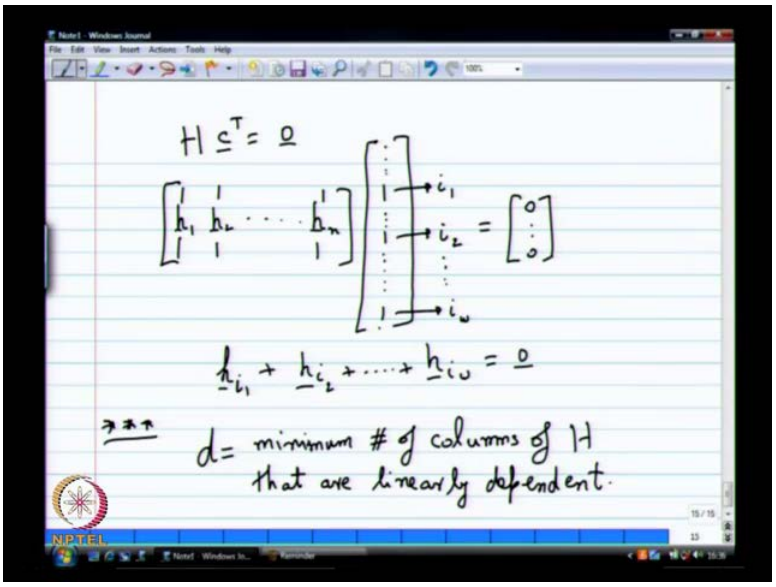
So, that is relationship you use again and again to relate minimum to design codes with a guaranteed minimum distance, so what is the relationship, it goes like this. So, let me write it down minimum distance and paretic check matrix it is a very simple relationship, but it takes some time to get your head around.

So, let me write it down first and we will do a few examples, I will ask you a few questions and eventually it will, so suppose I have an n k d codes remember I am writing n k d. Now, previously we were writing n, k code, now I am adding the d with paretic check matrix suppose you have an n, k, d code paretic check matrix, what is the connection between d and H should what I should ask.

So, what I am trying to prove, so let us see what is H if you have c being a non zero code word with weight, let us say we see this some non zero code word with weight w. What

does it mean, see if you write it down as a vector it will be one bunch of 0's, but how many 1's will there be w 1's, so everything else will be 0.

(Refer Slide Time: 29:24)



What do I know about the paretic check matrix and c, H time's c transpose is 0, now let me write down the column of H the first I am talking about the columns. So far I have never spoken what the columns, always it was always about the rows we were worried about the rows space of G which was recorded self and the rows space of H which was the dual code, so that seems simple enough just linear algebra. I am going to talk about the columns of H let us say the column first column of H is H 1, second column is H 2 so on, there will be n columns and I will assume that in c you have you know w 1's and these are in the i 1th position i 2th positions so on till i wth position.

So, what happens actually when i do this multiplication actually what happens is if i think these things as vectors H i 1 plus H i 2 plus so on till H i w equals 0. So, what is this means let me state what we did just now if there is a code word of weight w what does it mean there are w columns of H which had 2. Another way of phrasing it is there are w columns of H which are linearly dependent, so that is the way to think about it alright so the weight of code word is this implies that that many columns of the paretic check matrix are linearly dependent.

Now, how will I characterize the minimum weight or the minimum distance of the code in any dependent, so let me complete that the minimum distance of a code is the

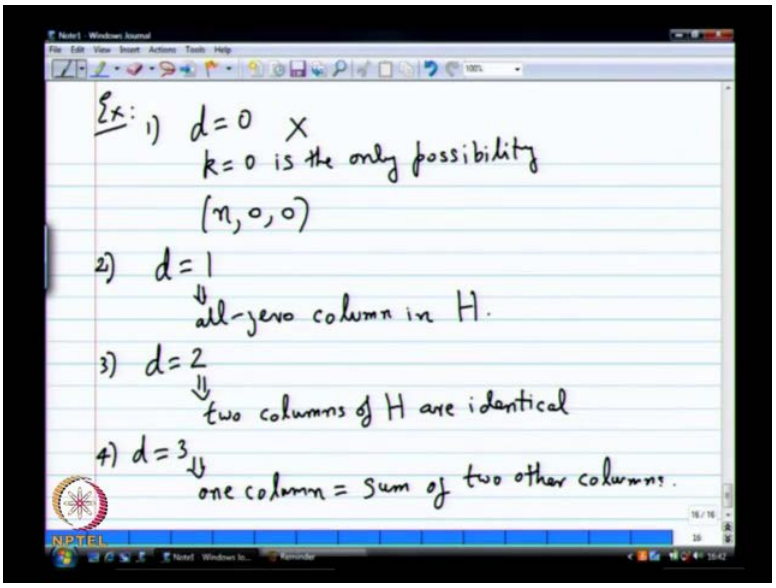
minimum number of columns of which are lineally dependent. So, that is from this, we get this following relationship which is an important relationship, so let me put few stars to denote that it is important d equals minimum number of columns H at that lineally dependent, so I gave u an argument.

There is certain converse which I did not really prove which is important, so the converse is equally important what is a converse if there are w columns of H which are linearly dependent then there is code word of weight less than or equal to w. So, that also meting you can quickly show it is not so difficult, so once you show that this equation becomes a. So, d equals minimum number of columns of which that are linearly dependent, we come to d and all that, but one thing I want to quickly point out is d is very different from rank it is defined using dependence of columns, but you know dependence of columns is roughly the same as dependence of dependence of rows.

It is not a big deal that is the same, but it is totally different from rank, it has no direct relationship with rank convenience yourself that it has no relationship. As we see more and more examples you will be sure there are some bounds using rank, but there is no equality relationship with rank. You can just compute the rank and get it if you can compute the rank get it then it wouldn't be a unsolved problem.

So, it is a very difficult thing to answer what is the rank how you would define rank maximum number of columns that are linearly independent. So, that is rank comes, so it is a bit different, so you here it is a minimum number of columns that lineally dependent you can have a edge which has which has an all 0 column. This means the minimum distance is one there is one column which is linearly dependent, but still the rank can be full, so there is no direct relationship, so let us let me give you some examples, so we will see how it goes so the examples are very illuminating.

(Refer Slide Time: 35:08)



So, let us see the first thing I am going to talk about is d is equals to 0 when will d be 0 go back to this definition. Suppose, d is 0 if no real relationship between distance and rank, so how will you have minimum distance 0 we cannot have minimum distance 0 non zero code word will have weight at least 1, so it will not have.

There is no sense in saying d equals 0 because by definition k 0 is the only possibility is the only possibility. So, for any non zero case it will happen, so let us not worry about this too much, but any way you can keep this in the back of the mind. So, basically n, 0, 0 code is the only possibility for d equals 0, you can take any number of examples.

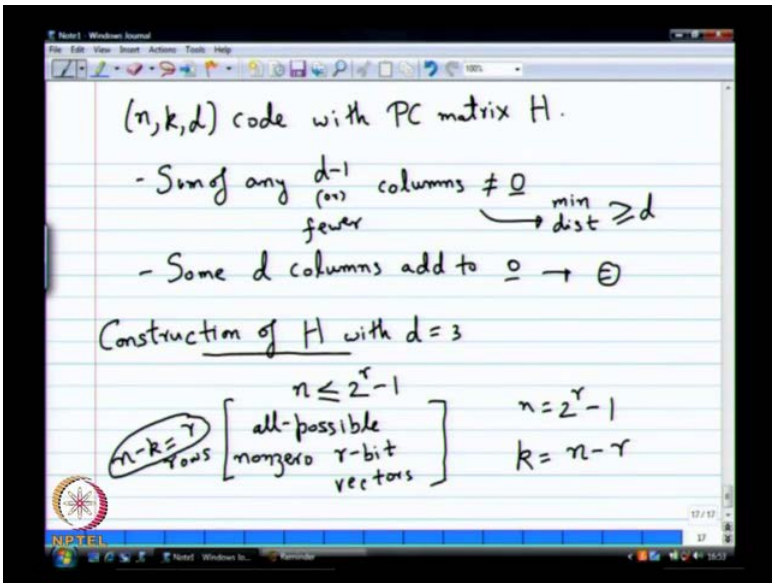
So, the next interesting example is d equals 1 from code of view I guess answering the question is very simple you just have a weight one code word there is nothing much to say. From the paretic check point of view what should happen if you have d equals 1 i will two columns d equals 1 means what one column is lineally dependent. So, that means what should happen you should have an all 0 column there should be an all 0 column in H, so is that is the only way in which you can have it.

So, the third question d equals 2 there should be some reputation in the column 1 column 2 column should be identical what about b equals 3, so that is the way it works. So, d equals 2 already I get this two columns of H should be identical, it should that are linearly dependent only way that will happen is the two columns add to 0. This means they are the same exact same statement to made for d equals 3 what should happen one

column equals sum of two other columns well when I say sum, it always modular to sum.

So, d equals 4 onwards is a little bit more complicated so d equals 4 may be is not too difficult, but d equals 5 onwards is definitely more complicated. So, many other is not too difficult to write down, but you see it is not very easy to just see and understand only up to d equals three you can just look at the paretic check matrix and quickly figure out something beyond that it is more difficult. So, you have to exhaust all possible cases, so next thing I am going to ask, I am going to make another statement you tell me it is true or not.

(Refer Slide Time: 39:27)



Suppose, I have an n k d code here it is another result which is interesting n k d code with paretic check matrix H and if I take, so what will happen to this some of any d minus one or fewer columns will be what it will be non zero. That is the way to that is another way of restarting the same if you have d minus 1 or fewer columns, it will be non zero is that enough what does this mean if I have sum of n a d minus 1 of columns not equal to 0, it means minimum distance is greater or equal to d.

So, what I should I have a n k d code, there is a set of d columns which add to 0, some set some d columns add to 0, both these conditions have to been satisfied if you want to claim that the minimum distance. If you know that the minimum distance is d, both these things should be satisfied it is not enough to just say d minus 1 of few columns do not

add to 0. This only gives a bound this tells you that minimum distance is greater or equal to d this only tells you that it is make the thing equal.

So, quite often as strategy to show that I have a paretic check matrix which defines a code with minimum distance at least d is what could be a strategy based on what I derived. Now, you show that any d minus one of your columns do not add to 0, so in that case the minimum distance will be at least d or if you want to put it in more general terms you have to say any d minus 1 of your columns are linearly independent. So, now we are dealing with binary vector space, so linear dependent is the same as sum if your binary becomes some other feel.

Then, you have to worry about co efficient also, in that case you make it lineally independent, so this is the strategy that is used most constructions what is shown usually is any d minus 1 of your columns are linearly independent in my H that I constructed. So, that implies the minimum distances at least d most of the very famous constructions use simply that argument the actual competition of d they will just leave out it is enough. If I know that this boundary, sometimes there are some other ways of getting to the minimum distance, but this is the most popular method of constructing paretic check matrices.

So, you construct a paretic check matrix and show d minus 1 of your columns are linearly independent, but remember how many competition will this take if you have to show d minus of your columns are linearly independent. Once you select a particular set of columns finding linear dependence is very easy, so it is just a question of rank you do Gaussian elimination you get the answer. In the other hand, how many such things you have to take and choose d minus 1 plus and choose d minus 2 if it is only n choose d minus 1, one can argue it is still polynomial.

Then, choose d minus 1 plus then d minus 2 plus and choose d minus 3 so on and if d becomes some alpha n this is going to go as 2 power alpha n, so that is going to be exponential. So, it is not a small number you have to add it up and it becomes a very large number so you have to do too many of these things. So, whichever way you look at it problem is difficult but it is a good designed tool showing that is a linearly independent alright so we have few minutes left.

So, let us try and tackle this problem of constructing a H with a guaranteed minimum distance for some simple cases so d equals 0 no point in looking at anything because there is no there is no code only the reveal all zero code for d equals 1. Also, one can argue it is it is I mean it is no too difficult come up with construction very easily d equals 2 may be it is not so interesting d equals 3 is a little bit interesting. I will tell you why d equals 3 is interesting, it turns out the d equals 3 gives you some what called some error correcting capability. We have not seen it, but it turns out d equals 3 is important for that if you do not have d equals 3 you cannot correct any errors.

So, this course is about correcting errors so we have to at least start with d equals 3 let us see how to do this it is not very difficult, you can do it very intuitively in a very simple way I want to construct. So, construction of H with d let me say d equals 3, so we already note down what has to happen if you want d equals 3 d minus 1 of your columns have to be linearly independent. So, you should not have all zero column that is rolled out and note two column should be the same, but two columns can act to the third column.

You are willing to live at d equals 3, two columns can act at the same column say add to some other column it is, but you should not have identical columns. So, now if I say I have r rows in each if I have r rows in each, so my meridians is r, so n minus k is r, so fix n minus k equal to r it is important thing to fix n minus k because I am looking at columns and trying to figure out what I can do. So, you can fix n minus k and then ask the question how large can n be, so that I can still maintain d equals 3. Remember, how many different possibilities are there once I fix n minus k how many different columns can I have 2 power n minus k 2 power r, I cannot have more than that many columns, so n is clearly has to be less than or equal to 2 power n minus k.

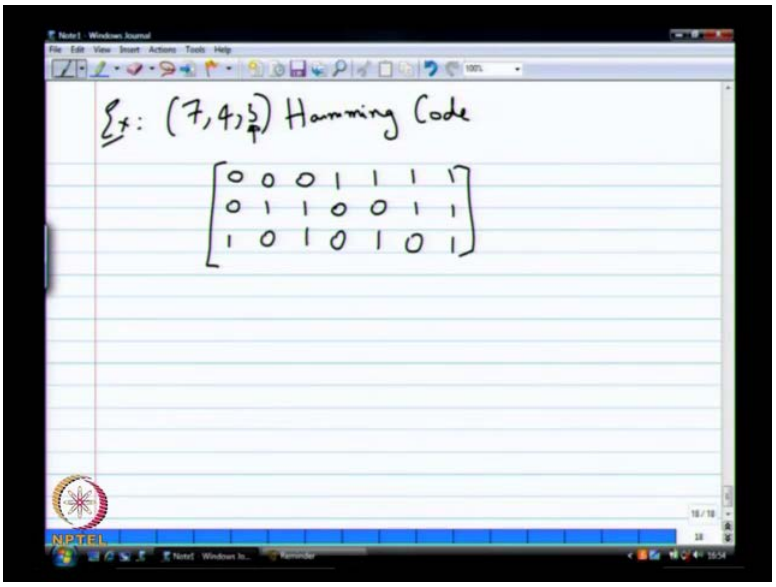
Now, how do this 2 minus 2 power n minus k what is ruled out all 0's ruled out and the reason why I had 2 power n minus k is I do not want the allowed repetition if I allow repetitions it will go on and on, but then I do not want d equals 2. I want d equals 3, I do not allow repetition, so clearly the maximum I can do is 2 power r minus 1. So, I can have n, let us say less than or equal to 2 power r minus 1 the extreme case I do not want to pick n equals to 2 power r minus 1 in which case how will I fill out all the columns, put all possible non zero orbit. It is simple enough, d equals 3 is not too hard b equals 3 is not really not hard it is an easy enough construction.

There is I mean one can easily argue that there is nothing better than you can do for a given r you want to maximize n, why would you want to maximize n for a given r these fixed an r is fixed n minus k is fixed. You are adding is fixed anything if you maximize n what will happen number of message bits is being maximized which is good for you. You know I mean it is good to have more and more message bits and if you are fraction of paretic bits, so you do not want to waste you want to maximize number of bits per I mean k by n has to be larger.

So, it is good to do that it is what about k it is going to be n minus n minus k i mean it looks like it is n minus r. So, if you pick distinct vectors here looks like it is n minus r but you have to be a little bit careful about linear dependence, linear independence. If I pick n equals to per r minus 1 what will be k, so for finding what is the problem with simply saying n minus k equals r. So, what is the problem with this equation, I have assumed what I have put here in putting here is linearly independent, it may not be linearly independent I have to check that as long it is linearly independent.

Then, n minus k becomes equal to r, then I can set the k very easily if I put n equals to 2 power r minus 1, I can definitely argue that k is equal to n minus r how will I argue that I am putting all possible orbit vectors. So, what is the in particular what vectors would I have put the vectors in a matrix. I would have put, so 1 0 0 1 0, so those things will clearly be linearly independent. So, rank will be equal to r you can argue k will be n minus 1, so that lets see a quick example may be at that point you will see what I mean by this.

(Refer Slide Time: 49:53)



Now, let me introduce the 7, 4, 3 hamming code officially name it, so I am running out of time, so I will just end here, so we saw this before. So, I told you also that it is called the hamming code, but now you know a little bit more about this code what do you know about this code the minimum distance is 3. So, that is something, so that was lot to do on, so let us stop here and will pick up from here tomorrow.