Convex Optimization

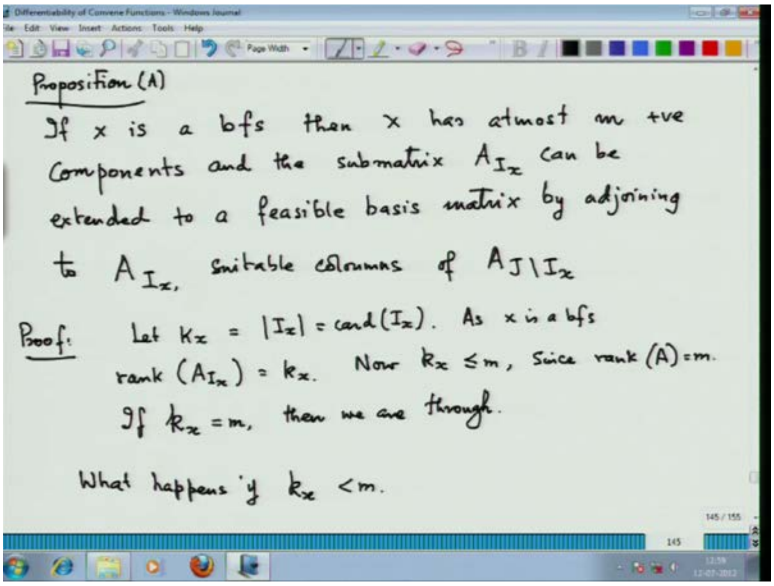
Prof. Joydeep Dutta
Department of Mathematics and Statistics
Indian Institute of Technology, Kanpur

Lecture No. # 24

Welcome once again to this little journey we are taking on linear programming as a part of the our course on convex optimization. Now, what I am going do today, these are the two fundamental theorems of linear programming. A proposition first which will help us to prove the fundamental theorem or linear program which has two two components. At every feasible set, if it is really non-empty, it will have a bfs and every optimal solution; if there is an optimal solution, there is a there is an optimal solution which is also in bfs.

Now, since most of you would be seeing this course through the You Tube, I would and you have the facility of going back and forth which is not possible in real time. So, it is very important that you look over this proofs which are now carry out of this fundamental theorem, because the in the proof of the fundamental theorem, the whole idea of pivoting, of the whole idea of the technique of doing simplest method is inherent. So, I would request to you that just do not look at this lecture once, but look at it two-three times. So, you get a hang of how the proof is done. It is those simple linear algebra, but it has slight it means slight tricks. So, just have a look at them.

(Refer Slide Time: 01:39)

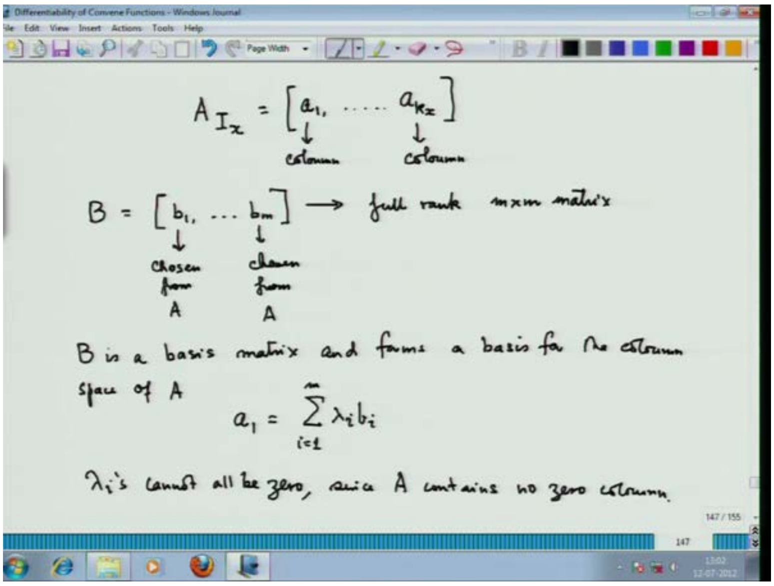


So, I start with this very important proposition which is needed to prove the fundamental theorem of linear programming which says that if x is a bfs that is I give you a bfs, then x has at most m positive components, because the blank of the matrix is m, we can at most have a m cross n sub matrix which is invertible, which is a full rank.

So, there cannot be more than m positive components in the basis in the basic feasible solution, because rest the non-basic part is considering as zero. Though some basic solution can become zero which is the case of degenerate basic feasible solution, but it cannot be more than m. And the sub matrix A I x which we have done, which we have defined earlier in the last class or in the last lecture that can be actually formed in to a feasible basis matrix by adjoining some columns from the matrix A J set minus I x. So, let us get into the proof of this. So, write it down step by step.

So, let k as before let k x is the cardinality of the set I x, which can also be symbolize like this, this absolute value signs. Now, as x is a bfs, rank of A I x is nothing but k x. Now, k x is less than equal to m, since the maximum number of linear independent columns or rows you can have in the… This matrix is m, it has m rows and some columns and the say k x columns. So, the number of linearly independent rows that you column that you will have cannot exceed k x. So, if x is the bfs it is k x, so and that k x cannot exceed m. So, x x is the bfs, this is by definition of bfs, this is true and now that k x cannot be more than m, because m is the maximum number of linearly independent columns allowed, because the matrix rank is m. This is true since rank of all matrix is m. Now, observe that if k x is equal to m, then we are through. The most important question is what happens when k x; what happens if k x is strictly less than m and that is what we are going to look into.

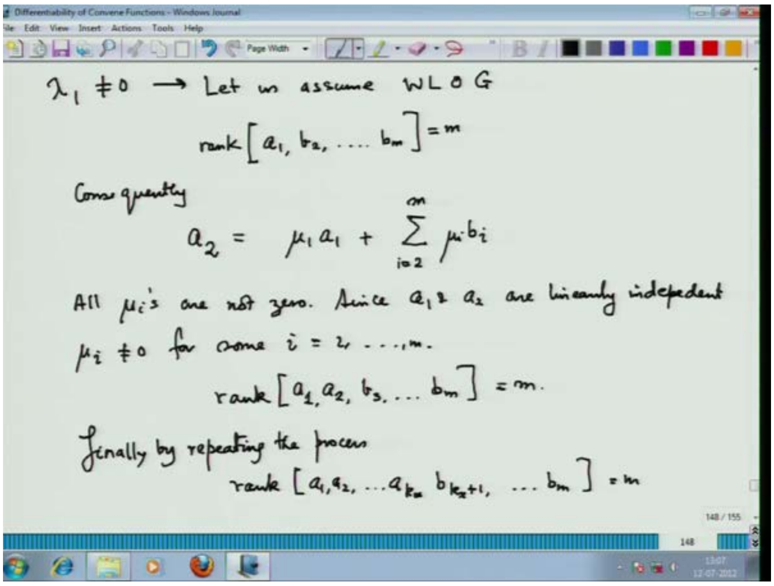
(Refer Slide Time: 04:55)



Now, what is this matrix A I x? It is a matrix formed of columns a 1… I I am just because we have assume without loss of generality that the first k x columns belong to the matrix A I x. So, this is the matrix A I x and all of these are columns of the matrix. Now, from the matrix A construct the matrix B with m columns. So, this is B is a full rank m cross m matrix. Now, this b 1, b 2, b m these are also chosen from m. These are all chosen from the set A sorry not the set A, I want to correct myself, chosen from the set from the matrix A. So, these are the column chosen from among the columns of A, and you form a linearly independent set of columns and then what you have is a full rank matrix.

Now, so B is a basis matrix and forms a basis for the column space away. We will you should know what is a column space away, if not just go and brush up your linear algebra bit, because we need a bit of linear algebra; quite a bit you see. Now, what does this show? It is simply shows that the column a 1 can be written as summation lambda i b i, i is equal to 1 sorry m vectors, I guess I am quite adopted making mistakes specially in this sort of things, anyway just a bit of fun on the way. And you see here lambda is cannot all be 0, because if all are 0 then a 1 is 0, but I have said that the matrix A contains no 0 column. So, lambda i(s) cannot all be 0, since A contains no 0 column.

(Refer Slide Time: 08:05)



So, let me assume for simplicity that lambda 1, let us assume this without loss of generality; let us assume without loss of generality WLOG. So, we assume lambda 1 is not equal to 0. So, what would happen? I can express now b 1 in terms of b 2, b 3, b dot, dot b m and a 1. So, I can basically replace b 1 in the expression in terms of a 1. Hence if I now make the new matrix, I will make a new matrix where I keep the a 1 column, I will now will change the matrix b of it and then I keep b 2. Then basically what happens I have just change one of the vectors, this is called replacement theorem in linear algebra that we will change one vector, but you still get a basis. So, the rank of this a 1 b 2 dot, dot b m, this would still remain m, because this would now become linearly independent. Because you just write b 1 in terms of a 1 and that is enough; b 1 in terms of a 1 and the others.

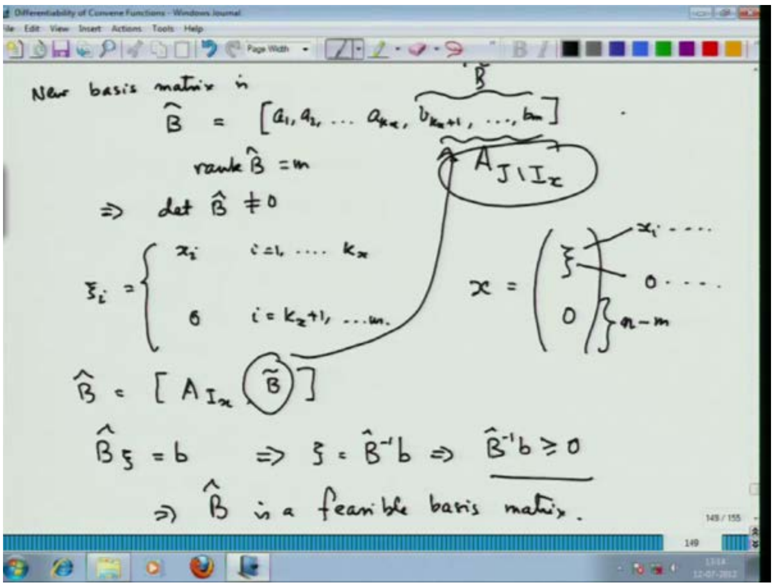
Consequently this now becomes a basis for the column space. I can write a 2 as mu 1 a 1 sorry i is equal to 2 not 1, because the b 1 is not replaced by a 1. Now, can mu 1 b equal to 0, mu 1 b not equal to 0 see. So, all mu i(s) are not 0 of course, because a 2 is not 0. Now, a 1 and a 2, a 1, a 2 dot, dot, dot a 1, a 2, a 3, dot, dot, a k x they form a linearly independent set of vectors. So, a 2 and a 1 is linearly independent. So, you can not have all the mu i(s) to be 0, so a mu from 2 to m. So, if all these are 0, then you have the mu 1 cannot be 0, so a 2 equal to mu 1 into a 1. So, a 2 would be linearly dependent on a 1 which is not true, because a 2 and a 1 are linearly independent. Since think over it a bit, a 1 and a 2 are linearly independent, because they are from the matrix A I x; mu i is not equal to 0 for some i is equal to 1 to sorry i equal to 2 to m, some i between 2 to m, this is not equal to 0; that is one step.

So, what I am able to do now, I can replace a b 2, because mu 1 would be of course, has to be zero. So, assume that mu 2 is not 0, so and the rest is 0. So, what you can do? If mu 2 is not 0, you can replace b 2 in terms of you can replace b 2 and express it in terms of a 2 a 1, b 2, b b 3, b 4, dot, dot dot b m. So, min(s) now I can make a new matrix by altering the columns of b that is a 1, a 2, b 3, b m and the rank of this matrix would become m. So, you can repeat the procedure. So, we had have in our hand in our disposal k x columns of a k x. So, what we can finally have? Finally by repeating the process…

(No audio from 11:55 to 12:12)

b k x plus 1, this indexing and b m, this is a m.

(Refer Slide Time: 12:26)



Now, let us now our new basis matrix, a new basis matrix is b hat which be give us follows.

(No audio from 12:29 to 12:40)

Is a 1, a 2…

(No audio from 12:44 to 12:56)

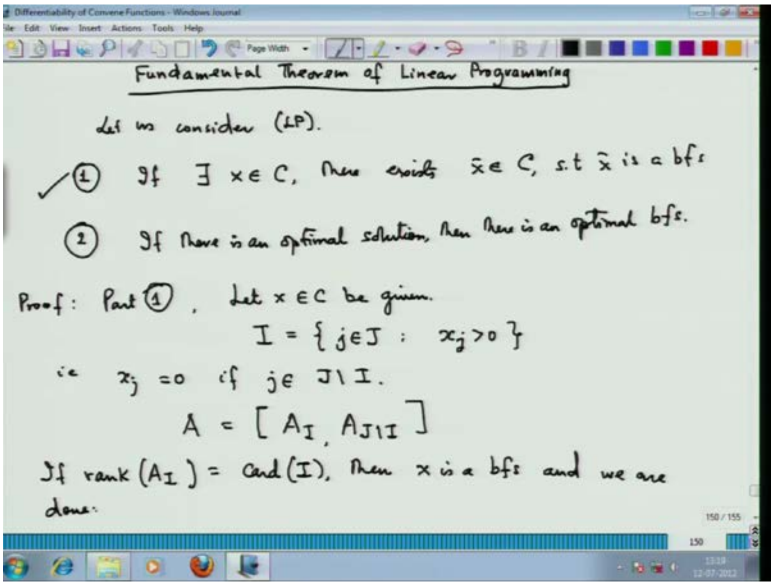
Now of course, you have this, is not sorry I should draw this. So, you have this B and rank of B hat we have just shown is m. So, B is a square matrix, m cross m full rank matrix. So, this implies the determinant of b hat is not equal to 0, fine. Now, x was a bfs, now psi i, let me set this psi i equal to x i, for i equal to 1 to k x and psi i is equal to basically I should be more nicer writing like instead of writing like this, I can write like this which is standard in books; psi i is equal to x i if it is this and if 0, if i is equal to… Now, if I look at this matrix, if I look at this then what is b hat in to… So, b hat is A I x and some other part. So, only the elements with A I x, the A I k x this this matrix. So, B hat actually has been broken into A I x and some matrix which I am writing for example N for the time (( )) or may be better way to it write it as say B tilde.

Now, A I x, because it is a basis matrix and I have assumed that… I have at most k x, I know that there are at least means there is at least k x, for sure k x non-zero quantities right, because the rank is k x of A I x. So… And the remaining I am assuming zero. So, if I put this then if I multiply this x i comes from my original x the bfs. So, what would happen? Because in the original bfs from k x plus 1 to m everything is 0; that is surely, because because of the rank i rank i is equal to k x, k x is the number of non-zero quantities.

So, what I have now what a basis matrix and using the given bfs; I have constructed new variable which is like this, which is exactly basically the B original bfs. So, now I can show that B hat of psi whose components are psi 1, psi 2, psi n, B hat of psi. So, up to beyond m it is all 0. So, up to m what is the situation? So, basically my original x is not divided like this psi and 0, this psi is what I am now writing like this, xi and x xi and 0; xi for some components, 0 for some components. So, this… So, part of the first m is non-zero and part is 0, and the remaining n minus m the non-basic part has to be 0. So, B of psi, B hat of psi is anyway equal to b right, because B of x is equal to b, so B hat of x is obviously equal to b, but it is B hat of x is what, B hat of psi plus A I x of psi plus 0, A I x of psi is actually giving me nothing but b right. So, you can easily prove this fact.

Now... So, this would imply because B hat is having determinant not equal to 0 which means that B hat is an invertible matrix, so psi can be written as B inverse of b. And of course, psi is greater than equal to 0 which implies that B inverse of b. So that is exactly what we wanted to prove some not sorry b B hat inverse. So, what we have done by adjoining matrix may certain... So, this b x plus b k x plus 1 dot, dot, dot b m, this part must come from the matrix A which is quite natural. So, adjoining these things, adjoining these things to the matrix A I x that is B hat; this is your B hat. So, adjoining this thing this columns from the matrix A J minus I x from this matrix, I have been able to get a basis matrix which is also feasible. So, implies that B hat is a feasible basis matrix and that is exactly what I wanted.

(Refer Slide Time: 18:16)



Now, we come and prove the fundamental theorem of linear programming. So, hold your breath.

(No audio from 18:16 to 18:35)

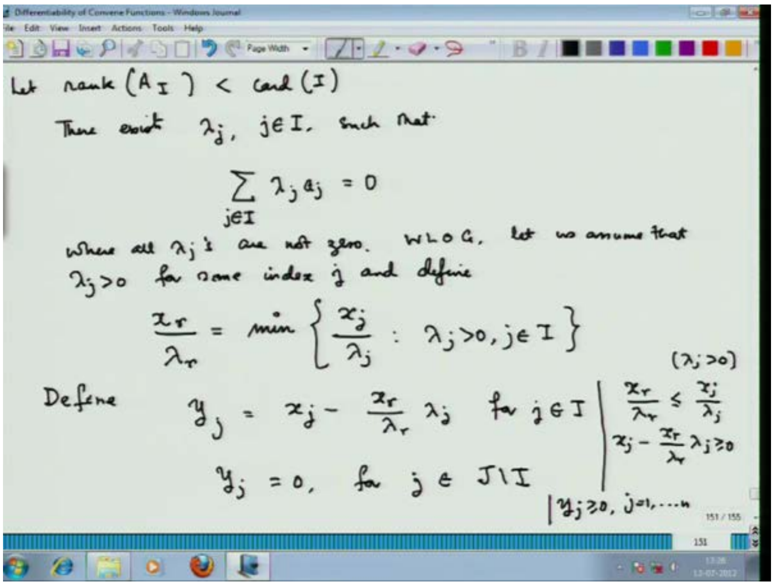
Let us that is the theorem; let us consider LP - a standard linear programming problem. If there exists x element of C, there exists x bar element of C which is a bfs. See we know is a set A x equal to b and x greater than equal to 0 - the feasible set; x bar is a bfs. If there is an optimal solution, then there is an optimal basic feasible solution. This two are the fundamental things. Once you know this, this is exactly the simplest method; this is simplest method will expert this idea.

So, we start again the proof those who are intimated by the fact that so much proof is done has to realize that what we are doing is a piece of beautiful mathematics, and you in mathematics you need to demonstrate the statements you make. So, proof is if you do not want to (( )) proved, do not (( )) proved, the French could (( )). So, you are demonstrating what you have said right. Like giving a proof in the quote of law, so it is to the mathematicians you are basically doing the same thing. You are giving a proof in giving a demonstration on what you are telling that he has, what I have said is correct in the frame work in which we are working. Part 1 - I am proving this part now.

So, let x element of C be given. So, I will given you this. Now, x would have some 0 part may be have all 0 part, may be have it may be all the components has 0, all the component has strictly greater than 0, some greater than 0, some non-zero. So, we assume the most general case and we did. Instead of writing I x now, I will write I because if I continue to write I x the things will get to much complicated right. So, I am just writing I, because you know that this is this I will correspond to this particular x that is all J in capital J which consists of 1 to n which we have already noted earlier. So, it is an index of those components of x which has positive value right. That is if this is true and if this does not cover the whole of j, if i is not equal to j, then x j is equal to 0 if J is element of… So, it might be that either this is empty or this is this is empty or that is immaterial.

So, now I can partition A the matrix again into two parts, again I am assuming the first part first columns, I columns corresponding to the indexes in the set I or in the first part of the matrix that is just assumption we can make without loss of generality. So, I have broken it up into. Of course, you have to say it into it is more correct to put an x here, but just I am not doing, because the symbols would get quite complicated. So, just to avoid the complications in the symbol, but if you want to write I x there is no harm if you write I x. If rank of A I or A I x is nothing but cardinality of I x or I, then x is a bfs and we are done.

(Refer Slide Time: 23:03)



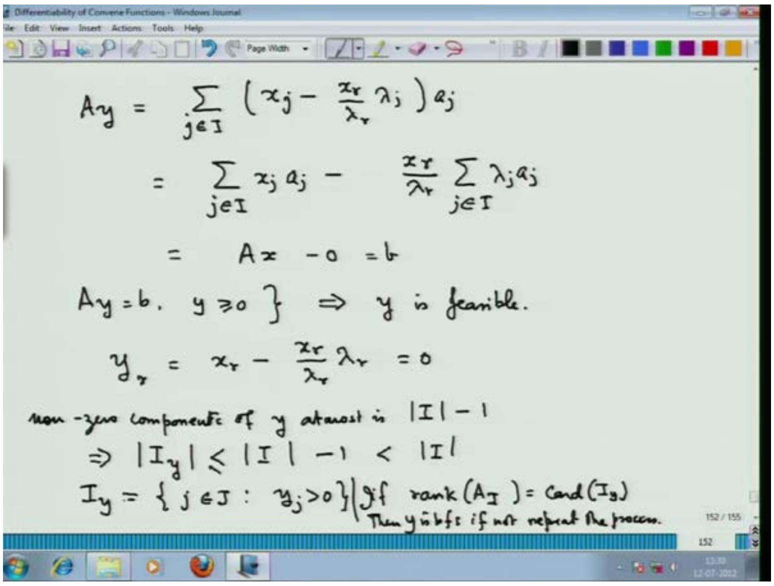
Interesting parts starts when A I x or rank of A I; now, let rank of A I is strictly less than the cardinality of I. This is the fun part, because this will this is the part where you know you are not having what you want, you know you know that x is no longer a bfs, but somehow I have to generate a bfs so let so let us see. I have to construct a point from the given data that will give me a bfs. So, what happens?

Now, rank of A I is strictly less than I which means that there exists lambda j, j is element of I that is corresponding to the those components those columns; such that rank of A I is not cardinality of I. So, rank of A I is a maximum number of linearly independent columns. Now, if I take so, but here there are I columns. So, the number of maximum number of linearly independent columns here is strictly less than the number of indexes in I. So, if I take all the columns in this, they form not a linearly independent set, but a linearly dependent set. As a result of which I can have this where all lambda j(s) are not 0. So, again without loss of generality, because this is 0 on both side you can just change sides.

Let us assume that lambda j is strictly greater than 0 for some index j and define this process. This is called the pivot what I am going to write down now is called the pivoting process and this is but is used in the simplest method or in the Gaussian elimination in for in that case. I should write r may be uniformly, so I will write x r divided by lambda r, this. So, of all j, x j is j element of i, x j is greater than 0, but lambda j could be greater than 0 could be not. But you have set that the there is only some index for which this is greater than 0. So, there may be more than one also. So, you collect all those ratios and take the minimum of that. So, because there only finite number of such elements; the minimum would be one among them.

Now define, so now I am constructing my bfs y j as x j minus x r by lambda r lambda j for j in i, and y j you set as 0 for j element of… You can keep on putting i x that is no problem, but just I avoid complicacy, so up to this it is clear. Now, let us see, what is a nature of y(s)? If you observe here x r by lambda r whenever lambda j is strictly bigger than 0, this is less than equal to x j by lambda j, this is the case when lambda j is strictly bigger than 0. So, you can immediately write again put the lambda j here and take it on the other side. So, I will have x j minus x r by lambda r lambda j to be greater than equal to 0. Now, if lambda j is equal to 0 less than equal to 0 right; lambda j is equal to 0, this is x j. So, x j is anyway greater than equal to 0, because x j is a feasible point. And if lambda j is strictly less than 0 then this would be negative. This would be negative, so negative into negative positive. So, the whole thing would be non-zero. So, what I conclude from here is that y j is equal to is greater than equal to 0 for j is equal to 1 to n. So, the first step of y(s) feasibility is proved, an important step.

(Refer Slide Time: 28:12)

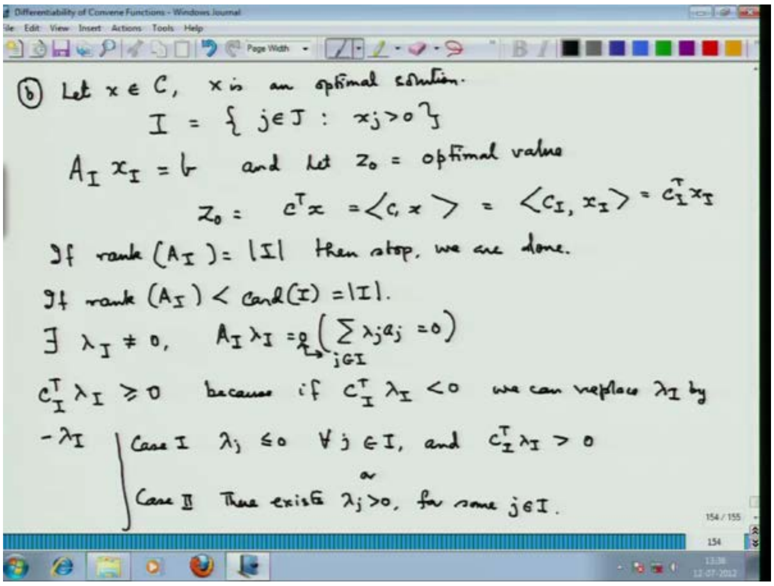


So, now what I am going to check is I will going to write A y. Now, A y is what? So, I can divided into two parts; j belonging to i and j belonging to capital J, but not to I. But for that y j(s) are zeros. So, A y is nothing but so j belonging to I now and that y j is nothing but x j minus so a… So, we have just writing down matrix multiplication. I would not go on explicitly how do you write things in matrix multiplication, because I am assuming you know this basics stuff. So, this is what you would have and then you would have x r lambda can be taken out, because it is fixed some minimum value. But you see this, this part lambda j, this part is equal to 0, but x j a j this part, because it is feasible right and j, j is only a non-zero part, so which means this is nothing but A x. So, A x is equal to A x minus 0, but A x is equal to b. So, A y is equal to b and y is obviously greater than equal to 0 what we have proved earlier implying that y is feasible.

Now, observed what is the value of y r? y r is x r minus x r by lambda r. So, how many positive components y has? It has definitely the positive components of y, so non-zero components of y at most is… There could be others which have the same value as x r, so x. So, this would be at most is now it cannot be more than this. So, this implies that if I denote the new set I y the set of y is for set of the components of y is which are strictly greater than 0, then this is strictly less than I y is strictly less than this, because this is maximum this is equal to this is equal to less than equal to this and strictly less than this.

But I y if you are not feeling what I am writing, I y is nothing but j element of J such that y of j is strictly bigger than 0. So, if now if rank A I is equal to cardinality of I y, then y is a bfs if not repeat the process. Then y is bfs if not repeat the process; stop here and think what it is. It is just like an computer program an algorithm, you take certain steps, you compute certain thing; if this happens you stop the problem, stop the program, if not repeat the problem. So, it is basically a repeat until loop or a do while sort of loops. So, do this sorry not do while, is repeat until do, repeat this process until this happens. So, basically if you observe this, this whole theorem gives you and a very algorithmic is of a very algorithmic flavor. So... So now, we are going into the second thing. So, by repeating this argument finite number of times we will finally get a bfs of course, because we have only finite number of components. Now, how we will get the equality? You have to think of it, I will leave it to you as homework.

(Refer Slide Time: 32:25)



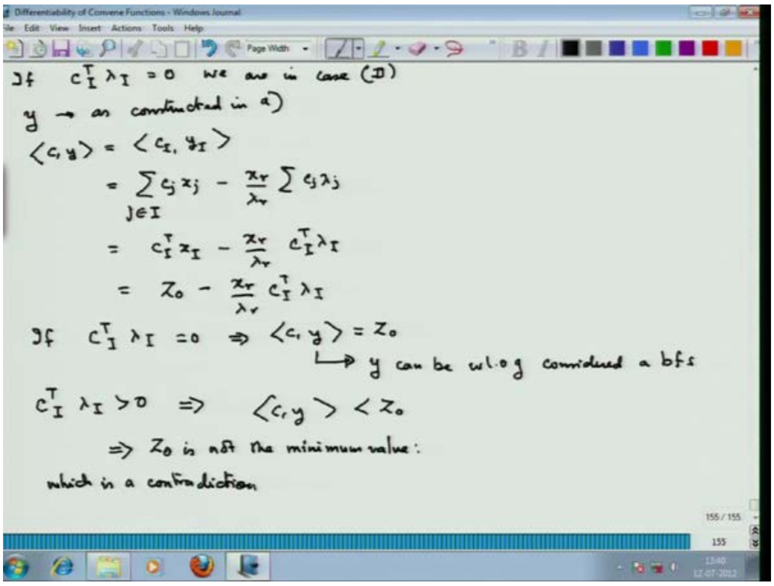
Now, we will go to the major part that part b. Let x be element of C and x is an optimal solution; that is good. Now, again I break up for x, I know that this is a solution. Now, then A I x I, so x I is the part of x whose all components are zero. So, I am formed a small new vector with all non-zero components of x strictly bigger than 0 components of x, and that is of course equal to b. And let z naught is the optimal value; of course, it is a finite optimal value, we are not now that is obvious.

So, z naught is equal to c transpose x equal to c x, but in other words it can we written as c I that is only those components of the vector c corresponding to this components in I x I or you can write if you want, I am giving many, many ways to write the same (( )) thing. So, A is of course partition that is I those columns corresponding to the indexes I and those columns which are not. Again again you see there is an algorithmic flavor. So, this is the first step, if rank of A I is I stop, if not do. So, repeat until this is coming. Basically, the new matrix that you are getting it is assigned to the in the spot of the whole matrix. So, the whole thing has an algorithmic flavor, and this is exactly the simplest method. The beauty is that the mathematics itself is generating the algorithm.

(No audio from 34:15 to 34:24)

Then stop we are done; that x is itself is a bfs, if not then continue; strictly less than the cardinality of I which is also written as this. Now, again, so there exist, because now if I take all the columns of A I they are linearly dependent. So, lambda I is a vector whose components corresponded corresponds those indexes in j which are in I. So, basically I am writing… So, this thing is nothing but writing this. So, this is more a matrix way (( )) compact way of representing this, same as this; this is same as this. See I can always considers c transpose I lambda I to be greater than equal to 0, because if C transpose I lambda I is strictly less than 0, we can replace lambda I by minus lambda I; because the same lambda minus lambda we will we will also work for this. So, now we will divide the whole thing into two cases. So, let me just write down; case one, case two; case one is this. You might be (( )) I am writing this for all j element of I, and this or the case two. So, this is a this is the all 0 and also this is occurring and…

(Refer Slide Time: 36:53)

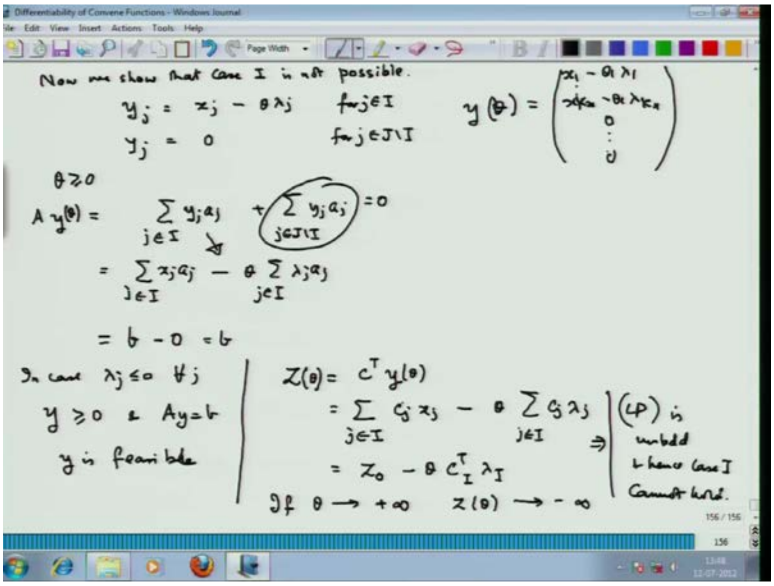


Note if c transpose sorry c I transpose lambda I is 0 then we are case two, because sign of lambda I is immaterial. Suppose I have this, it does not matter, if all of them on negative also I can take the whole thing on the other side and make all of them are positive or if I want to make some positive I can make some positive. So, we can always get, so if this is 0, we are in case two, because the sign does not matter; how did I write it case 100 (( )), so case one, so we are in case two right.

Now, we will consider the expression, we will consider a y as constructed in a; y as constructed in case a. So, then we will compute the c of y that is c I y I, because on the other part is 0 you know, it will be same as c j x j and j element of y minus x r lambda r c j lambda j. So, basically you have c I transpose x I minus x r lambda r summation or you can write it more compactly, this is nothing but c transpose I lambda transpose I. Now, this if I this I know to be 0, now this is nothing but the objective value z z 0.

Now, if, is equal to 0, then it implies that c of y is equal to z 0. Now, this y can be assumed without loss of generality by looking at the case a, y can be assumed to be the bfs. Because y has been constructed, if it is not a bfs we will again apply the same thing that we applied on x on y and get a bfs. So, finally we will get some y. So, let us take this y to be the bfs. So, y can be without loss of generality considered a bfs. So, c y is equal to z 0. Now, what would happen? If c transpose I lambda I would be strictly bigger than 0, then this would be a strictly bigger than 0 quantity which means this would imply c of y would be strictly less than z 0, but y is feasible that we have already proved are in the part a. This would imply that z 0 is not the optimal value which is a contradiction.

(Refer Slide Time: 40:35)



Now, we show that case one is not possible, because if case one is possible will show that the problem is unbounded. So, let us construct some y, but this is not the y that we constructed in a, we are now constructed in it like on our own.

(No audio from 41:01 to 41:13)

So, this is the standard construction, this theta is just a non-negative quantity, I have nothing to do with the original one there. Now, A of y can be written as summation in the same way y j a j plus y j a j whereas partition in the indexes over i and j not in i. So, this will I am applying for j here. So, this part goes to 0 and so what I will have is summation x j a j j element of i minus theta summation lambda j a j j element of sorry j minus not j minus i is gone, so it is j minus i. Now, basically this part goes to 0. So, I am just now analyzing this part and this part takes into this two. Now, this is nothing but b and this is 0 which you know. So, this is b. So, now if theta is positive and lambda j, because in this particular case lambda j is so in the case one, so lambda j is chosen to be negative in case one lambda j is less than equal to 0 for all j. So, means this is becoming positive; negative-negative positive. So, this is anyway positive right and y j is any the this part is 0. So, y y vector in this particular case, which of course y, y should write this y to be depending on theta. So, this y is greater than equal to 0 and A y is equal to b, so y is feasible.

So, let me try to compute the optimal value with this. So, let me compute the Z c transpose y which will be summation j element of I c j x j minus theta times summation c j lambda j. Now, in case one, c j lambda j had been consider to be strictly bigger than 0; so this is strictly bigger than 0, this is greater than equal to 0. Now, this is my z 0 minus theta times c transpose I lambda I. Now, this is strictly less than 0. Now, this thing depends on theta, so z naturally is depending on theta, you can put as z theta does not matter, if you put y in this case, this whole all this components adjoin of into a vector for y theta.

So, if theta goes to plus infinity, because this is now for the case one, this is now positive, the whole thing is negative. So, if theta goes to plus infinity which I can obviously take this whole thing goes to minus infinity. So, if theta goes to plus infinity, so moving along this direction z theta goes to minus infinity. So, if the case one holds I can show that I can create feasible elements y theta. So, I can take y theta to be a vector consisting of this of the form x 1 minus theta 1 lambda 1 x. Suppose I has k x number of elements cardinality of I A x minus theta 1 lambda k x and then 0, 0, 0; this is my y theta, and this is this y theta I am operating on here right. This you can write this also as y theta if you; does not matter. So, what I am telling there? I can again create a sequence again show that if I increase theta then I can get a sequence of points y theta which at which on which the objective values keep on decreasing and decreasing and decreasing and decreasing without bound.

So, this would imply LP is unbounded. So, case one cannot hold, because I have assume there is an optimal solution. This would imply LP is unbounded and hence case one cannot hold. So, this means only case two holds and we are done that there is optimal bfs, and with this I end my talk today and I would request you that when you see this on the You Tube repeat the run once again and carefully go through the arguments. Thank you very much.