Convex Optimization

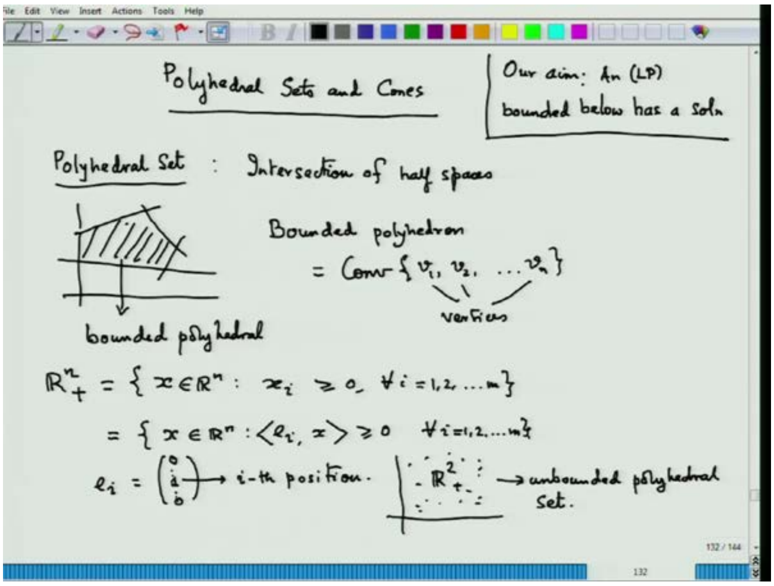
Prof. Joydeep Dutta
Department of Mathematics and Statistics
Indian Institute of Technology, Kanpur

Lecture No. # 22

Welcome to this course on convex optimization once again, and to this parallel little sub course which I am calling the pleasures of linear programming, because this is something which we can handle so well. And tell you a lot of things there is beautiful convex geometrical structures involved. So, in the last class or the last lecture we had shown that every BFS or basic feasible solution of a linear programming problem in a standard form, can be corresponded with a vertex; that is that corresponds to a vertex on the convex polyhedral and vice versa.

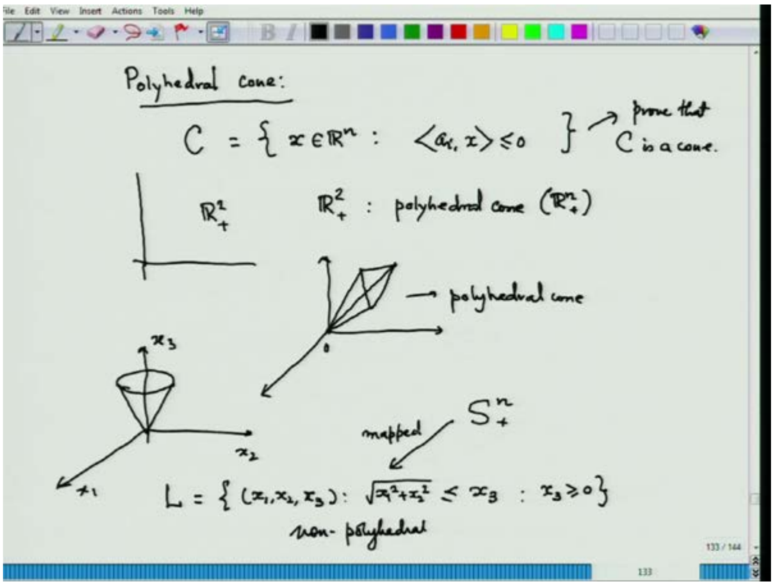
Now, what we are going to show that. So, every vertex is corresponding to some BFS; what we are going to now show that, if I know beforehand that my original problem has a solution, has a lower bound - lower bound is immediately guaranteed by the feasibility of the dual which we had learnt by through weak duality, which we had learnt earlier. So, once I have a lower bound, I can prove that it is a solution, and that solution would be one of the vertices. But in order to prove such a fact, we need to know certain more facts; so, we recall certain things for example, we recall polyhedral sets. Polyhedral sets are very very important, because polyhedral sets are the basic structure of a linear problem, because a linear problem is nothing but minimization of a function, linear function over a polyhedral sets.

(Refer Slide Time: 02:02)



So, we come and have a recollection of the polyhedral sets. So, a polyhedral sets is intersection of the half spaces, and it could be bounded or it could be unbounded; it is bounded for example, here. So, any bounded polyhedral set which is also called a polytope, can we written as a convex hull of its vertices. So, you have v 1, v 2, v 3, V 4, v 5; in this case this is a convex hull of this five vertices. But if there could be unbounded polyhedral sets two which has a vertex, but this whole thing is not a convex hull of this vertex naturally. So, for example, R n plus is an unbounded polyhedral set.

(Refer Slide Time: 02:50)



Now, we will go into the notion of what is called a polyhedral cone. So, polyhedral cone C can be written as set of all x in R n such that a i x is less than equal to 0, there is b i is equal to 0. Polyhedral cone for example, is like this. If you take R 2, R 2 plus or R n plus; R 2 plus is a cone and is a poly hull is polyhedral at the same time. So, R 2 plus is an example of a polyhedral cone. Similarly, R n plus is also an example of a polyhedral cone. Now, you can easily prove that that set C is a cone prove.

(No audio from 03:41 to 03:50)

On the other hand, there is is an also an another definition which came up a literature; as a finitely generated cone. Let me just draw a few more polyhedral cones, and three d for example, if I take.

(No audio from 04:09 to 04:17)

This is an example of polyhedral cone, because any point, you see these phases are the sides of the cone actually a hyper planes which are passing through 0. So, this is an example of polyhedral cone; this is the polyhedral cone of course, it is coming up like this. So, going up to infinity, but we do not bother about that in the drawing. But a very important cone like this, like the (( )) cone in three dimensions.

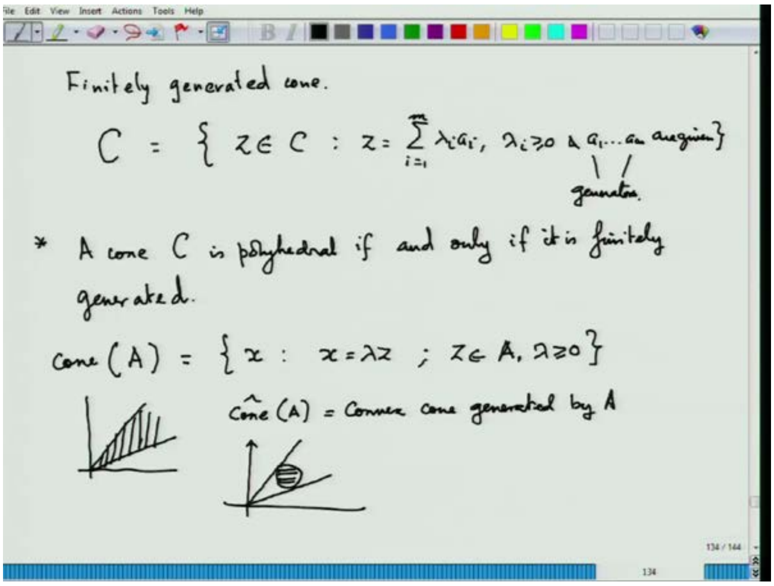
(No audio from 04:48 to 05:00)

Also called the second order cone; this cone sorry I had write it has a lorange cone l, as a huge impact in something call second order cone programming which will come later on. So, if I take l here, and the let me describe this it consists of all x 1, x 2, x 3 such that root of x 1 square plus x 2 square is less than x 3, where x 3 is greater than equal to 0. So, this is a non-polyhedral cone, because here I have quadratic inequality, not a linear inequality. So, it is not a polyhedral cone. So, this is non-polyhedral. In fact, the set of all positive semi definite matrices S n plus can be mapped in a natural way to this cone.

So, in that way many problems which are actually semidefinite programming problem, can be posed as second order conic problem about which we will come in detail later when we study semidefinite programming.

So, this is an example of a non-polyhedral cone, but if this be in a convex cone; if you look at any point in this polyhedral cone, you will observe that any point can be written as a positive linear combination of these 1, 2, 3 vectors right. So, these vector are essentially called the generator of the cone, and that lead to the definition of notion called finitely generated cone.

(Refer Slide Time: 07:03)



A cone C is finitely generated, if any element z in C can be written as z is equal to summation lambda i a i, say a where lambda i is greater than equal to 0, and a 1 to a m are given. So, this a 1 to a m this things are called generators of the cone. Now, interesting fact or a very very important fact in convex geometry is the following; a cone C is polyhedral, if and only if its finitely generated and that is the fascinating thing.

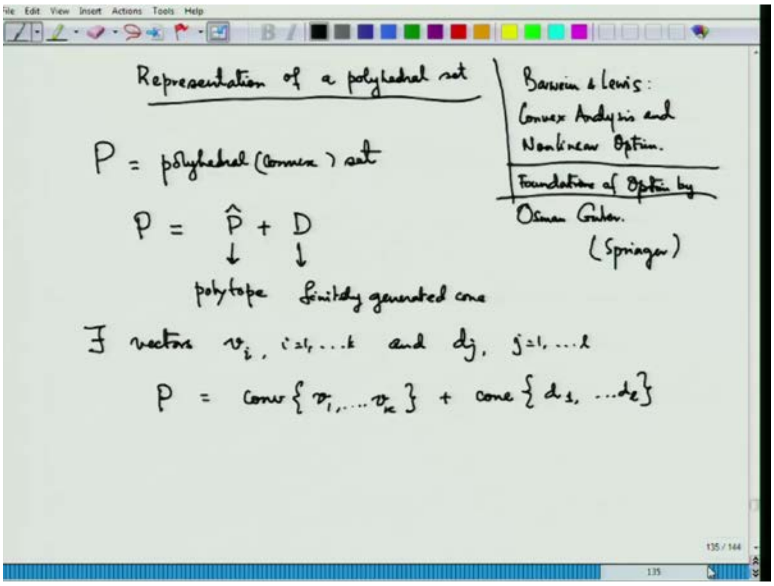
(No audio from 08:00 to 08:13)

Now, also I would like to recall, before you have very familiar notion of a cone generated by a set a, this consists of all the points x; such that x is equal to lambda z where z is an element of A, and sorry and lambda of course, is greater than equal to 0 this called the cone generated by A. Of course, you can also define the convex cone generated by A. So, for example, if you take these two lines only, and call these two lines the union of these two lines has A, then you can these two lines these to fork is what is the cone generated by A.

But the convex cone generated by A is the convex hull of the cone generated by A. So basically... So, I will say like this convex cone generated by generated by A. Now, if you take a full dimensional convex set like this; then you always bound to get a convex cone when any cone that is generated like this, any cone that is generated in this fashion has to be a convex cone. So, these notions are slightly different; and now why are we all doing this. Because once we have this idea, we would be able to make a interesting representation of any polyhedral set, and this representation can be carried over to the feasible set of a linear programming problem, and that would that would lead to what we want at the end. My next important interesting result is representation of polyhedral set.

(No audio from 10:23 to 10:37)

(Refer Slide Time: 10:24)



Now, once I want to how do I represent it. The interesting fact is that any polyhedral set can be represented as the Victoria sum of two sets. One of them is a convex poly tope, and another is a finite regenerated cone. So, the cone takes in unbounded thing. So, when you are talking about just the polytope, then this cone is nothing but the 0 vector. So, if P is when we say P is a polyhedral set; obvious its convex which I am not writing. Polyhedral set then P can be written as P hat plus D. a P hat is a polytope which is another name for bounded for bounded polyhedral, and if this is a finitely generated cone.

So, going back these a i are called generator of this cone. So, sometimes it is convenient to write finitely generated cone, as a cone generated by the vectors a 1 to a m, in could be any number it is not fix, some fix number.

Now, which means that there would exist vectors v i from 1 to k, and d j say j from 1 to l such that P is the convex hull. So, any polytope can be represented as the convex hull of its vertices v 1 this opposite as k vertices, then v 1, v 2, v k plus the finitely generated cone; generated by the generator d 1, d 2, d l; this facts some of proofs (( )). Now, you might ask me what is the proof of this, we are not getting into the proof of this; because of to prove this, it would force us to prove the fact that every polyhedral cone is finitely generated and vice versa; which is a time taking process, and we will not getting to this, because again we have to remind you that those who are in mathematics they can go, and we can tell you a book which you can read.

See if you want to know the prove, and the detail of this I will suggest you two books; Barwein and Lewis which I have already mentioned earlier, convex analysis and non-linear optimization; and second book is foundations of optimization by Osman Guler recent one.

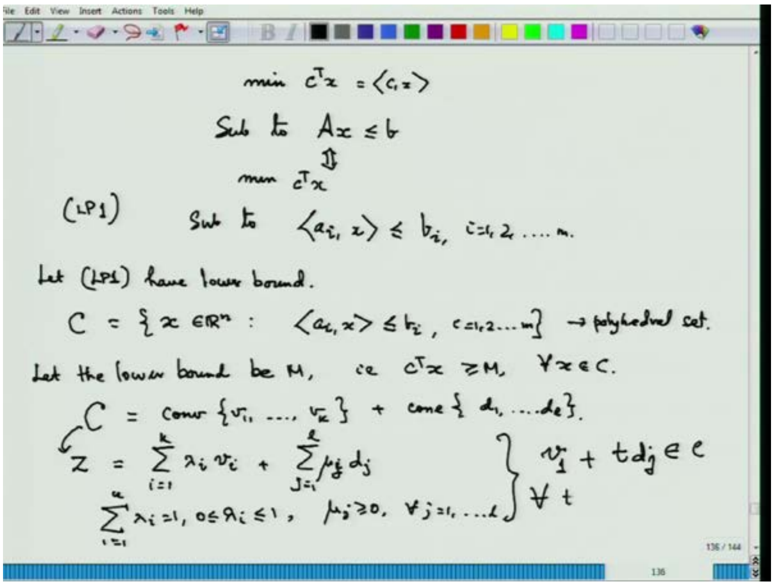
(No audio from 14:01 to 14:13)

Now, this by springer, this also by springer. So, a publisher is springer; costly books by the way do not worry, none of them as Indian addition. So far so, you need to go to the libraries.

Now, once I know this how can I use it, to prove what I intend to prove; that once a linear programming problem l p has a lower bound, this is just fascinating, it will always have a minimizer. Now, what I would do instead of the proof, I give you is due to Osman Guler from foundations of optimization, and I would like to state that the prove is given for a optimization - linear optimization problem in a much more different form than you have. So, what I have what Osman Guler has done is to consider a problem of this form.

(No audio from 15:10 to 15:21)

(Refer Slide Time: 15:14)



He has not put in any restrictions on the variable. So that, the most general form with an inequality. So, I can write this again into an equivalent form, if each a i is a vector representing the roles of A, and i can write this as say A is same in m cross and matrix and all those things which you all know quite well. Now, we will assume for this particular problem l p 1 I am calling it, I do not know why I am calling it one, but just calling it one. So, l p 1 might be take this problem l p 1. So, let l p 1 have a lower bound. Now, look this set C here.

(No audio from 16:27 to 16:42)

This is the polyhedral set.

(No audio from 16:44 to 16:51)

Now, this very very important to understand that this set could be bounded could be unbounded; if its bounded its fine, if it is not bounded then we have to use the representation that we have just seen. Now, l p has a lower bound. So, let the lower bound is m which means that is C transpose x is less than equal to M, for all x in C; that is exactly what is the precise, one is the precise meaning of the lower bound.

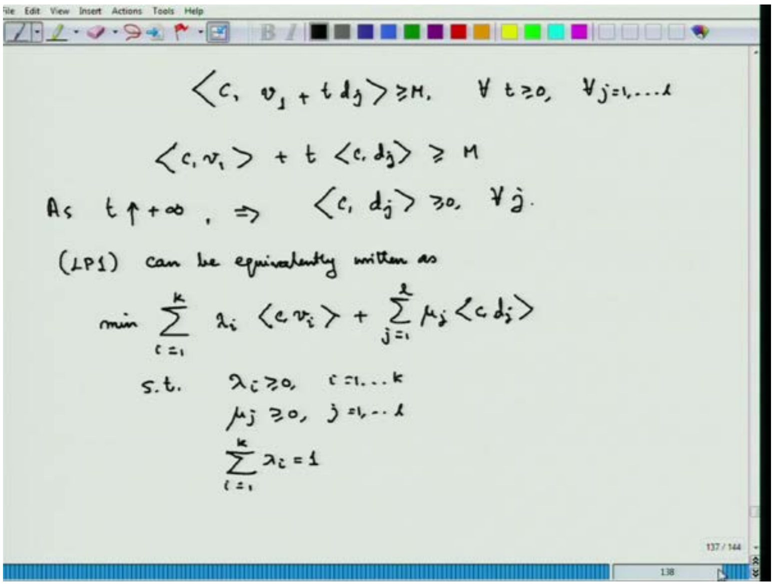
Now, if that is the case let me put now C by the previous result, this result can be decomposed in this way, I am just checking this decomposition. So, C can be written as convex hull of vertices v 1, v 2, v k plus the finitely generated cone generated by d 1, d 2, d l.

Now, once I know this what could I possibly do. So, if I take the convex hull of v 1. So, any element here, take any element z here, z in C z in C; this can be written as lambda i v i phi is equal to 1 to k plus mu i sorry mu j d j, j is equal to one to l; where your summation lambda i, i equal to 1 to k is 1, and lambda i is obliviously lying between one and 0 that is the convex combination and mu j is greater than equal to 0; for all j equal to 1 to l. So, that is exactly what you want. So, any j can be represented there would be some lambda is some mu j is any lambda can be represented like this. Now, what would happen suppose, I take lambda equal to 1; say lambda one equal to one and put all the other lambdas to be 0.

And I take some for some j. So, take a j and corresponding first take, corresponding to that j put mu j as some number, and put all the other mu j is to be 0. Then what from here what can I conclude that v 1, you could take any i also. So, I am just taking with v 1 simplicity. v 1 plus t of d j; for all t and all j means every... So, what you do, you first take one j say say d 1 stay put. So, v 1 plus take any fixed t whatever you want, say v 1 plus t of d 1 is element of C; v 1 plus t of d 2 is element of C n. So on.

So, this is an element of C for all t bigger than equal to 0, and for all j is equal to all j running from 1 to l. So, this is simple just you have to note this representation, put their proper values, put here basically putting all lambda is equal to 0 except lambda 1 which you put 1, and here take any t whatever you want, take any j you want the remaining all you put puts to 0; and then then by this is of course, an element of C just by this representation.

(Refer Slide Time: 20:58)



Now, once I know this what can I do; now, because this in C by my very definition of bounded the bound m.

(No audio from 21:09 to 21:11)

So, this is what I have, what I would have this simply just go back and recollect this definition; this is the meaning of the lower bound. Now, I can write this as C of v 1 plus t times C of d j is bigger than equal to m. Now, as t tends to infinity; now let me see what would happen as t tends to infinity. Now, if C of d j if one of them is less than 0, negative and I can make t go towards infinity. So, there will be a minus component which will become bigger and bigger and bigger and bigger.

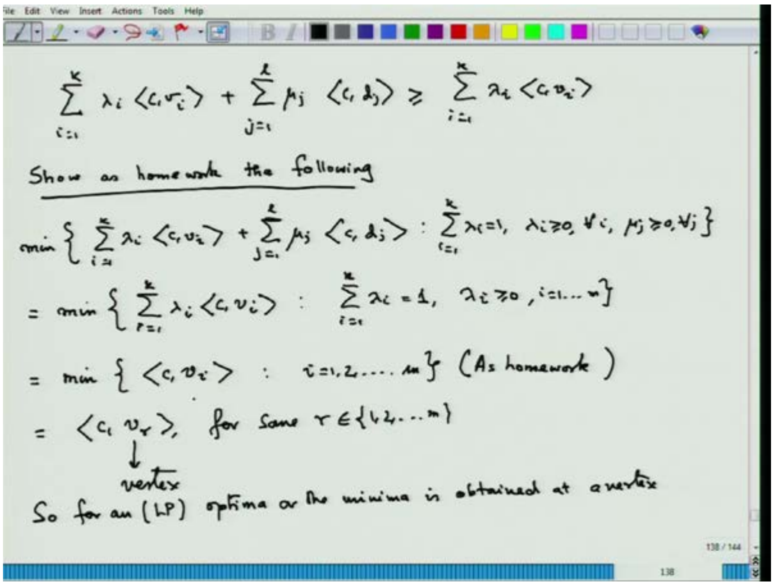
So, finally, it will just over power this C v 1 and can go into completely into the minus domain. And can the value can continue to decrease the value of this will continue to go down, because C v 1 is fixed, and it can go just below m. So, as... So, as t tends to infinity, because of this bound; it would imply that C of d j has to be greater than equal to 0 for all j, this is what I have. Now, once I know this fact, I can look into the original problem in a slightly different way. So, I have this problem. So, I can now write this problem as following.

Now, I have any x, any x here, is belonging to this set which I have an any element in this particular set C is represented like this. So, I can have that L P 1 can be equivalently written as.

(No audio from 23:16 to 23:38)

This where subject to of course, lambda i greater than equal to 0 for I equal to 1 to k, mu j is greater than equal to 0 for j equal to 1 to l, and the summation that must be equal to one. So, the problem L P 1 can be equivalently written like this. Now, knowing that this is greater than equal to 0, what should I get; I should get the following. I should get, because if what is happening is the following; and if you look at this, to this quantity lambda i C i we have added a positive quantity, non-negative quantity. So, this quantity is actually bigger than the previous term.

(Refer Slide Time: 24:22)



So, which means that summation i is equal to 1 to k lambda i C v i plus summation j is equal to 1 to l mu j C d j, because now this is the non-negative quantity; then if I add some non-negative quantity to a given number, I actually increase that number. So, this whole thing must be bigger than summation i is equal to 1 to k lambda i C v i. Now, I will allow ask you to prove the following, show as homework the following, show as homework the following the minimum value obtained by solving this problem.

(No audio from 25:40 to 26:14)

So, this minimizing this is same as minimizing very important, you have to minimize on two variables. So, here observe the minima here is transform from x to this two variable lambda if's, and mu if's lambda in r k plus lambda in r k and this is in r l. So, basically we are changing over from minimization on x to minimization on this multipliers. So, just let me oh my god…

So now, what I can prove, because of this fact; and because of this is greater than equal to 0. Then, this thing is nothing but…

(No audio from 26:58 to 27:16)

Now, this is quite intuitive, because this quantity is nothing but this quantity with putting mu j is equal to 0 here. So, I can always put mu j is equal to 0, because mu j is greater than is equal to 0 it is your choice whether you put it equal to 0 or greater than equal to 0 it is up to you. Now, hence whatever way we try the functional value has to be always bigger than this, and which is also one of the functional values. So, the minimum achieved by minimizing this functional values will be obviously less than this one.

So, but again the fact that we are minimizing over this objective would actually bring in this; it is just a simple writing down thing, using this put the minima on both sides then noting that; that minima of this has to be less than, the minima of this has to be less than this. For whatever lambda i this has to be less than equal to this. So, the minima of this have to be less than equal to this for whatever lambda i. So, you can minimize. So, you get inequality, but so, I will not do this work for you, your supposed to do this equality.

First note, once I have this I can operate minima on lambda mu minima on lambda mu here mu is irrelevant. So, minima of this is bigger than the minima of this fine. But now, for whatever lambda mu you take the minima over this the indium value is less than equal to the infimum value of this is less than equal to this, because this is nothing but a particular feasible choice lambda if's, some lambda if's and mu joss are all 0.

So, minimum value of this is; obviously, less than this. The minimum value exists indium exists, because we are assumed the lower bound of L P 1. So, but then for whatever lambda I choose combination, the minimum value of this is always less than this. So, the minimum of a lambda is always bigger than the minimum over this, and hence these two are equal which is absolutely simple. Now, the question is noting that summation lambda equal to 1, show as homework this is nothing but minimum of C v i, i is equal to 1, 2. So, you are this is exactly linear programming's game, that you are computing the objective function value at the vertex and taking the minimum one. So, show this equality as homework.

You see just try it out; it is going to be fun. And you would see that mathematics really works, and that is why it is so beautiful. So, what I have proved that the minimum value of L P 1, because now here I have only finite number of them m. So, one of the C v i values say C v j is the minimum; say this value is equal to C v j say or C v; say let me take C v r - for some r element of 1 to m which is; obviously, true just you have finite number of finite some few numbers, you have to choose the minimum; we will choose the minimum.

So, what happen the minimum of L P 1 which is this is obtained at this, is actually this value where v r is nothing but a vertex. So, for an L P. So, for L P 1. So, in general I am writing for or L P problem or L P in the standard form. So, for an L P optima or the minima in this case is obtained at the vertex, obtained at a vertex.

Now, these vertices these vertices v 1, v r these are also elements of the set C, but these are these are vertices of a polyhedral cone. So, actually what how do you represent the polyhedral set; basically, you take the vertex of the polyhedral sets, and make a you take the vertex of the polyhedral and make a polytope; taking the convex hull. And then add to it the finitely generated cone. So, these are actually vertices of the original polyhedral naturally; these are actually vertices of the original polyhedral. So, once you know this. So, for an L P optimum or the minimum is obtained at a vertex, and thus every optimum or every minimum is a b f s by the previous stage result.

Now, before I end this today's lecture, and after the next lecture we will start with the simple method which we will do in more of the non-linear programming style; now, what I have done - I have proved this fact for L P 1. Now, I have something more to tell you, can you prove this fact for L P, prove the above result for L P for the standard form. So, here L P is the setting of C would change is the same thing, you just do not have to worry about homework's, small modifications are needed which you should try to carry out, because it be fun and may be a good idea is that; this when you take this, these actually are vertices of C, and can you prove that these are actually vertices of C.

So, prove proving this to be vertices of C is not a very bad idea; that actually works that. These are not just some arbitrary points you have taken which are vertices of this polyhedral, but this is also they are also vertices of C. So, I can show that you cannot have two elements which are different from each other, and whose convex combination with lambda between 0 and 1, would give you v 1 or v 2 or v 3 or v k. So, that is something very very important.

So, if that is so, then they are those things are all equal. So, that can be done quite quite easily quite quite easily. So, let us not get too much about with that, but it is a good idea to try it out that, these are also actually vertices of set C. So, we have proved quite bit stuff today and tomorrow's class or the next class would be on the simplest method. So, with this I would like to end today's lecture, and because if I want to start simplex method, we will get into too much complications today; and we would not be able to finish the foundation, because foundations of simplex method or quite, heavy to lay down. And unless I do it start, and complete it is meaningless to just start and leave it off for today.

So, it will be good if we start in a separate lecture. So, what we have now is quite a broad view of optimization; you have learned about lagrange multipliers, saddle point conditions duality theory, convex analysis as well as a focus on as as well as, what we are doing now focusing on the most important class of problems; the semidefinite programming problem and the linear programming problem.

Semidefinite programming problem is so, so powerful which are extremely important applications at present, into into understanding quadratic programming, non convex quadratic problems in understanding polynomial optimization problems; so, we would thus focus ourselves on these two classes of problems, that would lead us to much better understanding a modern convex optimization theory, because modern convex optimization theory is essentially, the story of semidefinite programming; though they are many many other things which like bundle methods and all those things.

So, whatever we find time, we will try to push into in these forty lectures. One can think of an advance state of lecture later on, but let us just concentrate on these two important aspects, and do them in detail. So, that especially the engineering people here, who are listening to this lecture should know that L P, and semidefinite programming (( )) SDP or semidefinite programming per say is extremely important from the point of view of applications in engineering; extremely important. I think, so is important on my part to really concentrate on these two aspects of the subject. Thank you very much.