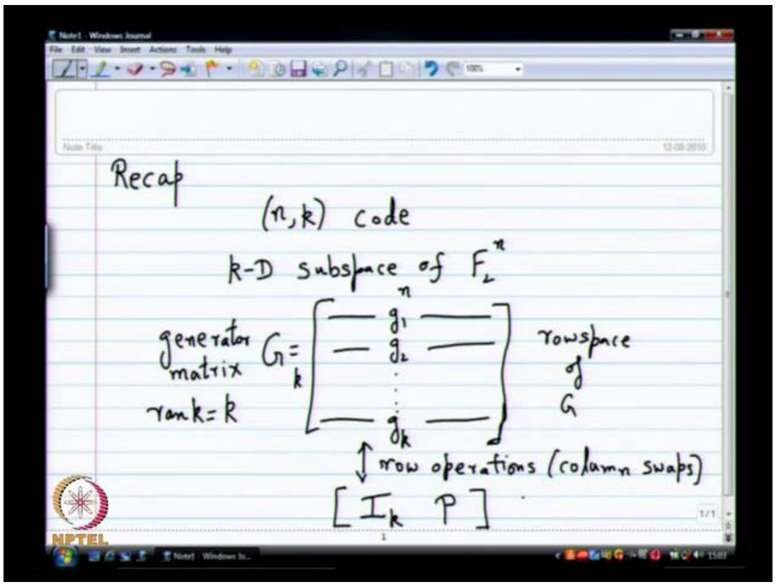
**Coding Theory**
**Prof. Dr. Andrew Thangaraj**
**Department of Electronics and Communication Engineering**
**Indian Institute of Technology, Madras**

**Lecture – 3**
**Dual of Linear Block Codes**

(Refer Slide Time: 00:12)


Recap

(n, k) code

k-D subspace of $F_2^n$

generator matrix

$$G = \begin{array}{c} \\ k \end{array}\overset{n}{\begin{bmatrix} — & g_1 & — \\ — & g_2 & — \\ & \vdots & \\ — & g_k & — \end{bmatrix}}$$

rowspace of G

rank k = k

$\updownarrow$ row operations (column swaps)

$[I_k \;\; P]$

We are going to start a quick recap of what we did last class. So, the basic thing that we saw was an idea of an n, k code; then k code is nothing but a k dimensional subspace of F 2 n remember what is F 2 n, F 2 n is the set of all n length binary vectors. You have a vector space when we define a vector space with them with modular 2 operations, so vector space over the scalars over the binary field k 0 1.

So, that is the basic idea and usual way to specify this k dimensional subspace is to specify a generator matrix k which is k cross n it is became an ugly looking matrix, but it is much better. So, k cross n matrix first row second row etcetera specify the basis for the for the n k code basically you specify a matrix and say the code n k code is the row space of g. So, clearly I am thinking of a rank k matrix, so that is it is another thing which is so important rank equals k.

So, this is pretty much all that we saw and the next thing we saw was to basically do Gaussian elimination on this matrix and convert it into what I called systematic form. So, you can do Gaussian elimination and convert it into a form which looks like this I k and then a parity part p. Remember what are the steps that are allowed in this Gaussian elimination you can do row operations which is basically row swaps and row additions and then column swaps are also allowed. Only then I will get I k P if you do not allow for column swaps what will happen is the columns of I k might be spread over the entire matrix it will look a little uglier that is all.

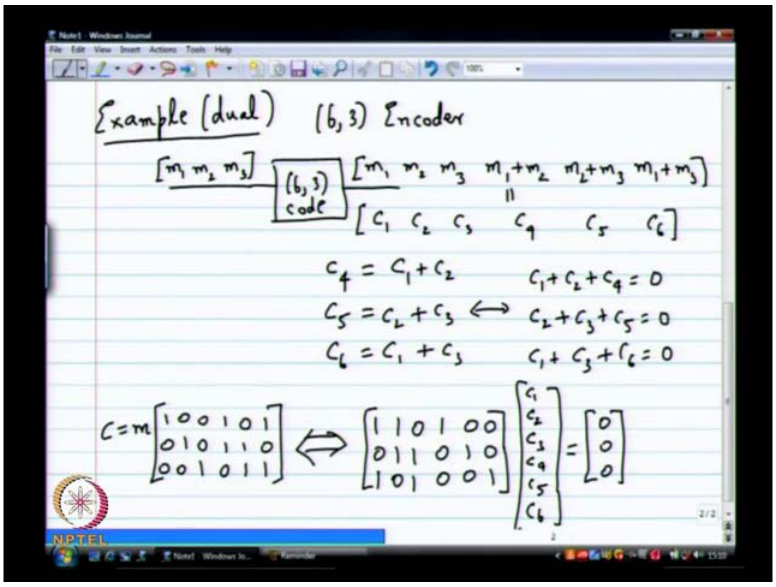
So, if you do a simple row or column permutation you get back to the original matrix so remember the row space will be altered when you do column swaps only if you assume that permutations are it is the same code. If you want to retain the same row space you should not do column swaps you should live with an uglier looking G which has the columns of i spread out over the entire matrix.

So, row operations are performed I will put column swaps in bracket a good linear algebraic question at this point just to test your a background a little bit is under what conditions will you not need a column swap under what conditions. On this generator matrix G think about it, so when you do Gaussian elimination you will have to find the pre words just with row swaps and row just row swaps.

So, you will not need any column swaps when you do the pre words, it is not a difficult condition to think about, but essentially one of the necessary conditions. For that to happen be the k column surface k columns of G should be should be also linearly independent. So, you should also get an invertible matrix that is a necessary condition on the top of that you also need the pre words to come one after the other nicely, only then you will get it.

So, that is that is the recap and now we are going to move on with the next way of viewing the same code. So, this n k code this is the normal basis view then there is also the dual view, so I was giving a brief motivation in the last class of how you can think of the 3 D space. It defines a plane either by specifying two linearly independent vectors on the plane or you define one vector which is normal to the plane. So, either define the dual or the original space itself, so here also we can do something similar, I am going to motivate it using a simple example and then I will tell you that this is general too, so let us see that.

(Refer Slide Time: 05:06)



Example (dual)   (6, 3) Encoder

$[m_1\ m_2\ m_3] \rightarrow \boxed{(6,3)\ \text{code}} \rightarrow [m_1\ m_2\ m_3\ \ m_1+m_2\ \ m_2+m_3\ \ m_1+m_3]$

$\qquad\qquad\qquad\qquad\quad\ \ [c_1\ c_2\ c_3\ c_4\ c_5\ c_6]$

$$c_4 = c_1 + c_2 \qquad c_1 + c_2 + c_4 = 0$$
$$c_5 = c_2 + c_3 \longleftrightarrow c_2 + c_3 + c_5 = 0$$
$$c_6 = c_1 + c_3 \qquad c_1 + c_3 + c_6 = 0$$

$$C = m\begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} \Longleftrightarrow \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

So, here is an example to show you how the dual space comes about example for dual remember our first example was this 6 3 code, how did we define the 6 3 encoder. Remember, I had a very simple example where I had a code where let us say the 3 bit message going in m 1, m 2, m 3 gets encoded and what comes out is m 1, m 2, m 3 and then m 1 plus m 2 m 2 plus m 3 m 1 plus m 3.

I had the code like this, now if I basically this is the code word, so I could denote it as c 1, c 2, c 3, c 4, c 5, c 6. So, every code word of this code it is formed in this way. So, basically I have 6 equations which tell me what the code word is what are those 6 equations c 1 equals m 1 c 2 equals m 2 c 3 equals m 3 and what is c 4 m 1 plus m 2 and c 5 is m 2 plus m 3 and c 6 is m 1 plus m 3. So, the first 3 equations are not really anything that you can control that is a free equation, you can pick any value you want for m 1 m 2 and m 3.

So, likewise for c 1, c 2, c 3 you can pick any 3 bits that you want and then c 4 c 5 and c 6 gets fixed in terms of the first 3 bits, so let me write that down a little bit differently. So, I have 3 equations which tell me how the parity bits are computed c 4 equals c 1 plus c 2 c 5 equals c 2 plus c 3 c 6 equals c 1 plus c 3. So, one thing I can do now is to write these equations which we are seeing as operations to get the parity bit as a condition on the bits of a code word, so what will I do? I will move the things on the right hand side to the left hand side, so equivalently I have these 3 equations c 1 plus c 2 plus c 4 equals 0 c.

So, remember whenever I write plus pretty much throughout this course it will be modulo 2 except if I am dealing with real numbers where it obviously going to be modulo 2 it will be always modulo 2. So, I know I have to write c 4 minus c 1 minus c 2 equals 0, but minus 1 is the same as plus 1 in my field k 0 1 that I am dealing with. So, it is just plus its binary exhort, so we will think about it c 2 plus c 3 plus c 5 equals 0 c 1 plus c 3 plus c 6 equals 0, so these are a different way of describing the encoding operation.

So, you can see you can go from the encoding operation to these 3 conditions we can also do the reverse. So, it is not too hard to see that you can do the reverse if you give you only these 3 conditions from here you can go to the encoder also how will you do it. You have to stare at it for a while or there is another way of doing it which is also equally the same, but you stare at it for a while and then you realize c 4, c 5 and c 6 are can be taught of as dependent variable and c 1, c 2, c 3 can be taught of as independent variables.

So, you have to pick an independent variables any which way you want and pick c 4 c 5 c 6 as dependent you will get the get all the solutions cannot be any other solution. So, the same code I can describe in 2 different ways, now what is the first description, I describe the code word as a message times a generator matrix, which look like this.

Now, I am getting a different description which I can loosely term as the dual description I will tell you why it is the dual description a code word has to satisfy every code word has to satisfy these three conditions. What are the three conditions we have written down here I can write them also in a matrix form, so let me write that out for you, so will see how it works out I want to have a matrix here.

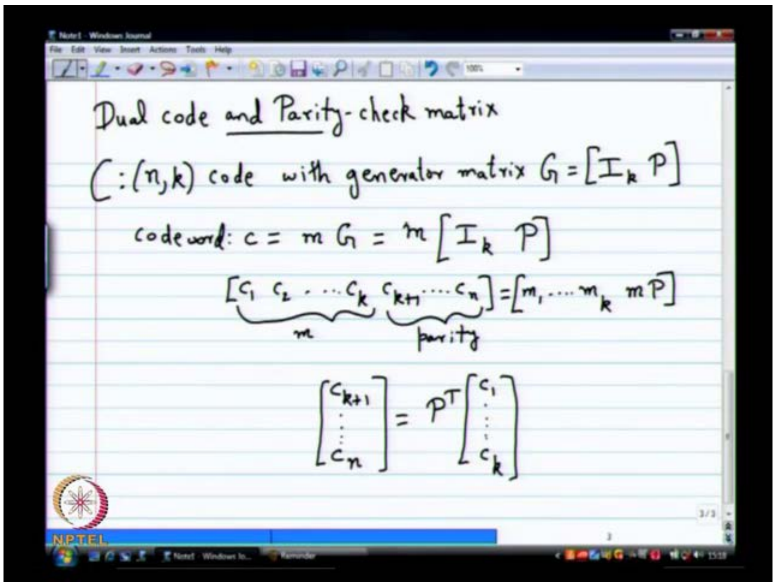
Then, I want to put c 1, c 2, c 3, c 4, c 5, c 6 here and what do I want from the right hand side, I want three 0's the question is what should I put in this matrix. So, just it should codify these equations not too difficult to see once you know how matrix multiplication occurs it is simply 1 1 0 1 0 0 0 1 1 0 1 0 1 0 1 0 0 1. So, I did not really prove it from that the way I did it its quite clear that both this descriptions are perfectly equivalent how I find the code words. I could find the code words by writing them down as m times this matrix and going through all the different possibilities for m how many different possibilities will I get 2 power 3 which is eight possibilities.

I will get all the code words I will get eight different code words or I could look at this set of equations which is a little bit different you cannot immediately create the code words from it, but the minute start will tell you how to create the code words. So, you look at the matrix a little bit you can do that, but this is also equivalent if I look at if I ask the question in the set of all 6 bit vectors give me all those vectors which satisfy these three conditions written down.

These three conditions written down here give me the set of all 6 bit vectors which satisfy these three conditions, guess what I will get the exact same code words that I got before there will be only eight of them and you can enumerate that. So, how do it in this particular case like I said I mean you have to stare at it for a while and realize that c 1, c 2, c 3 are Independent variables c 4, c 5, c 6 are dependent variables. You vary them a lot you can get it, so the description on the left is the basis description and the description on the right is the dual description.

You can see why it a dual description I have given you, I have given you a matrix on the left hand side whose row space is the code on the right hand side also there is a matrix how is the code related to that matrix, so there is a name for that space. So, you have the row space which is all linear combinations of the rows what is the space on the right side now it is a null space set of all vectors. So, that this matrix times that vector equals 0, so that is the null space on the left hand side I have the row space description on the right hand side I have a null space description of course the matrices are different, but clearly like I derived. Now, one can be derived from the other we will see the general way of doing it also, this is the motivation or the general basic idea behind the dual description this can be extended in general. Also, we will do immediately any questions quick questions on this, now is the good time to ask.

(Refer Slide Time: 13:36)



Dual code and Parity-check matrix

$C:(n,k)$ code with generator matrix $G=[I_k\ P]$

codeword: $c = mG = m[I_k\ P]$

$$[\underbrace{c_1\ c_2 \cdots c_k}_{m}\ \underbrace{c_{k+1} \cdots c_n}_{parity}] = [m_1 \cdots m_k\ mP]$$

$$\begin{bmatrix} c_{k+1} \\ \vdots \\ c_n \end{bmatrix} = P^T \begin{bmatrix} c_1 \\ \vdots \\ c_k \end{bmatrix}$$

So, let me do the let me do the dual code and parity check matrix, so that is the title for what we are going to see next it might actually be parity check matrix and dual code if I might do by the way. So, that is what we are going to see next, so if we start with an n k code with generator matrix G which is in systematic form, so like I said in any major generator matrix that you have can be put into this form with enough columns swaps. So, in case you cannot do it if it is not given in systematic form you do a series of Gaussian elimination steps get it to systematic form.

So, we can assume at this way it is not a big problem, so this is now we start with k, so let us say this is this code is called c is an n k code with generator matrix g. So, how do I think of a code word of c any code word of c is computed by m G code word, m G which is m times I k P is that. Now, just repeat the same steps I had before, so if the code word is c 1 c 2 so on till c k. Then c k plus 1 c n, I know this will be equal to what m 1 through m k and then I will write it as m times P.

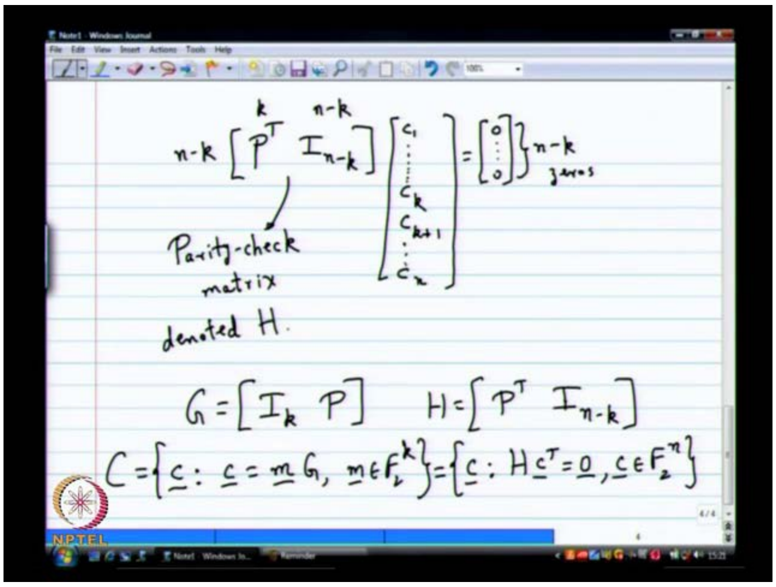
So, is it I know this is I mean if you have any experience with mat lab this kind of notation will be easy to you it is like all mat lab notation contaminations even I read m P it means you get a parity vector which has to go there and fill out the whole vector. Now, this k is the message part this is the parity part all that is known, now what should I do what should be the next step and then yeah moving to some side. Then through the idea, so that the similar step to what I had before let me write it down a little bit differently see how it works out.

So, the same thing I am going to write down little bit differently and in in a column vector formed. So, it might be a bit more difficult for you to see, but hope fully it is clear enough so the same as before, but I am doing it I am doing it little bit differently you want to write this as how should I write it. So, I have to do a transpose because m P is a row vector, I want a column vector, so I should do m P whole transpose which is the same as P transpose m on the right.

So, what is m if I transpose it, I simply get c 1 through c k, so stare it for a while and convince yourself it is the exact same thing. So, I have just written it in a little bit different form, I am writing c k plus 1 through c n equals m P and taking transpose just to get the same format as before.

Now, what do we do what was done in the next step after c k 1 plus through c n i should move to the other side. So, how will I do that when take it to the other side we quit the 0, then what should I do I want to write one matrix times the entire vector. So, all that I will do in one step in the next slide, but hope fully it will be clear to you, so I am going to move to the next pic next slide, I will just do everything in one step.

(Refer Slide Time: 18:15)



$$\begin{array}{cc} & \begin{matrix} k & n-k \end{matrix} \\ n-k & \begin{bmatrix} P^T & I_{n-k} \end{bmatrix} \end{array} \begin{bmatrix} c_1 \\ \vdots \\ c_k \\ c_{k+1} \\ \vdots \\ c_n \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \Big\} n-k \text{ zeros}$$

Parity-check matrix

denoted H.

$$G = [I_k \;\; P] \qquad H = [P^T \;\; I_{n-k}]$$

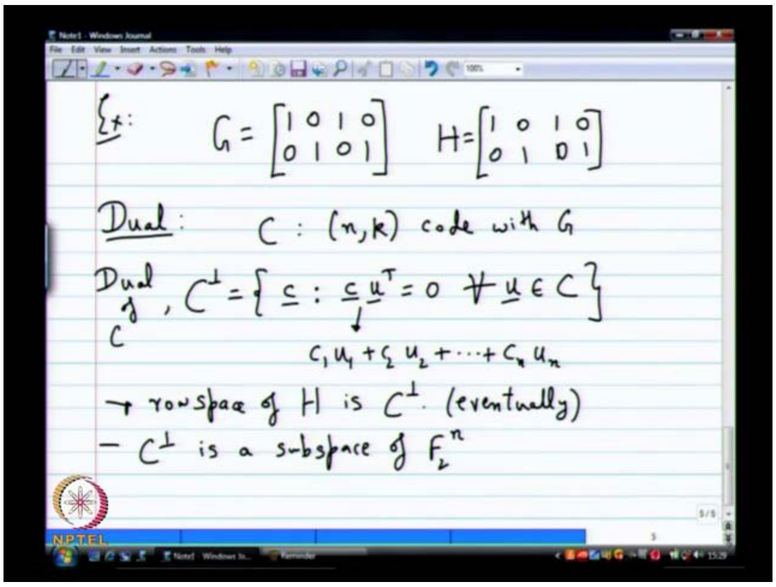
$$C = \{\underline{c} : \underline{c} = \underline{m}G, \underline{m} \in F_2^k\} = \{\underline{c} : H\underline{c}^T = \underline{0}, \underline{c} \in F_2^n\}$$

So, you will basically get P transpose I n minus k multiplying c one through c k and then c k plus one through c n and what should come on the right hand side 0. So, this is just same as which what we had before except that I have written everything in a convenient symbol matrix form the same thing I have moved everything to one side and I wrote. So, if you want to pass this P transpose times c 1 through c k plus i times c k plus 1 through c n i times c k plus 1 through c n is simply the column vector i. So, just draw you get the exact centre of that so one more thing to really check real quick is this is the dimension what is the dimension for P k by n minus k what is the dimensions for P transpose n minus k by k.

Then, you have an n minus k here remember that, so you get the same n columns as before, so how many how many 0's will be here n minus k 0's is that. So, this matrix is called the parity check matrix, usually its denoted h, the generator matrix is usually denoted G the parity check matrix is usually denoted h. You see there is very simple relationship between G and H if you have if you have it in systematic form, so G is given in systematic form as I k P then what happens to H is P transpose I n minus k all right the code.

Now, has 2 different descriptions set of all code words such that c equals m times G m belonging to F 2 k which is also given as set of all c such that H c transpose equals 0 c belonging to F 2 n. We saw it quite easily that these 2 are the exact same descriptions for the exact same code and if you have it in a systematic form it is very easy to write down k. One is the basis description the other is the dual description, so we all ready saw an example, so I am not going to give another example for the 6 3 code. We saw an example you can also see other more crazy examples, so let me give you one more example which is may be a little bit more crazy, so you might think will get your thinking.

(Refer Slide Time: 22:20)



Ex:

$$G = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} \quad H = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

Dual: $C$ : $(n,k)$ code with $G$

Dual of $C$, $C^{\perp} = \{ \underline{c} : \underset{\downarrow}{\underline{c}\,\underline{u}^T} = 0 \;\forall\, \underline{u} \in C \}$

$c_1 u_1 + c_2 u_2 + \cdots + c_n u_n$

→ rowspace of $H$ is $C^{\perp}$. (eventually)

– $C^{\perp}$ is a subspace of $F_2^n$

So, let us see G is 1 0 1 0 1, so it is a bit of simple looking generator matrix, so what will if you form H from it yeah we will put the exact same matrix something strange, something like this can never happen in the real space. If you look at r 4, if you want to describe a 2 dimensional sub space of r 4 you can describe 2 base of vectors, but the same basis vectors it can never describe the dual space. So, the dual and the regular subspace will not intersect at all they will intersect only in its 0.

So, here you have the strange case where both are the same, so it is I will tell you why this happens, so you will see there are lot of reasons why this happens, but see both of the strange things to keep in mind. So, let us now define the dual and then connect it with the parity check matrix, so it is not too difficult, so we will do that first. So, suppose I have an n k code c the dual is defined as follows c pop is its denoted c pop dual of c denoted c pop it is the set of all c such that c u transpose equals 0 for all u in c.

So, you see all ready there is some minor inconsistency in my notation with some vectors I am putting a bar below and in some vector I am not putting a bar below, usually I will put a bar if there is some confusion of using c and C. So, it is just to remind yourself that small c is the code word and capital c is the is the code itself it is the set of all code words. So, usually there would not be confusion, but sometimes if I do not write the capital c really big you might be confused, so that is why I put the bar b.

So, what is this c u transpose which is basically c 1 u 1 plus c 2 u 2 plus so on till c n u n remember everything is modulo 2. So, I never wrote modulo 2 here, but it is clearly modulo 2 any matrix multiplication I am doing with binary vectors binary vectors and binary matrices is modulo 2, it is always assumed to be modulo 2. So, this is defined as the dual on first glance it looks like maybe there is no connection with the parity check matrix. There is there is some connection what is the connection that you can may be see real quick, so words you can see many corrections, but in terms of equations you see that c u transpose equals 0 is giving you some hint.

So, H times c transpose is zero for every code word in the code something that we can identify from this is very important. So, we will come to that soon enough but before that we have to first see what is this dual. So, we will try prove some abstract properties for the dual and then quickly see what it is actually, so you will see the dual ultimately will become the row space of h. So, you can imagine that this happen you know that mean so the definitions work, but we mean that this is quite clear from linear algebra but I want to develop this little bit more slowly, so that the impact of this is we get the impact of this is quite important.

The dual code is very important many of the decoding algorithms work with the dual code so it is a very important idea to get clear ahead across. So, this will we get to this eventually, but let us see how this works out, so the first claim I want to make the first thing we will the first property that we will show is that c pop is a subspace of what F 2 n I would not rigorously prove this.

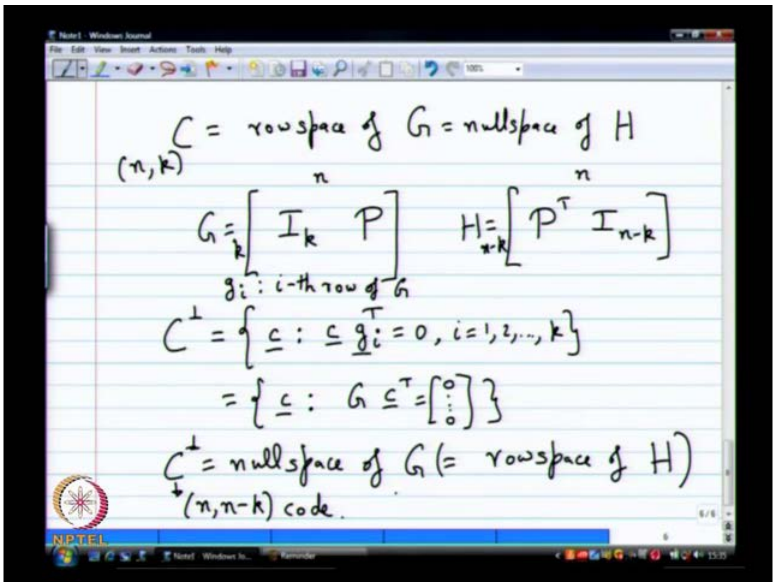
I will simply tell you what the proof is it is not too difficult, so what is the subspace of a vector space what conditions create a subspace any 2 vectors in the space. When we are doing linear combinations with them, you should be in the same subspace what the linear combinations are.

You take a vector and multiply with a scalar for us the scalar is really nothing 0 or 1, so it will be trivially satisfied no problem, but when you add 2 vectors I should get another vector in the same space can we prove that with this definition. So, if I take a c 1 which satisfies this condition and take a c 2 which satisfies this condition c 1 plus c 2 will also satisfy this condition, so it is a very easy thing to show.

So, we are able to show c pop is the subspace of F 2 n, the next thing is with the dimension and the basis. So, I know c pop is a subspace, I know c, c is an n k code which is clearly a subspace and I have I have a nice description of it, maybe I have the generator matrix, so maybe I should include that say that generator matrix g.

So, that is the definition for the n k code, now I might want to find the generator matrix for c pop then that completes my description of c pop so I am able to quickly see it is a subspace there is no problem with that. Then how can I find the generator matrix, now I am going to invoke some basic ideas from linear algebra and quickly get to the result, so without too much of worry.

(Refer Slide Time: 29:14)



$\underset{(n,k)}{C}$ = rowspace of $G$ = nullspace of $H$

$$G = \begin{array}{c|c} & n \\ \hline k & \begin{bmatrix} I_k & P \end{bmatrix} \end{array} \qquad H = \begin{array}{c|c} & n \\ \hline n-k & \begin{bmatrix} P^T & I_{n-k} \end{bmatrix} \end{array}$$

$\underline{g}_i$ : $i$-th row of $G$

$$C^{\perp} = \{ \underline{c} : \underline{c}\, \underline{g}_i^T = 0,\ i = 1, 2, \ldots, k \}$$

$$= \{ \underline{c} : G\, \underline{c}^T = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \}$$

$C^{\perp}$ = nullspace of $G$ (= rowspace of $H$)

$(n, n-k)$ code.

So, the first thing we know is c is the row space of g, we know that do we know that or not right so that we know. So, essentially I am looking for what if you think about the dual in terms of row space column space etcetera we also have another description of c which becomes the null space of h. So, which is some other matrix, so it seems like slightly different definition.

Now, suppose I ask you for the null space of G what will be the null space of g, so let us let us look at this a little bit more clearly? So, G is your basis so let us say I k p, so H is going to be a very simple again the transpose I n minus k. I have c described in this form, so notice what is c pop c pop is set of all vectors in F 2 n which are octagonal to all vectors in c.

So, remember if I want something to be orthogonal to all the vectors in c it is enough if it is octagonal to the basis vectors of c. So, that is the first idea from algebra that we will use c pop I defined it as set of all vectors which are orthogonal to every vector in c. So, equivalently simply becomes set of all c such that c times G I transpose equals 0 where what is G I. So, rows of g, I remember is I through of g, so this condition I can write it slightly differently which will clearly tell you where the definition comes from G times c transposes. So, that it is the same thing c times G transpose 0 you transpose that what do you get G I times c transposes 0.

This is true for every row, you just put it all together into a matrix equation you get the answer alright this are all completely equivalent ideas this comes, because octogonality with all vectors in the code is the same as octogonality with the basis vectors. So, once I do that, I am done no more problems, now the answer just falls out of the equation if I define if I look at G and think of G as the paretic matrix of some code. That code be it will be c pop, so that is the idea alright so that that just immediately gives you the answer.

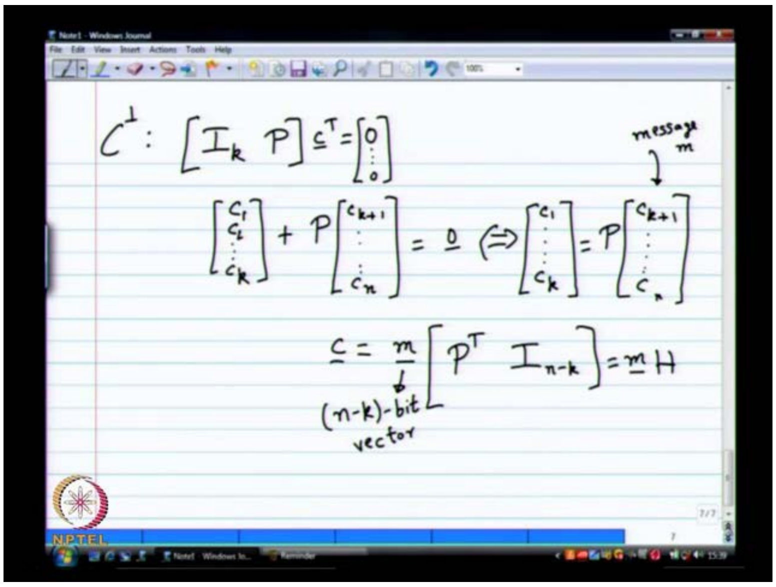
So, c pop will be null space of G which you can show will be the same as those space of H also I have not proved this, but it will be the same as that. So, the first we have proved the second part I have put in bracket because it is necessarily proved, but it is kind of it is kind of clear. If you know enough linear algebra, it may be it may be we will prove it sooner is there any question no question do not want to ask in the public. So, hopefully this simple derivation is clear I mean there is nothing much in this but the first time you see it might be a little bit confusing too many spaces floating around.

So, let us let us let us do a little bit more of quick competition with c pop, so remember his dimension n minus k cross n G has dimension k cross n and c became an n k code G is the paretic check matrix for c pop and it is dimension is what k cross n. So, what kind of kind will c pop be n comma n minus k code so all these are a kind of interesting and obvious properties. So, good to know this is that is that clear just by looking at this dimensions if you have paretic check matrix for a code c it is going to have dimension n minus k cross n of course of these full rank assumptions are ready already made it.

So, where is it you would not you would not' get this answer so n minus k of c pop becomes all space of G which is paretic check matrix. So, c par and just by looking at the equations you see c power n, n minus k code. Now, how I quickly see that the null space of G and row space of H has to be the same you can go through the same operation as splatted. So, you start with a null space of G define a code and then do the left to right switch with I and P you will get back your h.

So, we did some c equals m G we went to H c transpose equals 0, now you start with G c transpose equals 0 and you are do the same thing and go to some c equals m h. So, you will get the same thing if you want I can do that again let me quickly do that it is not a big deal. So, what do we have I thought I had a 50 clock is it, so it gone, so how do I will go on just give me a warring when 50 minutes are up, so let us see that.

(Refer Slide Time: 36:27)



$$C^{\perp}: \begin{bmatrix} I_k & P \end{bmatrix} \underline{c}^T = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_k \end{bmatrix} + P \begin{bmatrix} c_{k+1} \\ \vdots \\ c_n \end{bmatrix} = \underline{0} \iff \begin{bmatrix} c_1 \\ \vdots \\ c_k \end{bmatrix} = P \begin{bmatrix} c_{k+1} \\ \vdots \\ c_n \end{bmatrix}$$

$$\underline{c} = \underset{\substack{\downarrow \\ (n-k)\text{-bit} \\ \text{vector}}}{\underline{m}} \begin{bmatrix} P^T & I_{n-k} \end{bmatrix} = \underline{m} H$$

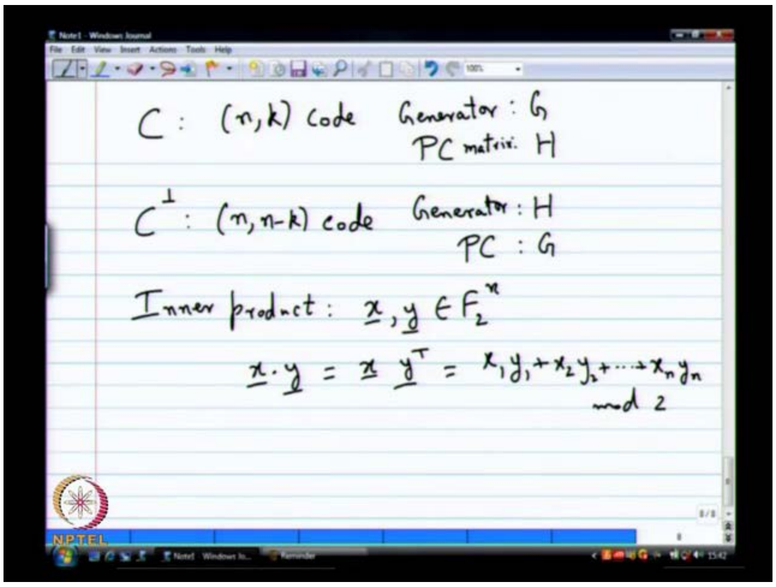
So, now I have the dual code described as null space of G which is what I k P times c transpose is 0. This is the equation that I have, so what should I do, now more interesting to answer this question this way. So, let us see so write it differently write it as c one c 2 so on until c k plus P times c k plus one all the way down to c n equals 0. I will put a 0 over the bar below put as many 0's you need. Now, this is the same as what c one through c k equals P times c k plus 1 I know enough of you are saying minus the same as plus. So, it goes to other side without any sign change it is the nice thing about dealing with binary.

So, you have this now which is the independent variable and dependent variable c 1 through c k are dependent variables. So, remember it your m should be c k plus 1 through c n, so the role of the paretic and the message case which in this in this row. So, you have to think of this as your message m these are the independent variables, so if I do that, I can write c as m times what P transpose n I what is this I. Now, I n minus k and remember m is an n minus k bit, so I simply get the same thing as before which is m times.

So, my c power which started as null space of G became the row space of h, so all these things are very obvious if you have if you have linear algebras where you search in your mind. I think this derivation should convenience you that it is the same, so these are quick and dirty derivations using the systematic form if you do not use this systematic forms notation becomes bit messier, but you get exact same answers. So, I will not be direct it will be somewhere else to collect the different columns and put the b in different places will get the same answer so that is the idea.

So, let me go back to the previous slide hopeful this derivation is clear, so this is this is what is very important, now I can erase the bracket because I clearly proven that to you and now it is all the more clear why n comma n minus k code. So, it is row space of H it has got n minus k vectors, so clearly minus k goes which l in the independent you get an n m.

(Refer Slide Time: 40:17)



$C$ : $(n,k)$ code   Generator : $G$

PC matrix: $H$

$C^{\perp}$ : $(n, n-k)$ code   Generator : $H$

PC : $G$

Inner product : $\underline{x}, \underline{y} \in F_2^n$

$$\underline{x} \cdot \underline{y} = \underline{x}\,\underline{y}^T = x_1 y_1 + x_2 y_2 + \cdots + x_n y_n \bmod 2$$

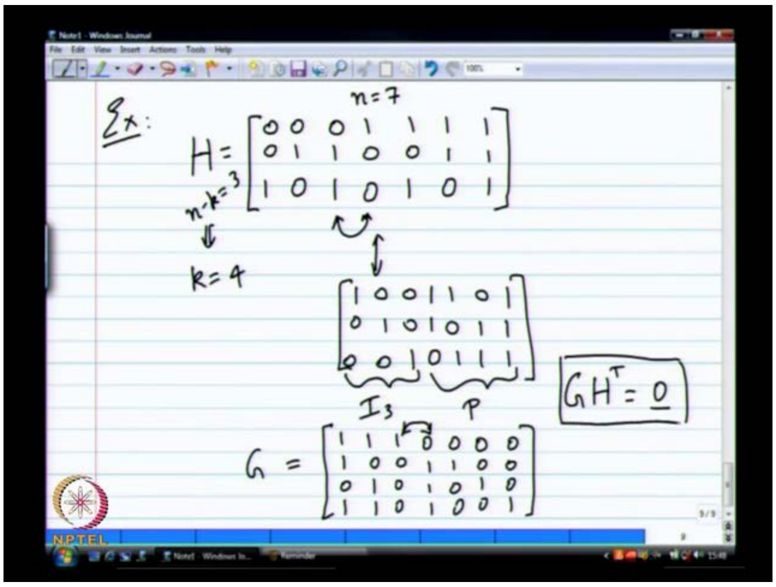
So, what does what does the summary, now in case we have seen being an n k code generator G parity check I will use pc for parity check matrix H now see pop is what it is can n comma n minus k code what is the relation between c and c pop. You take you take a vector and see it is going to be octagonal to all the vectors in c pop you take any vector in c pop. It is going to be octagonal to all the vectors in c and you have to do a switch here what is the generator for H and parity check is G s this is our switch.

So, this duality or occasionally is with respect to inner product the usual inner product except you have to do model 2. So, that is the inner product involved in this duality is basically if you have 2 vectors x comma y in f 2 n x dot y is basically x y transpose which is basically x one y one plus x 2 y 2 plus so until X n y n model 2. So, those duality is with respect this inner product, so that is what I mean when I say octagonal take a vector in c vector in c pop through this operation you will get 0.

So, this inner product is not as nice as the inner product and r n and all that, so r intense you know that x dot y x dot x for intense can never be 0 in the inner product, but extra decks can be very easily be made 0. In this, even if x is not 0 right if you have any x which has even number of once if you do extra decks you going to get 0, so vector can be octagonal to itself which is very strange idea an dense space.

So, that is why you get such examples I had an example for you right c and c pop were exactly the same you can have c equal c pup that is because the same method can be octagonal to itself. So, that is the bit of a contrary into idea about this binary vector spaces this are finite vector space over finite fills some strange things happen with the regular inner product that you are used to the model 2 is a very nice inner product.

(Refer Slide Time: 43:27)



Ex:

$n=7$

$$H=\begin{bmatrix}0&0&0&1&1&1&1\\0&1&1&0&0&1&1\\1&0&1&0&1&0&1\end{bmatrix}$$

$n-k=3 \Downarrow k=4$

$$\begin{bmatrix}1&0&0&1&1&0&1\\0&1&0&1&0&1&1\\0&0&1&0&1&1&1\end{bmatrix}$$

$\underbrace{\phantom{000}}_{I_3}\ \underbrace{\phantom{0000}}_{P}$

$$\boxed{GH^T=\underline{0}}$$

$$G=\begin{bmatrix}1&1&1&0&0&0&0\\1&0&0&1&1&0&0\\0&1&0&1&0&1&0\\1&1&0&1&0&0&1\end{bmatrix}$$

So, that is with the dual let us see a little bit more of a complicated example and see some nice in the last 4 minutes that I have. So, let me do, let m take let me take generate a matrix or let me take a parity check matrix. So, this is the parity check matrix, this happens to be a parity check matrix, so very famous code which is the 7 4 humming code and the first of all how do I know that this is the seventh parametric matrix for a 7 4 code. So, that is the first question given the parametric matrix you have to find out the dimensions n and k for the code, so n equals 7 that is I mean that the basic counting.

So, it is the number of callers what about, so you have see n minus k and how do you find n minus to k it equals to rank of the matrix h. In this case the rank is obviously 3 how do you quickly say the rank is 3, the first second and 4th column are clearly indent and this that becomes the rank and that is 3. So, rank can cannot be greater than 3 you know clear right it has to be either minimum of less than minimum, so n minus k equals 3 n is 7 which implies k 4, so this is in place k is 4.

So, the rank competition is important I am not even writing it down, but hopefully it is clear to you why does the matrix have full rank equals 3 because first second and 4th column have are linearly independent. So, because 0 0 1 0 1 0 and 1 0 0 now you have to go through your Gaussian elimination and find the parity check generate a matrix corresponding to this. So, that might be a good exercise to try, but it is easy to do a rows column first apply a formula and then undo the column swap to get the valid generate a matrix.

So, it is good to when you deal with row space column space it good it is good not to do column swaps, but then if you do not do column swap the I P formula will apply then you will get confused. So, the best thing to do is do the column swap, but keep a record of it know which columns you have swapped after you do the I p, then do the P transpose I what do you do you undo the column swap. Then you do not have to do any permutation it is the exact same match so the first step I have to do is column swap here I will swap this 2 columns and then I will rearrange the rows.

So, I get into the form I P which is c not the standard form for the parity check matrix we always thought of it as P I. It does not matter the same formula will hold whether it is I P or pi just I have to do put the P transpose the right location I do the column swap and then i rearrange the rows. So, I get an equivalent from which looks like this 1 0 0 1 0 1 0 1 0 0 0 1 1 1 0 1, i am sorry 1 0 1 1 and then 1 1 no 0 1 1 1 1 is that so that is the form I get.

So, this is my I 3 and this is some P so how do I get G corresponding to this I have do a P transpose as an I 4. So, you do a P transpose you get 1 1 0 1 1 0 1 1 and remember method and 4th column got swapped I should put I here. Then 0 1 1 1 here and then I would get the exact same so this is anything but a P transpose i with this swap is that.

Now, you might still wonder if I did all the swapping and operation correctly there is quick way of checking what is the quick way of checking. So, each row if H has to be octagonal to each row of G and that is actually necessary and sufficient condition enough, if you check for those two, I know that the row space H and row space of G are octagonal, but to check if row space is octagonal. How can I check you can simple check the row it is enough the base of octagonal are in place everything is octagonal that you can check essentially you check G H transpose equal 0. So, you can check that that will be valid condition, so let us stop for this lecture and take a break and come back and do the next lectures.